CERN/LHCC 02-26
CMS TDR 6
December 15, 2002

CMS

The TriDAS Project

Technical Design Report, Volume 2:

Data Acquisition and High-Level Trigger

Also available at http://cmsdoc.cern.ch/cms/TDR/DAQ/daq.html

CMS TriDAS project		
Chairperson Institution Board: Paris Sphicas, CERN, MIT and University of Athens, Paris.Sphicas@cern.ch		
Trigger/DAQ Project Manager	**Trigger Project Manager**	**DAQ Project Manager**
Sergio Cittolin, CERN Sergio.Cittolin@cern.ch	Wesley Smith, University of Wisconsin wsmith@hep.phys.wisc.edu	Sergio Cittolin, CERN Sergio.Cittolin@cern.ch
Trigger/DAQ Resource Manager	**Trigger Technical Coordinator**	**DAQ Technical Coordinator**
Joao Varela, LIP/Lisboa Joao.Varela@cern.ch		Attila Racz, CERN Attila.Racz@cern.ch

CMS Spokesperson	CMS Technical Coordinator
Michel Della Negra, CERN Michel.Della.Negra@cern.ch	Alain Herve, CERN Alain.Herve@cern.ch

Editor

P. Sphicas

Chapter Editors

D. Acosta, J. Branson, S. Cittolin, A. De Roeck, S. Eno, S. Erhan, F. Glege, J. Gutleber, G. Maron, F. Meijers, E. Meschi, S. Murray, A. Oh, L. Orsini, V. O'Dell, A. Racz, P. Scharff-Hansen, C. Schwick, C. Seez, L. Silvestris, K. Sumorok.

Cover Design

S. Cittolin

Acknowledgements

The CMS Data Acquisition community wishes to thank all the people who have contributed to the work presented in this Technical Design Report.

We would like to thank our CMS internal reviewers C. Foudas, R. Clare and H. von der Schmidt, for all their help, constructive comments and suggestions. We would also like to thank our LHCC referees S. Bertolucci, J. Nash, K. Tokushuku and C. Vallee, who have followed the project in the past several years and have offered us plenty of support.

Our daily work has received precious help from Shamaila Ahmad, Kirsti Aspola, Madeleine Azeglio, Nadejda Bogolioubova, Dawn Hudson, Delphine Labrousse, Anne Lissajoux, Guy Martin and Marie-Claude Pelloux.

The Spring02 production would not have been successful without the effort of a large team of people running and supporting the production in various ways. We wish to thank in particular the following people: S. Aziz, C. Charlot, D. Colling, M Ernst, A. Filine, R. Gowrishankara, N. Kruglov, A. Kryukov, J. Letts, P. Lewis, C. Mackay, P. Mine, S. Muzaffar, S. Singh, E. Tikhonenko, Y. Wu.

We also wish to thank our collaborators on CMS, and especially the CMS management, for their support and encouragement.

CMS Collaboration

Yerevan Physics Institute, Yerevan, ARMENIA
G.L. Bayatian, N. Grigorian, V.G. Khachatrian, A. Margarian, A.M. Sirunyan, S. Stepanian

Institut für Hochenergiephysik der OeAW, Wien, AUSTRIA
W. Adam, T. Bauer, T. Bergauer, J. Erö, M. Friedl, R. Fruehwirth, J. Hrubec, A. Jeitler, M. Krammer, N. Neumeister[1], M. Pernicka, P. Porth, H. Rohringer, H. Sakulin, J. Strauss, A. Taurok, W. Waltenberger, G. Walzel, C.-E. Wulz

Research Institute for Nuclear Problems, Minsk, BELARUS
A. Fedorov, N. Gorodichenine, M. Korzhik, V. Panov, V. Yeudakimau, R. Zuyeuski

National Centre for Particle and High Energy Physics, Minsk, BELARUS
V. Chekhovsky, I. Emeliantchik, M. Kryvamaz, A. Litomin, V. Mossolov, S. Reutovich, N. Shumeiko, A. Solin, A. Tikhonov, V. Zalessky

Byelorussian State University, Minsk, BELARUS
V. Petrov

Vrije Universiteit Brussel, Brussel, BELGIUM
O. Devroede, R. Goorens, J. Lemonne, S. Lowette, S. Tavernier, B. Van De Vyver[1], W. Van Doninck[1], L. Van Lancker, C. Yu

Université Libre de Bruxelles, Bruxelles, BELGIUM
O. Bouhali, B. Clerbaux, G. De Lentdecker, P. Marage, C. Vander Velde, P. Vanlaer, J. Wickens

Université Catholique de Louvain, Louvain-la-Neuve, BELGIUM
S. Assouak, J.L. Bonnet, E. Burton, B. De Callatay, J. De Favereau De Jeneret, G. De Hemptinne, C. Delaere, T. Delbar, D. Favart, E. Forton, J. Govaerts, G. Grégoire, G. Leibenguth, V. Lemaitre, Y. Longree, D. Michotte, O. Militaru, A. Ninane, F. Payen, K. Piotrzkowski, X. Rouby, O. van der Aa

Université de Mons-Hainaut, Mons, BELGIUM
I. Boulogne, E. Daubie, P. Herquet

Universitaire Instelling Antwerpen, Wilrijk, BELGIUM
W. Beaumont, T. Beckers, E. De Langhe, E. De Wolf, F. Moortgat, F. Verbeure

Institute for Nuclear Research and Nuclear Energy, Sofia, BULGARIA
K. Abadjiev, T. Anguelov, I. Atanasov, J. Damgov, L. Dimitrov, V. Genchev, P. Iaydjiev, B. Kounov, P. Petkov, S. Piperov, G. Sultanov, I. Vankov

University of Sofia, Sofia, BULGARIA
A. Dimitrov, I. Glushkov, L. Litov, M. Makariev, M. Mateev, I. Nasteva, B. Pavlov, P. Petev, S. Stoynev, V. Velev

Institute of High Energy Physics, Beijing, CHINA, PR
J. Bai, J.G. Bian, G.M. Chen, H.S. Chen, Y.N. Guo, K. He, G. Huang, C.H. Jiang, Z.J. Ke, J. Li, W.G. Li, H. Liu, J.F. Qiu, X. Shen, Y.Y. Wang, M. Yang, B.Y. Zhang, S.Q. Zhang, W. Zhao, Z.G. Zhao, L. Zhou, G.Y. Zhu, Z. Zhu

Peking University, Beijing, CHINA, PR
Y. Ban, J. Cai, H.T. Liu, Y.L. Ye, J. Ying

University for Science and Technology of China, Hefei, Anhui, CHINA, PR
Z.P. Zhang, J. Zhao

Shanghai Institute of Ceramics, Shanghai, CHINA, PR (associated institute)
Q. Deng, P.J. Li, D.Z. Shen, Z.L. Xue, P. Yang[1], H. Yuan

National Central University, Chung-Li, CHINA (TAIWAN)
Y.H. Chang, E.A. Chen, A. Go, W. Lin

National Taiwan University (NTU), Taipei, CHINA (TAIWAN)
S. Blyth, P. Chang, Y. Chao, K.F. Chen, Z. Gao[1], G.W.S. Hou, H.C. Huang, K. Ueno, Y. Velikzhanin, M. Wang, P. Yeh

Technical University of Split, Split, CROATIA
N. Godinovic, I. Puljak, I. Soric

University of Split, Split, CROATIA
Z. Antunovic, M. Dzelalija, K. Marasovic

University of Cyprus, Nicosia, CYPRUS
P. Demetriou, C. Nicolaou, P.A. Razis, D. Tsiakkouri

National Institute of Chemical Physics and Biophysics, Tallinn, ESTONIA
E. Lippmaa, M. Raidal[1], J. Subbi

Laboratory of Advanced Energy Systems, Helsinki University of Technology, Espoo, FINLAND
P.A. Aarnio

Helsinki Institute of Physics, Helsinki, FINLAND
K. Banzuzi, S. Czellar[1], P.O. Friman[1], J. Härkönen, M.A. Heikkinen, J.V. Heinonen, V. Karimäki, H.M. Katajisto, R. Kinnunen, T. Lampén, K. Lassila-Perini, V. Lefébure[1], S. Lehti, T. Lindén, P.R. Luukka, S. Nummela, E. Tuominen, J. Tuominiemi, D. Ungaro

Dept. of Physics & Microelectronics Instrumentation Lab., University of Oulu, Oulu, FINLAND
T. Tuuva

Digital and Computer Systems Laboratory, Tampere University of Technology, Tampere, FINLAND
J. Niittylahti

Laboratoire d'Annecy-le-Vieux de Physique des Particules, IN2P3-CNRS, Annecy-le-Vieux, FRANCE
J.P. Guillaud, M. Maire, J.P. Mendiburu, P. Nedelec, J.P. Peigneux, M. Schneegans, D. Sillou

DSM/DAPNIA, CEA/Saclay, Gif-sur-Yvette, FRANCE
M. Anfreville, C. Bouchand, P. Bredy, R. Chipaux, M. Dejardin[1], D. Denegri, B. Fabbro, J.L. Faure, F.X. Gentit, A. Givernaud, P. Jarry, F. Kircher, T. Lasserre, M.C. Lemaire, Y. Lemoigne, B. Levesy, E. Locci, J.P. Lottin, M. Mur, E. Pasquetto, A. Payn, J. Rander, J.M. Reymond, F. Rondeaux, A. Rosowsky, A. Van Lysebetten, P. Verrecchia

Laboratoire Leprince-Ringuet, Ecole Polytechnique, IN2P3-CNRS, Palaiseau, FRANCE
M. Bercher, J. Bourotte[1], P. Busson, D. Chamont, C. Charlot, C. Collard, L. Dobrzynski, Y. Geerebaert, J. Gilly, M. Haguenauer, A. Karar, D. Lecouturier, G. Milleret, P. Miné, R. Morano, P. Paganini, P. Poilleux, N. Regnault, T. Romanteau, I. Semeniouk

Institut de Recherches Subatomiques, IN2P3-CNRS - ULP, LEPSI Strasbourg, UHA Mulhouse, Strasbourg, FRANCE
J.D. Berst[2], R. Blaes[3], D. Bloch, J.M. Brom, F. Charles[3], G.L. Claus, J. Coffin, F. Didierjean, F. Drouhin[3], J.P. Ernenwein[3], J.C. Fontaine[3], U. Goerlach, P. Graehling, L. Gross, D. Huss, C. Illinger[2], P. Juillot, A. Lounis, C. Maazouzi, S. Moreau, I. Ripp-Baudot, R. Strub, T. Todorov, P. Van Hove, D. Vintache

Institut de Physique Nucléaire de Lyon, IN2P3-CNRS, Univ. Lyon I, Villeurbanne, FRANCE
M. Ageron, M. Bedjidian, A. Bonnevaux, E. Chabanat, C. Combaret, D. Contardo, R. Della Negra, P. Depasse, M. Dupanloup, T. Dupasquier, H. El Mamouni, N. Estre, J. Fay, S. Gascon, N. Giraud, C. Girerd, R. Haroutounian, J.C. Ianigro, B. Ille, M. Lethuillier, N. Lumb, L. Mirabito[1], S. Perries, O. Ravat, B. Trocme

High Energy Physics Institute, Tbilisi State University, Tbilisi, GEORGIA
R. Kvatadze, D. Mzavia

Institute of Physics Academy of Science, Tbilisi, GEORGIA
V. Roinishvili

RWTH, I. Physikalisches Institut, Aachen, GERMANY
R. Adolphi, W. Braunschweig, H. Esser, A. Heister, W. Karpinski, S. König, C. Kukulies, J. Olzem,

A. Ostaptchouk, D. Pandoulas, G. Pierschel, F. Raupach, S. Schael, A. Schultz von Dratzig, B. Schwering, G. Schwering, R. Siedling, B. Wittmer, M. Wlochal

RWTH, III. Physikalisches Institut A, Aachen, GERMANY
A. Adolf, C. Autermann, A. Böhm, M. Bontenackels, H. Fesefeldt, T. Hebbeker, S. Hermann, G. Hilgers, K. Hoepfner, D. Lanske, B. Philipps, D. Rein, H. Reithler, M. Sowa, T. Stapelberg, H. Szczesny, M. Tonutti, O. Tsigenov, M. Wegner

RWTH, III. Physikalisches Institut B, Aachen, GERMANY
M. Axer, F. Beissel, V. Commichau, G. Flügge, T. Franke, K. Hangarter, T. Kress, J. Mnich, A. Nowack, M. Poettgens, O. Pooth, P. Schmitz, M. Weber

Institut für Experimentelle Kernphysik, Karlsruhe, GERMANY
V. Bartsch, P. Blüm, W. De Boer, A. Dierlamm, G. Dirkes, M. Erdmann, M. Fahrer, M. Feindt, A. Furgeri, P. Gras, E. Grigoriev, F. Hartmann, S. Heier, J. Kaminski, S. Kappler, T. Müller, C. Piasecki, G. Quast, K. Rabbertz, H.J. Simonis, A. Skiba, A. Theel, T. Weiler, C. Weiser, V. Zhukov[4]

University of Athens, Athens, GREECE
A. Angelis, G. Daskalakis, A. Panagiotou[1]

Institute of Nuclear Physics "Demokritos", Attiki, GREECE
M. Barone, T. Geralis, C. Kalfas, D. Loukas, A. Markou, C. Markou, K. Zachariadou

University of Ioánnina, Ioánnina, GREECE
A. Asimidis, X. Aslanoglou, I. Evangelou, P. Kokkas, N. Manthos, G. Sidiropoulos, F.A. Triantis, P. Vichoudis

KFKI Research Institute for Particle and Nuclear Physics, Budapest, HUNGARY
G. Bencze, L. Boldizsar, P. Hidas, G. Odor, G. Vesztergombi, P. Zalan

Institute of Nuclear Research ATOMKI, Debrecen, HUNGARY
J. Molnar

Kossuth Lajos University, Debrecen, HUNGARY
G. Marian, P. Raics, Z. Szabo, Z. Szillasi, G. Zilizi

Panjab University, Chandigarh, INDIA
S.B. Beri, V. Bhatnagar, M. Kaur, J.M. Kohli, J. Singh

University of Delhi, Delhi, INDIA
A. Bhardwaj, S. Chatterji, K. Ranjan, R.K. Shivpuri

Bhabha Atomic Research Centre, Mumbai, INDIA
S. Borkar, M. Dixit, M. Ghodgaonkar, S.K. Kataria, S.K. Lalwani[1], V. Mishra, A. Topkar

Tata Institute of Fundamental Research - EHEP, Mumbai, INDIA
T. Aziz, S. Banerjee, S. Chendvankar, P.V. Deshpande, A. Gurtu, G. Majumder, K. Mazumdar[1], M.R. Patil, K. Sudhakar, S.C. Tonwar

Tata Institute of Fundamental Research - HECR, Mumbai, INDIA
B.S. Acharya, S. Banerjee, S. Bheesette, S. Dugad, S.D. Kalmani, V.R. Lakkireddi, N.K. Mondal, N. Panyam, P. Verma

Institute for Studies in Theoretical Physics & Mathematics (IPM), Tehran, IRAN
F. Ardalan, H. Arfaei, R. Mansouri, M. Mohammadi

Università di Bari, Politecnico di Bari e Sezione dell' INFN, Bari, ITALY
M. Abbrescia, E. Carrone[1], E. Cavallo, A. Colaleo, D. Creanza, N. De Filippis, M. De Palma, L. Fiore, D. Giordano, G. Iaselli, F. Loddo, G. Maggi, M. Maggi, B. Marangelli, S. My, S. Natali, S. Nuzzo, G. Pugliese, V. Radicci, A. Ranieri, F. Romano, F. Ruggieri, G. Selvaggi, L. Silvestris, P. Tempesta, R. Trentadue, P. Vankov, G. Zito

Università di Bologna e Sezione dell' INFN, Bologna, ITALY
S. Arcelli, A. Benvenuti, D. Bonacorsi, P. Capiluppi, F. Cavallo, M. Cuffiani, I. D'Antone, G.M. Dallavalle, F. Fabbri, A. Fanfani, P. Giacomelli[5], C. Grandi, M. Guerzoni, S. Marcellini, P. Mazzanti, A. Montanari,

C. Montanari, F. Navarria, F. Odorici, A. Perrotta, A. Rossi, T. Rovelli, G. Siroli, R. Travaglini

Università di Catania e Sezione dell' INFN, Catania, ITALY
S. Albergo, V. Bellini, S. Cavalieri, M. Chiorboli, S. Costa, R. Potenza, C. Randieri, C. Sutera, A. Tricomi, C. Tuve

Università di Firenze e Sezione dell' INFN, Firenze, ITALY
A. Bocci, E. Borchi, V. Ciulli, C. Civinini, R. D'Alessandro, E. Focardi, A. Macchiolo, N. Magini, C. Marchettini, M. Meschini, S. Paoletti, G. Parrini, R. Ranieri, M. Sani

Università di Genova e Sezione dell' INFN, Genova, ITALY
P. Fabbricatore, M. Fossa, R. Musenich, C. Pisoni

Laboratori Nazionali di Legnaro dell' INFN, Legnaro, ITALY (associated institute)
L. Berti, M. Biasotto, E. Ferro, U. Gastaldi, M. Gulmini[1], G. Maron, N. Toniolo, L. Zangrando

Istituto Nazionale di Fisica Nucleare e Universita Degli Studi Milano-Bicocca, Milano, ITALY
M. Bonesini, G. Franzoni, A. Ghezzi, P. Govoni, A. Meregaglia, P. Negri, M. Paganoni, A. Pullia, S. Ragazzi, N. Redaelli, T. Tabarelli de Fatis

Istituto Nazionale di Fisica Nucleare de Napoli (INFN), Napoli, ITALY
L. Lista, P. Paolucci, D. Piccolo

Università di Padova e Sezione dell' INFN, Padova, ITALY
M. Bellato, M. Benettoni, D. Bisello, A. Candelori, P. Checchia, E. Conti, M. Corvo, M. De Giorgi, A. De Min, U. Dosselli, C. Fanin, F. Fanzago, F. Gasparini, U. Gasparini, F. Gonella, A. Kaminski, S. Lacaprara, I. Lippi, M. Loreti, O. Lytovchenko, M. Mazzucato, A.T. Meneguzzo, M. Michelotto, F. Montecassiano[1], M. Nigro, M. Passaseo, M. Pegoraro, P. Ronchese, E. Torassa, S. Vanini, L. Ventura, S. Ventura, M. Zanetti, P. Zotto, G. Zumerle

Università di Pavia e Sezione dell' INFN, Pavia, ITALY
G. Belli, R. Guida, S.P. Ratti, P. Torre, P. Vitulo

Università di Perugia e Sezione dell' INFN, Perugia, ITALY
M.M. Angarano, E. Babucci, M. Biasini, G.M. Bilei[1], M.T. Brunetti, B. Checcucci, N. Dinu, L. Fanò, M. Giorgi, P. Lariccia, G. Mantovani, D. Passeri, P. Placidi, V. Postolache, D. Ricci, M. Risoldi, R. Santinelli, A. Santocchia, L. Servoli

Università di Pisa, Scuola Normale Superiore e Sezione dell' INFN, Pisa, ITALY
G. Bagliesi, A. Bardi, A. Basti, J. Bernardini, T. Boccali, L. Borrello, F. Bosi, P.L. Braccini, R. Castaldi, R. Cecchi, R. Dell'Orso, S. Donati, F. Donno, S. Dutta, L. Foà, S. Galeotti, S. Gennai, A. Giassi, S. Giusti, G. Iannaccone, A. Kyriakis, L. Latronico, F. Ligabue, M. Massa, A. Messineo, A. Moggi, F. Morsani, F. Palla, F. Palmonari, F. Raffaelli, G. Sanguinetti, G. Segneri, P. Spagnolo, M. Spezziga, F. Spinella, A. Starodumov[6], G. Tonelli, C. Vannini, P.G. Verdini, J.C. Wang

Università di Roma I e Sezione dell' INFN, Roma, ITALY
S. Baccaro[7], L. Barone, A. Bartoloni, F. Cavallari, S. Costantini, I. Dafinei, M. Diemoz, C. Gargiulo, E. Longo, P. Meridiani, M. Montecchi[7], G. Organtini, R. Paramatti

Università di Torino e Sezione dell' INFN, Torino, ITALY
N. Amapane, M. Arneodo, F. Bertolino, C. Biino[1], R. Cirio, M. Costa, D. Dattola[1], L. Demaria, G. Favro, S. Maselli, E. Migliore, V. Monaco, C. Peroni, A. Romero, M. Ruspa, R. Sacchi, A. Solano, A. Staiano, P.P. Trapani

Chungbuk National University, Chongju, KOREA
Y.U. Kim

Kangwon National University, Chunchon, KOREA
S.K. Nam

Wonkwang University, Iksan, KOREA
S.Y. Bahk

Cheju National University, Jeju, KOREA
Y.J. Kim

Chonnam National University, Kwangju, KOREA
J.Y. Kim, I.T. Lim

Dongshin University, Naju, KOREA
M.Y. Pac

Seonam University, Namwon, KOREA
S.J. Lee

Konkuk University, Seoul, KOREA
S.Y. Jung, J.T. Rhee

Korea University, Seoul, KOREA
B.S. Hong, S.J. Hong, K.S. Lee, S.K. Park, K.S. Sim

Seoul National University of Education, Seoul, KOREA
D.G. Koo

Seoul National University, Seoul, KOREA
K.K. Joo, B.J. Kim, S.B. Kim, I.H. Park

Sungkyunkwan University, Suwon, KOREA
B.G. Cheon, Y.I. Choi

Kyungpook National University, Taegu, KOREA
K. Cho, S.W. Ham, D.H. Kim, G.N. Kim, W.Y. Kim, J.Y. Lee, S.K. Oh, Y.D. Oh, S.R. Ro, D.C. Son

National Centre for Physics, Quaid-I-Azam University, Islamabad, PAKISTAN
Z. Aftab, M. Ahmad, I. Ahmed, M.I. Asghar, H.R. Hoorani, S.M. Khan, A. Osman[1], N. Qaiser, R. Riazuddin, T. Solaija

National University of Sciences And Technology, Rawalpindi Cantt, PAKISTAN (associated institute)
A. Ali, A. Bashir, A. Muhammad, M. Saeed

Institute of Experimental Physics, Warsaw, POLAND
M. Bluj, K. Bunkowski, M. Cwiok, H. Czyrkowski, R. Dabrowski, W. Dominik, K. Doroba, M. Kazana, J. Krolikowski, I. Kudla, M. Pietrusinski, K. Pozniak[22], W. Zabolotny[22], J. Zalipska, P. Zych

Soltan Institute for Nuclear Studies, Warsaw, POLAND
M. Górski, L. Goscilo, K. Nawrocki, G. Wrochna, P. Zalewski

Laboratório de Instrumentação e Física Experimental de Partículas, Lisboa, PORTUGAL
R. Alemany-Fernandez, C. Almeida, N. Almeida, J. Augusto, P. Bordalo, S. Da Mota Silva, J. Da Silva, O.P. Dias, J. Gomes, F.M. Goncalves, N. Leonardo, S. Ramos, R. Ribeiro, M. Santos, J. Semiao, I. Teixeira, J.P. Teixeira, J. Varela, N. Vaz Cardoso

Joint Institute for Nuclear Research, Dubna, RUSSIA
I. Anissimov, A. Belkov, A. Cheremukhin, A. Dmitriev, V. Elsha, Y. Erchov, I. Golutvin, N. Gorbunov, I. Gramenitsky, A. Janata, V. Kalagin, V. Karjavin, A. Karlov, S. Khabarov, V. Khabarov, Y. Kiryushin, V. Kolesnikov, V. Konoplyanikov, V. Korenkov, I. Kossarev, V. Ladyguine, V. Lysiakov, A. Malakhov, I. Melnichenko, G. Meshcheryakov, V.V. Mitsyn, P. Moisenz, S. Movchan, V. Palichik, V. Perelygin, A. Rogov, S. Sergeev, A. Sergueev, S. Shmatov, S. Shulha, I. Slepnev, V. Smirnov, D. Smolin, E. Tikhonenko, A. Urkinbaev[8], E. Vedenyapina, A. Vichnevski, A. Volodko, N. Zamiatin, A. Zarubin, P. Zarubin, E. Zubarev

Petersburg Nuclear Physics Institute, Gatchina (St Petersburg), RUSSIA
A. Atamantchouk, A. Baldychev, V. Barashko, N. Bondar, V. Golovtsov, D. Goulevich, Y. Ivanov, V. Kim, E. Kouznetsova, V. Kozlov, E. Lobatchev, G. Makarenkov, E. Orishchin, V. Rasmislovich, B. Razmyslovich, V. Sknar, I. Smirnov, S. Sobolev, I. Tkach, L. Uvarov, G. Velitchko, S. Volkov, A. Vorobyev, D. Yakorev

Institute for Nuclear Research, Moscow, RUSSIA
I. Andreev, P. Antipov, G.S. Atoyan, S. Gninenko, N. Goloubev, E.V. Gushin, M. Kirsanov, A. Kovzelev,

N. Krasnikov, V. Matveev, A. Pashenkov, A. Poliarouch, V.E. Postoev, V. Shmatkov, A. Skassyrskaia[9], A. Solovey, L. Stepanova, A. Toropin[9]

Institute for Theoretical and Experimental Physics, Moscow, RUSSIA
V. Gavrilov, N. Ilina, V. Kaftanov, I. Kisselevitch, V. Kolosov, M. Kossov, A. Krokhotine, S. Kuleshov, N. Stepanov, V. Stoline, S. Uzunyan, V. Zakharov

P.N. Lebedev Physical Institute, Moscow, RUSSIA
A.M. Fomenko, N. Konovalova, V. Kozlov, A.I. Lebedev, N. Lvova, S. Potashov, S.V. Rusakov

Moscow State University, Moscow, RUSSIA
E. Boos, A. Cherstnev, L. Dudko, A. Erchov, R. Gloukhov, A. Gribushin, V. Ilyin, O.L. Kodolova[1], A. Kryukov, I.P. Lokhtin, V. Mikhaylin, L. Sarycheva, V. Savrin, L. Shamardin, A. Snigirev, I. Vardanyan

High Temperature Technology Center of Research & Development Institute of Power Engineering (HTTC RDIPE), Moscow, RUSSIA (associated institute)
D. Chmelev, D. Druzhkin, A. Ivanov, V. Koudinov, O. Logatchev, S. Onishchenko, A. Orlov, V. Sakharov, V. Smetannikov, A. Tikhomirov, S. Zavodthikov

State Research Center of Russian Federation - Institute for High Energy Physics, Protvino, RUSSIA
A. Abramov, V. Abramov, A. Annenkov[10], I. Azhgirey, S. Belyanchenko, S. Bitioukov, P. Goncharov, V. Goussev, V. Grichine, A. Inyakin, V. Katchanov, A. Khmelnikov, E. Kolatcheva, A. Korablev, Y. Korneev, A. Kostritski, A. Krinitsyn, V. Kryshkin, E. Kvachina, O. Lapyguina, A. Levine, A. Markov, V. Medvedev, M. Oukhanov, V. Pak, V. Petrov, V. Pikalov, P. Podlesnyy, V. Potapov, A. Riabov, A. Sannikov, Z. Simonova, E. Skvortsova, S. Slabospitski, A. Sobol, A. Soldatov, S. Stepouchkine, A. Surkov[1], A. Sytin, V. Talanov, B. Tchuiko, S. Tereschenko, S. Troshin, L. Turchanovich, N. Tyurin, A. Uzunian, A. Volkov, A. Zaitchenko, S. Zelepoukine

Russian Federal Nuclear Centre - Scientific Research Institute for Technical Physics (RFNC-VNIITF), Snezhinsk, RUSSIA (associated institute)
A. Andriyash, D. Batin, D. Gorchkov, D. Griaznykh, O. Gueinak, D. Korotchine, S. Koshelev, S. Kotegov, Y. Kretinin, A. Maloiaroslavtsev, M. Naoumenko, I. Pavlov, V. Pravilnikov, S. Samarine, D. Shadrin, R. Skripov

Electron National Research Institute, St Petersburg, RUSSIA (associated institute)
V. Lukyanov, G. Mamaeva, Z. Prilutskaya, I. Rumyantsev, E. Sidorovitch, S. Sokha, S. Tataurschikov, I. Vasilyev

Myasishchev Design Bureau, Zhukovsky, RUSSIA (associated institute)
V. Shirinyants

Vinca Institute of Nuclear Sciences, Belgrade, SERBIA
P. Adzic, I. Anicin, S. Drndarevic, J. Ilic, D. Krpic, P. Milenovic, J. Puzovic, G. Skoro, N. Smiljkovic, M. Zupan

Slovak University of Technology, Bratislava, SLOVAK REPUBLIC
K. Vitazek

Centro de Investigaciones Energeticas Medioambientales y Tecnologicas, Madrid, SPAIN
M. Aguilar-Benitez, J. Alberdi, P. Arce[1], C. Burgos, M. Cerrada, N. Colino, M. Daniel, B. De La Cruz, C. Fernandez Bedoya, A. Ferrando, M.C. Fouz, P. Garcia-Abia, M.I. Josa, J.M. Luque, J. Marin, A. Molinero, J.C. Oller, J. Puerta Pelayo, L. Romero, J. Salicio, C. Willmott

Universidad Autónoma de Madrid, Madrid, SPAIN
C. Albajar, R. Macias[1], J.F. Trocóniz

Universidad de Oviedo, Oviedo, SPAIN
F. Arteche[1], J. Cuevas, J.M. Lopez

Instituto de Física de Cantabria (IFCA), CSIC-Universidad de Cantabria, Santander, SPAIN
A. Calderon, E. Calvo, C. Figueroa, M.A. Lopez Virto, J. Marco, R. Marco, C. Martinez Rivero, F. Matorras,

T. Rodrigo, D. Rodriguez Gonzalez, A. Ruiz Jimeno, I. Vila

Universität Basel, Basel, SWITZERLAND
S. Cucciarelli, M. Konecki, L. Tauscher, S. Vlachos

CERN, European Organization for Nuclear Research, Geneva, SWITZERLAND
D. Abbaneo, M. Algar, S. Ashby, P. Aspell, E. Auffray, P. Baillon, A. Ball, N. Bangert, D. Barney, P. Bartalini, M. Battaglia, P. Bloch, M. Bosteels, A. Branson, H. Breuker, V. Brigljevic, G. Bruno, D. Campi, T. Camporesi, E. Cano, A. Cattai, G. Cervelli, R. Chierici, J. Christiansen, S. Cittolin, J. Cogan, J. Correia Fernandes[11], A. Csilling, B. Curé, A. De Roeck, T. de Visser, D. Delikaris, M. Della Negra, A. Elliott-Peisert, H. Foeth, R. Folch, S. Fratianni, A. Frey, A. Frohner, W. Funk, D. Futyan, A. Gaddi, J.C. Gayde, H. Gerwig, K. Gill, F. Glege, W. Glessing, J.P. Grillet, J. Gutleber, R. Hammarstrom, M. Hansen, A. Hervé, A. Honma, M. Huhtinen, V. Innocente, W. Jank, K. Kloukinas, Z. Kovacs, V. Lara, H. Larsen, C. Lasseur, E. Laure, M. Lebeau, P. Lecoq, M. Lenzi, M. Letheren, M. Liendl, L. Linssen, C. Ljuslin, B. Lofstedt, R. Loos, G. Magazzu, I. Magrans, M. Mannelli, J.M. Maugain, F. Meijers, E. Meschi, J. Mocholi Moncholi, F. Mossiere, A. Moutoussi, S.J. Murray, G. Nuessle, A. Oh, A. Onnela, M. Oriunno, L. Orsini, C. Palomares Espiga, L. Pape, G. Passardi, G. Perinic, P. Petagna, A. Petrilli, A. Pfeiffer, M. Pimiä, R. Pintus, B. Pirollet, V. Poireau, J.P. Porte, H. Postema, J. Pothier, R. Principe, A. Racz, P. Rebecchi, S. Reynaud, J. Roche, P. Rodrigues Simoes Moreira, G. Rolandi, D. Samyn, J.C. Santiard, P. Scharff-Hansen, R. Schmidt, C. Schwick, P. Sempere Roldán, S. Sequeira Tavares, G. Sguazzoni, P. Sharp[12], P. Siegrist, N. Sinanis, P. Sphicas[13], M. Stavrianakou, H. Stockinger, A. Strandlie, F. Szoncsó, B.G. Taylor, O. Teller, J. Troska, E. Tsesmelis, A. Tsirou, J. Valls, I. Van Vulpen, F. Vasey, L. Veillet, J.P. Wellisch, P. Wertelaers, I. Wichrowska-Polok, M. Wilhelmsson, I.M. Willers, M. Winkler

Paul Scherrer Institut, Villigen, SWITZERLAND
W. Bertl, K. Deiters, K. Gabathuler, S. Heising, R. Horisberger, Q. Ingram, H.C. Kaestli, D. Kotlinski, A. Macpherson[1], D. Renker, T. Rohe, T. Sakhelashvili

Institut für Teilchenphysik, Eidgenössische Technische Hochschule (ETH), Zürich, SWITZERLAND
B. Betev, B. Blau, P. Cannarsa[1], G. Dissertori, M. Dittmar, L. Djambazov, J. Ehlers, R. Eichler, W. Erdmann, G. Faber, K. Freudenreich, R. Goudard[1], C. Grab, A. Holzner, C. Humbertclaude, P. Ingenito, P. Lecomte, A. Lister, W. Lustermann, F. Nessi-Tedaldi, A.S. Nicollerat, R.A. Ofierzynski, A. Patino Revuelta[1], F. Pauss, P. Riboni, U. Roeser, H. Rykaczewski[1], F. Sanchez Galan[1], C. Schaefer[1], S. Stoenchev, H. Suter, S. Udriot, G. Viertel, H. Von Gunten, S. Waldmeier-Wicki

Universität Zürich, Zürich, SWITZERLAND
C. Amsler, A. Dorokhov, C. Hoermann, K. Prokofiev, H. Pruys, C. Regenfus, P. Robmann, T. Speer, S. Steiner

Cukurova University, Adana, TURKEY
I. Dumanoglu, E. Eskut, A. Kayis, A. Kuzucu-Polatöz, G. Önengüt

Middle East Technical University, Physics Department, Ankara, TURKEY
K. Cankocak[14], A. Esendemir, O. Polat, M. Serin-Zeyrek, R. Sever, H. Yildiz, M. Zeyrek

Bogaziçi University, Department of Physics, Istanbul, TURKEY
E. Gulmez, E. Isiksal[15], M. Kaya[16], S. Ozkorucuklu[17], R. Unalan

Institute of Single Crystals of National Academy of Science, Kharkov, UKRAINE
B. Grinev, V. Senchyshyn, V. Vasilchuk

National Scientific Center, Kharkov Institute of Physics and Technology, Kharkov, UKRAINE
L. Levchuk, V. Popov, P. Sorokin

Kharkov State University, Kharkov, UKRAINE
V. Kovtun

University of Bristol, Bristol, UNITED KINGDOM
D.S. Bailey, T. Barrass, J.J. Brooke, D. Cussans, G.P. Heath, H.F. Heath, C.K. Mackay, O. Maroney, D.M. Newbold, M.G. Probert, V.J. Smith, R.J. Tapper

Centre for Complex Cooperative Systems, University of the West of England, Bristol, UNITED KINGDOM (associated institute)
N. Baker, A. Barry, P. Brooks, G. Chevenier[1], F. Estrella[1], S. Gaspard, G. Mathers[1], R. McClatchey[1], A. Solomonides, N. Toth[1], F. Van Lingen[1]

Rutherford Appleton Laboratory, Didcot, UNITED KINGDOM
S.A. Baird, K.W. Bell, R.M. Brown, D.J.A. Cockerill, J.A. Coughlan, P.S. Flower, V.B. Francis, M. French, B. Gannon, J. Greenhalgh, R. Halsall, W.J. Haynes, J. Hill, L. Jones, B.W. Kennedy, L. Lintern, A.B. Lodge, J. Maddox, Q. Morrissey, P. Murray, S. Quinton, J. Salisbury, A. Shah, B. Smith, M. Sproston, R. Stephenson, I.R. Tomalin, M. Torbet, J.H. Williams

University of Strathclyde, Glasgow, UNITED KINGDOM (associated institute)
M. Roper

Imperial College, University of London, London, UNITED KINGDOM
R. Bainbridge, G. Barber, D. Britton, W. Cameron, E. Corrin, G. Dewhirst, C. Foudas, G. Hall, P. Lewis, B.C. MacEvoy, N. Marinelli, E.M. McLeod, A. Nikitenko[6], E. Noah Messomo, M. Noy, X. Qu, D.M. Raymond, S. Rutherford, P.J. Savage, C. Seez[1], H. Tallini, T. Virdee[1], O. Zorba

Brunel University, Uxbridge, UNITED KINGDOM
B. Camanzi[1], C. Da Via, P.R. Hobson, P. Kyberd, I. Reid, O. Sharif, S.J. Watts

University of California, Davis, Davis, California, USA
R. Breedon, M. Case, M. Chertok, P.T. Cox, B. Holbrook, W. Ko, R. Lander, P. Murray, D. Pellett, J. Smith, M. Tripathi

University of California, San Diego, La Jolla, California, USA
S. Bhattacharya, J.G. Branson, I. Fisk, J. Letts, M. Mojaver, H.P. Paar, A. White

University of California, Los Angeles, Los Angeles, California, USA
V. Andreev[1], K. Arisaka, D. Cline, R. Cousins, S. Erhan, J. Hauser, P. Kreuzer, M. Lindgren, B. Lisowski, B. Liu, C. Matthey, J. Mumford, S. Otwinowski, Y. Pischalnikov, P. Schlein, Y. Shi, B. Tannenbaum, V. Valuev, M. Von Der Mey, H.G. Wang, X. Yang

California Institute of Technology, Pasadena, California, USA
E. Aslakson, A. Bornheim, J. Bunn, G. Denis, P. Galvez, M. Gataullin, M. Hafeez, K. Holtman, S. Iqbal, T. Lee, I. Legrand, V. Litvine, H.B. Newman, S.P. Pappas, S. Ravot, S. Shevchenko, S. Singh, E. Soedarmadji, C. Steenberg, K. Wei, Q. Wei, A. Weinstein, R. Wilkinson, L. Xia, L.Y. Zhang, K. Zhu, R.Y. Zhu

University of California, Riverside, Riverside, California, USA
I. Andreeva[1], G. Benelli, R. Clare, I. Crotty[1], J.W. Gary, M. Giunta, G. Hanson, J.G. Layter, G. Pasztor[18], B.C. Shen, V. Sytnik, P. Tran, S. Villa, D. Zer-Zion

University of California, Santa Barbara, Santa Barbara, California, USA
C. Campagnari, D. Hale, J. Incandela, D. Stuart, R. Taylor, D. White

Fairfield University, Fairfield, Connecticut, USA
C.P. Beetz, G. Cirino, V. Podrasky, C. Sanzeni, D. Winn

University of Florida, Gainesville, Florida, USA
D. Acosta, P. Avery, D. Bourilkov[19], R. Cavanaugh, S. Dolinsky, A. Drozdetski, R.D. Field, S. Klimenko, J. Konigsberg, A. Korytov, P. Levchenko, A. Madorsky, K. Matchev, G. Mitselmakher, H. Pi, P. Ramond, J.L. Rodriguez, B. Scurlock, H. Stoeck, S.M. Wang, J. Yelton

Florida Institute of Technology, Melbourne, Florida, USA
M.M. Baarmand, L. Baksay[20], M. Hohlmann, R.S. Jin, I. Vodopianov

Florida State University, Tallahassee, Florida, USA
M. Bertoldi, S. Hagopian, V. Hagopian, K.F. Johnson, H. Prosper, J. Thomaston, H. Wahl

Fermi National Accelerator Laboratory, Batavia, Illinois, USA
J. Amundson, G. Apollinari, M. Atac, S. Aziz, B. Banerjee, L.A.T. Bauerdick, A. Baumbaugh, U. Baur,

M. Bowden, N. Chester, I. Churin, S. Cihangir, D.P. Eartly, J.E. Elias, J. Freeman, I. Gaines, J. Goldstein, G. Graham, D. Green, J. Hanlon, S. Hansen, A.H. Heering, S.L. Holm, U. Joshi, T. Kramer, S. Kwan, G. Lanfranco, M. Larwill, D. Lazic, M. Litmaath, S. Los, K. Maeshima, J.M. Marraffino, N. Mokhov, C. Moore, V. O'Dell, D. Petravick, R. Pordes, O. Prokofiev, N. Ratnikova, M. Reichanadter, C.H. Rivetta, A. Ronzhin, T. Shaw, E. Skup, R.P. Smith, L. Spiegel, I. Suzuki, W. Tanenbaum, S. Tkaczyk, R. Vidal, R. Wands, H. Wenzel, J. Whitmore, W.M. Wu, Y. Wu, V. Yarba

University of Illinois at Chicago (UIC), Chicago, Illinois, USA
M.R. Adams, E. Chabalina, C.E. Gerber, W. Qian

Northwestern University, Evanston, Illinois, USA
B. Gobbi, M. Kubantsev, S. Malik, R. Tilden

University of Notre Dame, Notre Dame, Indiana, USA
B. Baumbaugh, N.M. Cason, M. Hildreth, D.J. Karmgard, A. Kharchilava, R. Ruchti, J. Warchol, M. Wayne

Purdue University, West Lafayette, Indiana, USA
K. Arndt, V.E. Barnes, G. Bolla, D. Bortoletto, A. Bujak, A.F. Garfinkel, L. Gutay, Y. Kozhevnikov, A.T. Laasanen, D. Miller, J. Miyamoto, A. Pompos, C. Rott, A. Roy, A. Sedov, I. Shipsey

Iowa State University, Ames, Iowa, USA
E.W. Anderson, O. Atramentov, J.M. Hauptman

The University of Iowa, Iowa City, Iowa, USA
U. Akgun, A.S. Ayan, A. Cooper, F. Duru, M. Fountain, E. McCliment, J.P. Merlo, A. Mestvirishvili, M.J. Miller, E. Norbeck, Y. Onel, I. Schmidt

The University of Kansas, Lawrence, Kansas, USA
P. Baringer, A. Bean, L. Christofek

Kansas State University, Manhattan, Kansas, USA
T. Bolton, R. Demina, W.E. Kahl, A. Khanov[6], N. Poukhaeva, N. Reay, F. Rizatdinova, R. Sidwell, N. Stanton, E. Von Toerne

Johns Hopkins University, Baltimore, Maryland, USA
B.A. Barnett, C.Y. Chien, D. Kim, G. Liang, S. Spangler, M. Swartz

University of Maryland, College Park, Maryland, USA
S. Abdullin[6], D. Baden, R. Bard, S.C. Eno, D. Fong, T. Grassi, N.J. Hadley, C. Jarvis, R.G. Kellogg, S. Kunori, A. Skuja

Boston University, Boston, Massachusetts, USA
G. Antchev[19], E. Hazen, E. Machado, D. Osborne, J. Rohlf, L. Sulak, S. Wu

Northeastern University, Boston, Massachusetts, USA
G. Alverson, A. Kuznetsov, J. Macleod, J. Moromisato, Y.V. Musienko[21], S. Muzaffar, I. Osborne, S. Reucroft, J. Swain, L. Taylor, L. Tuura

Massachusetts Institute of Technology, Cambridge, Massachusetts, USA
M. Ballintijn, G. Bauer, J. Friedman, C. Paus, S. Pavlon, C. Roland, K. Sumorok, S. Tether, F. Wuerthwein, B. Wyslouch

University of Minnesota, Minneapolis, Minnesota, USA
D. Bailleux, P. Cushman, A. Dolgopolov, R. Egeland, W.J. Gilbert, J. Grahl, M.M. Obertino, N. Pearson, R. Rusack, A. Singovsky, Y. Vetter

University of Mississippi, University, Mississippi, USA
L.M. Cremaldi, J. Reidy, D. Sanders, D. Summers

University of Nebraska-Lincoln, Lincoln, Nebraska, USA
W.B. Campbell, D.R. Claes, C. Lundstedt, G.R. Snow

Rutgers, the State University of New Jersey, Piscataway, New Jersey, USA
E. Bartz, J. Conway, T. Devlin, J. Doroshenko, K. Horvath, P.F. Jacques, M.S. Kalelkar, T. Koeth, A. lath,

L. Perera, R. Plano, S. Schnetzer, S. Somalwar, R. Stone, G. Thomson, T.L. Watts, S. Worm

Princeton University, Princeton, New Jersey, USA
P. Elmer, W.C. Fisher, V. Gupta, J. Mans, D. Marlow, P. Piroué, D. Stickland, C. Tully, T. Wildish, S. Wynhoff, Z. Xie

University of Rochester, Rochester, New York, USA
A. Bodek, H. Budd, Y.S. Chung, P. de Barbaro, R. Eusebi, G. Ginther, E. Halkiadakis, A. Hocker, M. Imboden, D. Ruggiero, W. Sakumoto, P. Slattery, P. Tipton

The Ohio State University, Columbus, Ohio, USA
B. Bylsma, L.S. Durkin, J. Gilmore, J. Gu, D. Herman, D. Larsen, T.Y. Ling, C.J. Rush, V. Sehgal

Carnegie Mellon University, Pittsburgh, Pennsylvania, USA
T. Ferguson, J. Russ, N. Terentyev, H. Vogel, I. Vorobiev

Rice University, Houston, Texas, USA
G. Eppley, M. Matveev, T. Nussbaum, B.P. Padley, J. Roberts, P. Yepes

Texas Tech University, Lubbock, Texas, USA
N. Akchurin, J. Cranshaw, O. Lobban, V. Papadimitriou, A. Sill, R. Thomas, E. Washington, R. Wigmans, L. Zhang

University of Wisconsin, Madison, Wisconsin, USA
D. Bradley, D. Carlsmith, P. Chumney, S. Dasu, F.R. Di Lodovico, F. Feyzi, L. Greenler, M. Jaworski, J. Lackey, R. Loveless, S. Lusin, W. Mason, D. Reeder, W.H. Smith, D. Wenman

Institute of Nuclear Physics of the Uzbekistan Academy of Sciences, Ulugbek, Tashkent, UZBEKISTAN
M. Belov, Y. Koblik, B.S. Yuldashev

1: Also at CERN, European Organization for Nuclear Research, Geneva, Switzerland
2: Also at Université Louis Pasteur, Strasbourg, France
3: Also at Université de Haute-Alsace, Mulhouse, France
4: Also at Moscow State University, Moscow, Russia
5: Also at University of California, Riverside, Riverside, USA
6: Also at Institute for Theoretical and Experimental Physics, Moscow, Russia
7: Also at ENEA - Casaccia Research Center, S. Maria di Galeria, Italy
8: Also at Institute of Nuclear Physics of the Uzbekistan Academy of Sciences, Ulugbek, Tashkent, Uzbekistan
9: Also at Università di Pisa, Scuola Normale Superiore e Sezione dell' INFN, Pisa, Italy
10: Also at Bogoroditsk Technical Plant, Moscow, Russia
11: Also at Porto University, Portugal
12: Also at Rutherford Appleton Laboratory, Didcot, United Kingdom
13: Also at MIT, Cambridge, USA and University of Athens, Greece
14: Also at Mugla University, Turkey
15: Also at Marmara University, Istanbul, Turkey
16: Also at Kafkas University, Kars, Turkey
17: Also at Suleyman Demirel University, Isparta, Turkey
18: Also at KFKI Research Institute for Particle and Nuclear Physics, Budapest, Hungary
19: Also at Institute for Nuclear Research and Nuclear Energy, Sofia, Bulgaria
20: Also at Kossuth Lajos University, Debrecen, Hungary
21: Also at Institute for Nuclear Research, Moscow, Russia
22: Also at Institute of Electronic Systems, Technical University of Warsaw, Poland

Table Of Contents

List of Figures

List of Tables

List of Glossary Terms

ACK	Acknowledgement
API	Application Programming Interface
aTTS	asynchronous Trigger-Throttling System
BCN	Builder Control Network
BDN	Builder Data Network
BM	Builder Manager
BS	Barrel Shifter
BU	Builder Unit
BX	Bunch Crossing
CIOQ	Combined Input Output Queuing
CMS	Compact Muon Solenoid
CPU	Central Processing Unit
CRC	Cyclic Redundancy Code
CSC	Cathode Strip Chambers
CVS	Concurrent Versioning System
DAQ	Data Aquisition
DCC	Data Concentrator Card
DCS	Detector Control System
DMA	Direct Memory Access
DOM	Document Object Model
DT	Drift Tube Muon Detector
EB	Electromagnetic Calorimeter (Barrel)
ECAL	Electromagnetic Calorimeter
EE	Electromagnetic Calorimeter (Endcap)
EM	Electromagnetic Calorimeter
EVB	Event Builder
Event-ID	Event Identifier
EVM	Event Manager
FBI	Front-end Builder Input
FC	Filter Column
FCN	Filter Control Network
FDN	Filter Data Network
FEC	Front-End Controler
FED	Front-End Driver
FEDB	Front-End Driver Builder
FES	Front-End System
FIFO	First In First Out
FM	Function Manager
FPGA	Field Programmable Gate Array
FRL	Front-End Readout Link
FU	Filter Unit
GIII	Generic PCI Platform III
GbE	Gigabit Ethernet
GNU	GNU’s not Unix

GTL	Global Trigger Logic
HB	Hadron Calorimeter (Barrel)
HCAL	Hadron Calorimeter (
HE	Hadron Calorimeter (Endcap)
HF	Hadron Calorimeter (Forward region)
HLT	High-Level Trigger
HO	Hadron Calorimeter (Outer Barrel)
HPC	High Performance Computing
HTTP	Hyper Text Transfer Protocol
I/O	Input / Output
I2O	Intelligent Input/Output
IMS	Information and Monitoring Service
IP	Internet Protocol
IQ	Input Queuing
JAS	Java Analysis Studio
JC	Job Control
LAN	Local Area Network
LHC	Large Hardron Collider
LHCrx	LHC orbit and clock recovery circuit
LVDS	Low Voltagge Differential Signal
MAC	Media Access Control
MCP	Myrinet Control Program (firmware for Myrinet NIC)
MPI	Message Passing Interface
MTU	Maximum Transfer Unit
MySQL	A free SQL database implementation
NACK	Negative Acknowledgement
NIC	Network Interface Card
OS	Operating System
OQ	Output Queue
PC	Personal Computer
PCI	Peripheral Component Interconnect
PS	Problem Solver
RAM	Random Access Memory
RCMS	Run Control and Monitoring System
RCN	Readout Control Network
RCNP	Readout Control Network Protocol
RD	Research and Development
ReSync	Re-Synchronization Command issued by TCS
RISC	Reduced Instruction Set Computer
RPC	Resistive Plate Chamber
RM	Readout Manager
rms	root mean square
RS	Resource Service
RU	Readout Unit
RUI	Readout Unit Input
RUO	Readout Unit Output
SCADA	Supervisory Control and Data Acquisition

SDRAM	Synchronous Dynamic Random Access Memory
SE	Preshower Endcap
s-fragment	super-fragment (assembled by the FED Builder)
SM	Subfarm Manager
SMR	Session Manager
SOAP	Simple Object Access Protocol (an XML Protocol)
SS	Security Service
SSC	Sub-System Controler
sTTS	synchronous Trigger-Throttling System
TCP	Transmission Control Protocol
TCS	Trigger Control System
TDR	Technical Design Report
TTC	Trigger Timing and Control
TTCcf	TTC Fanout Module
TTCmi	TTC Machine Interface
TTL	Transistor Transistor Logic
TTS	Trigger-Throttling System
UDP	User Datagram Protocol
UTP	Unshielded Twister Pair
VLAN	Virtual Loacal Area Network
VME	Versa Module Eurocard (IEEE 1014)
W3C	World Wide Web Consortium
Xbar	Cross bar
XDAQ	Cross DAQ (online software framework)
XML	Extensible Markup Language

Part 1
DAQ Architecture and Design

1 Overview

The Trigger and Data Acquisition (DAQ) system of an experiment at a hadron collider plays an essential role because both the collision and the overall data rates are much higher than the rate at which one can write data to mass storage. At the LHC, the proton beams will cross each other at a frequency of 40 MHz. At the design luminosity of $10^{34}cm^{-2}s^{-1}$ each crossing results in an average of ~20 inelastic pp events producing approximately 1 MB of zero-suppressed data. These figures are many orders of magnitude larger than the archival storage capability of $O(10^2)$ Hz at data rates of $O(10^2)$ MB/s.

The CMS Trigger and Data Acquisition System (TriDAS) is designed to inspect the detector information at the full crossing frequency and to select events at a maximum rate of $O(10^2)$ Hz for archiving and later offline analysis. The required rejection power of $O(10^5)$ is too large to be achieved in a single processing step, if a high efficiency is to be maintained for the physics phenomena CMS plans to study. For this reason, the full selection task is split into two steps. The first step (Level-1 Trigger) is designed to reduce the rate of events accepted for further processing to less than 100 kHz. The second step (High-Level Trigger or “HLT”) is designed to reduce this maximum Level-1 accept rate of 100 kHz to a final output rate of 100 Hz. The design of the Level-1 Trigger has already been extensively documented in Volume I of the TriDAS Technical Design Report (TDR), submitted to the LHCC in December 2000 [1-1]. The present TDR documents the systems necessary to acquire the detector data, inspect it, provide the final selection stage, and archive the selected events. Even though the system is optimized for the running conditions relevant to pp collisions, its performance is also adequate for Heavy Ion collisions.

The functionality of the CMS DAQ/HLT system is three-fold:

- perform the readout of the front-end electronics after a Level-1 Trigger accept;
- execute physics selection algorithms on the events read out, in order to accept the ones with the most interesting physics content;
- forward these accepted events, as well as a small sample of the rejected events, to the online services which monitor the performance of the CMS detector and also provide the means of archiving the events in mass storage.

Another crucial function of the DAQ system is the operation of a Detector Control System (DCS) for the operation and supervision of all detector components and the general infrastructure of the experiment. The DCS is a key element for the operation of CMS, and guarantees its safe operation to obtain high-quality physics data.

This Technical Design Report describes the current development of the design of a large-scale distributed computing system to provide the above functionality for the CMS experiment.

1.1 System Features

An analysis of the number of detector elements to be read out, the amount of data contained in the front-end electronics, the frequency at which events will be accepted by the Level-1 Trigger system and the computing-intensive subsequent event selection, yields requirements on the DAQ system of unprecedented scale in High Energy Physics experiments. In order to satisfy the very high performance requirements of CMS and its operation at the LHC, the CMS DAQ is a very large and complex computing system which requires extended development and integration. Although the other subsystems of CMS also involve the development of new technologies and techniques to satisfy the tight requirements of ex-

perimentation at the LHC, there are additional requirements, issues and externally-driven facts that differentiate the development of the DAQ system from the development of the other detector components.

Foremost among these characteristics is the fact that the DAQ system, by its very nature of being a computing system, relies on technologies that evolve extremely rapidly, even as this Technical Design Report (TDR) is being written. Figure 1-1 displays the evolution of some key technologies as a function of time.

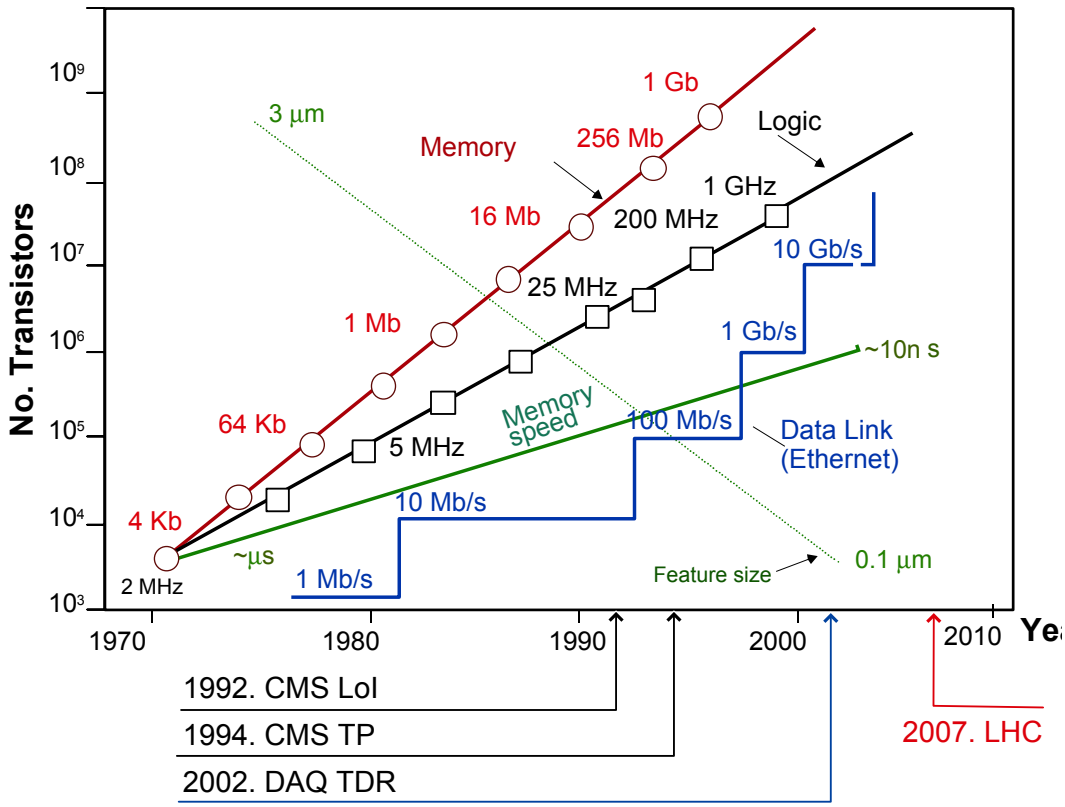


Figure 1-1 Evolution of network, processor and memory technologies as a function of time. Shown on the time axis are the times at which the CMS DAQ concept was described, namely the CMS Letter of Intent, the CMS Technical Proposal, and the DAQ Technical Design Report (this document).

In the ten year-period between the CMS Letter of Intent (in 1992) and the present Technical Design Report, network speeds have grown by a factor of 1000, the speed of CPUs has increased by a factor ~100, and memories are ~60 times larger. The implications of this are far reaching. First, it is certain that today's key computing and communications technologies will no longer be in the mainstream five years from now, while new technologies will have been introduced. This implies that, for the purposes of this TDR, the DAQ design, while complete, is not final. The design must be flexible and modular enough to facilitate the introduction of select new technologies into various parts of the system. Large monolithic (sometimes referred to as "vertical" solutions in the literature) cannot be adapted easily to utilize new technologies or devices that provide new functionality or increased performance. It is thus necessary to ensure that the system be distributed not only in geographic terms but also in function. A further implication is that the system design must rely on established industrial standards to the greatest possible extent. Home-grown hardware and software systems should be introduced only when commercially available solutions are clearly inadequate either in function or in performance. And, finally, given the impressive rate at which the cost of computer and telecommunications products continues to fall, the DAQ elements should be procured as late as possible, constrained only by the installation and commissioning schedule.

A second distinguishing characteristic of the DAQ system is that its required performance depends on the performance of the accelerator and of the detector. At the LHC start-up, once the commissioning of the accelerator is complete and stable machine running is achieved, the instantaneous luminosity is expected to be significantly below the design luminosity and to grow with time. Both the amount and the complexity of the data to be read out will therefore be reduced during this early phase of CMS operation. Coupled with the need to procure key parts of the system as late as possible, this implies that the DAQ design must be sufficiently modular to accommodate both planned and unplanned increases in machine performance. An ideal design, with optimized cost performance, would ensure the presence of a system that is just adequate for the operation of the physics program of the experiment at any instant in time. If too large a system is in place, then the experiment will have expended too many resources in securing performance at an unnecessarily high cost. On the other hand, if the system in place is not adequate for the detector and machine conditions of the time, the physics output of the experiment will suffer. Since it is impossible to know the precise experimental conditions in advance, the DAQ structure must be such that its performance can be increased in a straightforward manner with the installation of additional components, without any additional design and/or development time.

A third difference between the DAQ system and the other detector components is the near-certainty of major changes in the system over the lifetime of the experiment. Experience with past and present experiments in HEP clearly demonstrates that the DAQ systems of all recent collider experiments have undergone major upgrades during the lives of the corresponding experiments. Such upgrades are necessitated by a combination of reasons including, but not confined to,

- the opportunity to employ newer technologies that can increase the system performance, simplify its operation and increase its reliability,
- the inability to maintain out-of-date hardware,
- the desire to introduce new, unforeseen capabilities into the system,
- changes in the accelerator and experiment parameters.

There is every reason to believe that the CMS DAQ system will go through one or more upgrades as well.

1.2 Design Principles

Consideration of the issues discussed in the previous section as well as experience with the DAQ systems of previous experiments at high-energy lepton and hadron colliders result in the establishment of several fundamental design principles that have been embedded in the architecture of the CMS TriDAS and the DAQ more specifically, from the very beginning.

The most important decision is the number of physical entities, or “trigger levels”, that will be used to provide the selection of 1:1000 from the maximum average Level-1 output rate of 10^5 Hz to the final storage rate of 10^2 Hz. Current practice for large general-purpose experiments operating at CERN, DESY, Fermilab, KEK and SLAC is to use at least two more entities, colloquially referred to as the “Level-2” and “Level-3” trigger. Some experiments even have a “Level-4” trigger. The higher the number, the more general-purpose the implementation, with the Level-3 and Level-4 trigger systems relying on farms of standard commercial processors. The implementation of the Level-2 trigger system varies significantly across experiments, from customized in-house solutions to independent processor farms.

The problem encountered by all experiments that have opted for multiple trigger levels is the definition of the functionality that the Level-2 system should provide. Of all the trigger levels after Level-1, the Level-2 trigger is the most challenging one as it has to operate at the highest event rates, usually still without

the benefit of full-granularity and full-resolution data, though with data of higher quality than that used by the Level-1 Trigger. Decisions that have to be made are the rejection factor that the Level-2 must provide, the quality of the information it will be provided with, the interconnects between the detector Front-ends, the Level-1 Trigger and the Level-2 trigger, and finally, the actual implementation of the processing units that will execute the selection algorithms. Unavoidably, the entire design is very much dependent on the assumed machine conditions and the simulated detector response. Traditionally, the Level-2 trigger system has been designed as the last non-standard filtering step, with parameters driven by the desire to implement Level-3 and higher trigger levels as farms of commercial processors.

Ideally, the High-Level Trigger system should have no built-in architectural or design limitations other than the total bandwidth and CPU that can be purchased given the experiment's resources. Indeed, from very early on, the desire to provide the maximum possible flexibility to the HLT process led to the first design principle that the entire High-Level Trigger (HLT) selection should be provided by algorithms executed in a single processor farm, avoiding all questions and uncertainties concerning the functionality of a Level-2 trigger system. In the CMS HLT the physics algorithms have the maximum possible freedom in what data to access, the accuracy of this data, as well as the sophistication of the reconstruction and analysis tools to employ. The flow of data and the associated filtering elements, for both the traditional three-element Trigger option and the CMS choice are displayed schematically in Figure 1-2.

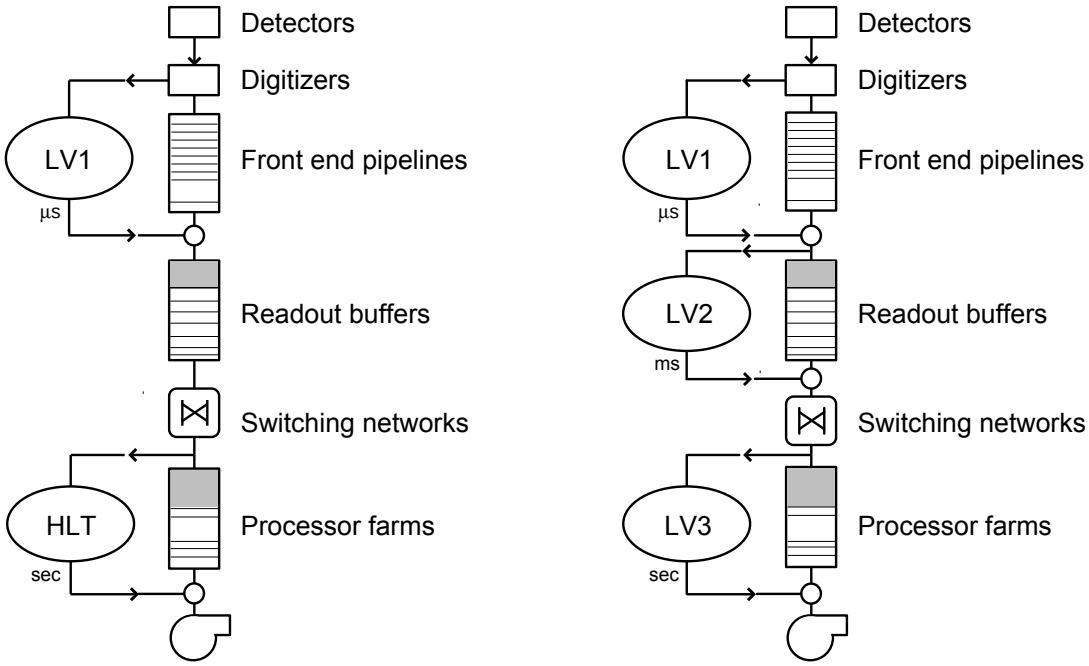


Figure 1-2 Data flow in the Trigger/DAQ system. Left: the CMS choice, with a single entity, the High-Level Trigger (HLT) providing all filtering after the Level-1 decision. Right: the HLT function is split into two stages (LV2 and LV3).

This first principle has been established after particular consideration of all of the issues discussed in Section 1.1. Foremost is the extraordinary rate of technological advances witnessed over the last twenty years, a rate that until now remains constant with time. It was decided to invest in this advance of technology and especially in the two main fronts that drive it, processing power and network speed. The decision not to build an independent Level-2 trigger but to incorporate it in a single processor farm results in a system that can benefit maximally from today's strides of technology. An additional consideration is the expected evolution of the experiment and its data acquisition system, rendering a fully programmable

High-Level Trigger system highly desirable to avoid major design changes. The added flexibility provided by the fully programmable environment of a standard CPU also implies that algorithmic changes necessary for the resolution of unforeseen backgrounds or other adverse experimental conditions can be easily introduced. A final consideration was the desire to minimize the amount of non-standard, in-house solutions. In fact, further analysis of this issue led to the second design principle of the CMS DAQ, described below.

The second design principle is the reliance on industrial standards to the greatest possible extent, as well as the use of commercially available components, if not full subsystems, wherever requirements can be met. This applies to both hardware and software systems. The benefits of this decision are numerous. The most important ones are the resulting economies in both the development and production costs, the prompt availability of the relevant components from multiple competing sources, and a maintenance and support mechanism that does not employ significant in-house resources.

A third general design principle, adopted at the very earliest stages of development, is that of maximal scaling. This addresses the fact that the accelerator conditions, the experimental conditions, and finally the physics program itself are all expected to evolve significantly with time. As will be explained in Chapter 3, "DAQ Architecture", an easily scalable system is one in which the functions, and thus the challenges as well, are factorized into subsystems whose performance is independent of the rest of the system.

A fourth and final principle has been that of decoupling the understanding of the required functionality of the various elements of the system from their performance. This led to the most challenging subsystem components being developed in two independent paths. In the first development path we concentrated on the identification and implementation of the full functionality needed for operation in the final DAQ. In the second, we concentrated on the issues that arise when the functions identified in the first path are executed at the performance levels required by the final DAQ system. Following this principle, CMS has pursued an R&D program which has resulted in a system that could be implemented even today and yet function at the early luminosities of the LHC since the system architecture is such that in a number of incremental steps, the performance of the system can be increased proportionally.

1.3 Software Systems

For a system which is based on commercial components, most of the development work is carried out in designing, writing and deploying software packages that facilitate the operation of the system. The extent of this project is such that a set of top-level guidelines and architectural decisions is necessary.

The online software infrastructure comprises all software elements that support the configuration, control and implementation of the event data flow procedures. The distributed nature of the system requires a set of services and application programming interfaces, commonly referred to as middleware [1-2], that are replicated in each processing node. Middleware constitutes a software layer that supports the applications with uniform mechanisms to access the available hardware and system services. Through the protocols that it implements, it allows applications to communicate across networks, system and programming language boundaries. The deployment of the middleware components introduces a uniform system environment with the following characteristics:

- a single programming model for all application modules,
- concurrent and transparent use of heterogeneous networking equipment through a single, inter-application programming interface,

- upgrade and replacement of application modules, without the need to modify interfacing software elements
- interoperability with different external subsystems, such as various run-control services.

The applications communicate with each other through the middleware to perform the data acquisition tasks. They also interact with the run-control and the detector-control services for configuration and control purposes, as shown in Figure 1-3.

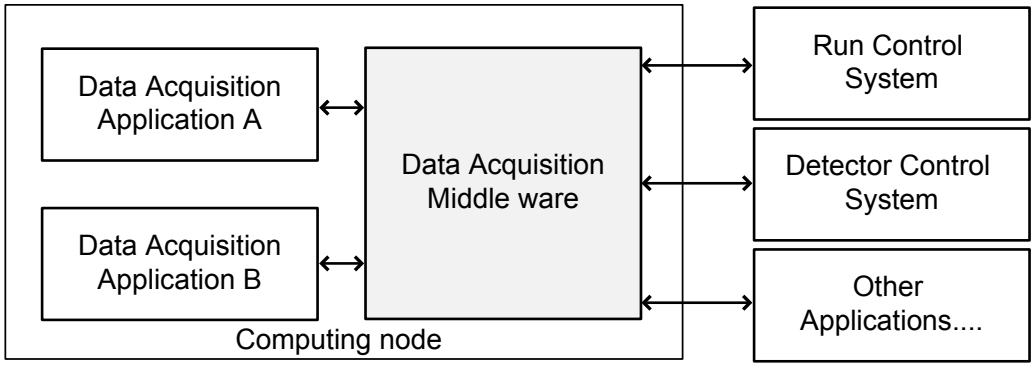


Figure 1-3 Online software on a computing node and its interfaces to external systems.

From the many pervasive attributes that software has to have, a middleware solution for distributed data acquisition systems must fulfill at least three requirements: efficiency, flexibility and system management.

The middleware must fully exploit the communication hardware through the use of efficient algorithms, such as zero-copy operations, buffer-loaning and data block chaining operations. The overhead introduced by the middleware must not be significantly larger than the overhead generated by a custom built application, while offering the same functionality. The middleware must not prevent optimizations for a given application scenario by imposing a particular programming model (e.g. client/server).

The middleware must also provide enough flexibility to allow software changes and hardware replacements without disrupting other software components (isolation). It must be possible to include new protocols and qualities of services with no additional programming effort (extension).

The third dimension considered is system management. The software infrastructure must support the configuration and steering of all components, hardware, middleware elements and applications, through a common scheme. This scheme must be open, i.e. based on a widely agreed standard, and documented, to allow the adaptation to as yet unanticipated external systems. A software architecture that covers the described requirements is outlined in Chapter 10, "Online Software".

1.4 Uncertainties and Likely Evolution

At the time of writing, the first LHC collisions are planned to occur in 2007, with the first “physics” run reaching an instantaneous luminosity of $2\times10^{33}\mathrm{cm}^{-2}\mathrm{s}^{-1}$. This first run is currently planned to last several months, but long enough for $10\mathrm{fb}^{-1}$ to have been delivered to the CMS and ATLAS experiments. Assuming this schedule remains in place for the coming four years, the schedule of the DAQ system is driven by the installation schedule of the full CMS detector. In this schedule, the pit at LHC Point 5, where CMS

will be installed, will be “delivered” to the experiment in July 2004. In Spring 2005 the installation of the electronics systems will begin. At that point the various electronics modules will need to be read out and eventually integrated into the full readout of the experiment. Most of these front-end electronics modules are being constructed today.

With this schedule, the parts of the DAQ that are linked to the detectors should be considered as final. The remaining 1.5 years until the beginning of the installation at Point 5 will be required for the actual implementation of the readout system described here. Changes can, however, still be accommodated in the higher system parts, i.e. in the computer farm, in the networks interconnecting the various DAQ elements, and in the control and monitor systems. For these elements, this Technical Design Report should not be seen as a blueprint of an engineering design but rather as the description/documentation of the current stage in the development of the system.

1.5 References

1-1 CMS Coll.,“The Trigger and Data Acquisition project, Volume I, The Level-1 Trigger, Technical Design Report”, CERN/LHCC 2000-038, CMS TDR 6.1, 15 December 2000.

1-2 P. Bernstein, “Middleware: A Model for Distributed Systems Services”, Communications of the ACM, 39 (1996) 86.

2 Requirements

The CMS experiment employs a general-purpose detector with nearly complete solid-angle coverage. The physics program of CMS, and its discovery potential in particular, is thus very broad. As stated in the overview, the main purpose of the Data Acquisition (DAQ) and High-Level Trigger (HLT) system is to read the CMS detector information for those events that are selected by the Level-1 Trigger, and to select, from amongst those events, the most interesting ones for output to mass storage. The proper functioning of the DAQ at the desired performance will be a key element in reaching the physics potential of the experiment. In addition, to maximize this physics potential, the selection of events by the HLT must be kept as broad and as inclusive as possible. In this chapter the main physics and system requirements on the DAQ/HLT system are reviewed. The requirements listed here are not an exhaustive set, in the sense that a commercial firm provided with this set could design and produce the complete DAQ system. The aim of this chapter is to highlight the top-level, or "user-level", requirements of the DAQ. Technical requirements arising from details of implementation, as well as requirements that are well-known in the context of DAQ systems in High Energy Physics experiments are assumed to be known to the reader.

2.1 General Description of Environment and System Parameters

The Level-1 Trigger has to process information from the CMS detector at the full beam crossing rate of 40 MHz. The time between two successive beam crossings (25 ns), along with the wide geographical distribution of the electronic signals from the CMS detector renders the real-time processing of the full detector data by general-purpose, fully programmable processing elements impossible. Moreover, the time available for processing in the Level-1 Trigger system is limited by the resources available in the front-end electronics that store the detector data during the Level-1 decision-making process. Channel count and technology considerations dictate that some of the front-end electronics can store (in pipelines, i.e. FIFOs) the data from at most 128 continuous bunch crossings, i.e. the equivalent of approximately 3 μs of data. This 3 μs is the maximum time interval for a Level-1 decision to be received by the FE electronics. It therefore includes the unavoidable latencies associated with the transfer of the detector information to the processing elements of the Level-1 Trigger, as well as the latency of the propagation of the Level-1 decision signals back to the FE electronics. The resulting time available for actually processing the data is no more than ~1μs. Therefore the Level-1 Trigger can only processes data from a subset of CMS subdetectors, the calorimeters and muon chambers. Moreover, this data do not represent the full information recorded in the CMS FE electronics, but only a coarse-granularity and lower-resolution set. The processing elements of the Level-1 Trigger system are custom-designed. Details on the architecture and design of the Level-1 Trigger can be found in reference [2-1].

Given the total time available to the Level-1 Trigger processors, and the information sent to them, it is expected that, at the highest LHC luminosities, the Level-1 Trigger system will achieve a crossing reduction factor of 400, for a maximum mean event rate of 100 kHz. The next selection step, the High-Level Trigger, will therefore receive, on average, one event every 10 μs. This time is sufficiently long for the storage of all data from the FE electronics to be feasible in commercial, random-access memories. Experience with equivalent filtering processes from previous experiments in High Energy Physics indicates that further event selection has to be based on better granularity and resolution information than that available to the Level-1 Trigger, and to also include information from tracking detectors. Moreover, the algorithms employed must be almost as sophisticated as those used in the offline reconstruction of the events. This implies the usage of fully programmable commercial processors for the execution of the HLT. The current best estimate for the total average processing time required for the design selection of 1:1000 (i.e. a rate reduction from 100 kHz to 100 Hz) is roughly 40 ms, with some events requiring up to 1 second. This in turn dictates that the CMS Data Acquisition system must provide significant buffering capability.

In summary, the online event filtering process in the CMS experiment will be carried out in two steps.

- The Level-1 Trigger, with a total processing time of 3μs, including the latencies for the transport of all data and control signals. During this time interval, the data are stored in pipeline memories in the front-end electronics. The Level-1 Trigger is designed to accept a maximum rate of 100 kHz.
- The High-Level Trigger, with a total processing time of up to ~1 s. During this time interval, the data are stored in random-access memories. The High-Level Trigger is designed to output a maximum mean event rate of ~100 Hz.

The architecture for a DAQ system that will provide the above functionality, is constrained by the following considerations.

- The total amount of data in a CMS event: current estimates are that the full data set recorded during a single bunch-crossing is about 1 MB.
- The number and location of the modules that store this data after the acceptance of an event by the Level-1 Trigger: current estimates, taking into account the number of channels in the CMS subdetectors and the modularity of the various front-end electronics modules, result in roughly 700 such modules, each carrying 1-2 kB of data per Level-1 Trigger accept.
- The maximum rate of events output by the Level-1 Trigger: as stated above, the Level-1 Trigger is designed to output a maximum event rate of 100 kHz.
- The reduction rate demanded out of the High-Level Trigger selection: limitations in the ability to store and reconstruct data offline result in a maximum rate of $O(10^2)$ Hz for events to be accepted by the DAQ. This factor of 1,000 rejection requires that full-fledged physics algorithms, running on essentially fully-reconstructed detector data, be employed. Actual measurements of the CPU required for the full HLT selection using today's Pentium-III processors running at 1GHz yield a mean time of ~300 ms per event passing the Level-1 Trigger[1]. At a maximum Level-1 output rate of 100 kHz and using benchmarks that measure the Pentium III 1 GHz CPU capacity at 41 SI95 [2-2], the measurement implies a total computing power of $\sim 1.2 \times 10^6$ SI95 is required.

The expectation that the sophistication of the algorithms employed by the High-Level Trigger will demand a mean processing time of $O(10^{-2})$ s, along with the maximum event rate of 10^5 Hz, implies that of $O(10^3)$ processors must be employed for this processing stage. This, in turn, implies that the DAQ must provide the means to connect ≈700 modules to about 1,000 processors. The interconnection of such a large number of elements necessitates the introduction of a switching network.

The most stringent requirement on the DAQ system results from (a) the introduction of this switching network, derived from the HLT specification, and (b) the basic performance requirements stated previously. In summary, the DAQ system must provide the means to feed data from 700 front-end modules to $O(10^3)$ commercial processors, at a sustained bandwidth of up to 10^5 Hz · 1MB = 100 GB/s.

The main parameters of the CMS Trigger/DAQ System are listed in Table 2-1. There are no changes with respect to the corresponding parameters listed in the CMS Technical Proposal [2-3].

1. This is discussed extensively in Chapter 15, "Physics Object Selection and HLT Performance". The discussion includes the extrapolation of the 300 ms/event measured in 2002 to the corresponding CPU times of 40 ms/event for the year 2007 mentioned earlier.

Table 2-1 Nominal parameters of the CMS DAQ system.

Parameter	Value
Level-1 Trigger rate	100 kHz
Event size	1 MB
Event Builder bandwidth	100 GB/s
Event Filter Computing Power	10^6 SI95
Data Production	10 TB/day
Number of Front-ends	700

2.2 Physics Requirements

The High-Level Trigger must reduce the event rate output by the Level-1 Trigger by a factor ~ 1000 for a total output to storage of ~10^2 Hz. At the design luminosity of the LHC this total expected to be output to mass storage, corresponds to a cross section of 10nb. Given that the $W^+ \rightarrow e^+\nu_e$ production cross section alone is of this order, a significant physics selection has to take place online. It is this aspect of the HLT system that places the most stringent requirements on the system.

The main requirements on the system are thus:

- The system has to provide enough bandwidth and computing resources to minimize the dead-time at any luminosity, while maintaining the maximum possible efficiency for the discovery signals. The current goal is to have a total dead-time of less than 2%. Half of this dead-time is currently planned to be spent in the Level-1 Trigger system.
- The HLT system should tag all events selected as candidate events for specific physics processes. This information can then be used by the offline system for a quick sorting of the events from the experiment.
- The system should allow for the readout, processing and storage of events that will be needed for calibration purposes. It should also be robust, i.e. its efficiency should not depend strongly on changes of the calibration and detector alignment constants.
- It must be possible to validate the trigger and to compute the overall selection efficiencies using only the data itself, with as little reference to simulation as possible.
- The system has to be flexible enough to adapt to changing run and/or fill conditions. As an example, the instantaneous luminosity is expected to drop in the course of a fill, and therefore an optimal allocation of resources might be to change trigger conditions, for instance by lowering trigger thresholds or decreasing pre-scale factors for select channels. All such changes, along with any other changes in the running conditions, must be logged.
- The system should provide enough resources to monitor the status of the CMS detector, and to provide enough information to the experimenters in case of problems.
- To maximize the efficiency of the filtering process, an event should be rejected as soon as possible. Furthermore, the system should not rely on the presence of all the information from the CMS detectors. Such information will be required only for events selected for storage.

Given the unprecedented rate of online rejection, a most important task of the HLT is, after achieving this rejection, to provide enough information on what is rejected. It is for this reason that, a major part of the HLT system will be the control and monitoring of the processor farm.

2.3 System Requirements

The CMS DAQ has to read out all the front-end electronics, assemble the data from each bunch crossing into a single standalone data structure, provide this data to processing elements that execute the HLT selection and forward the events accepted by the processor farm to the online Computing Services which contain, among others, the mass storage. The formal requirements on the system are characterized according to their source. Requirements on the DAQ arise from the characteristics of the CMS detectors, from the HLT system and from the interface to the experimenters.

2.3.1 DAQ Requirements from the CMS Sub-detectors

The DAQ system has to read out all the detector front-end electronics modules, which are quite heterogeneous in terms of functionality, buffer management, details of the readout process and multiplicity. The main requirement is thus to interface to this multiplicity of different modules in a seamless fashion.

As will be explained in detail in later chapters, interfacing to this set of disparate modules is possible by introducing a common interface that all front-end modules (the “FED”s) in the CMS readout have to use. This interface has been kept as simple as possible to ensure that maximum flexibility is left to the developers of the Front-ends.

2.3.2 DAQ Requirements from the HLT

The requirements that arise from the filtering function, i.e. from the High-Level Trigger (HLT) system, are direct by-products of the physics requirements listed in Section 2.2. In brief, the DAQ has to provide:

- the data from all the detectors of CMS to a processor capable of analysing the event, in order to select the “best” events;
- enough computing resources to execute the HLT algorithms at the design luminosity;
- enough buffering in the system so that the mismatch between the time between two successive events from the Level-1 Trigger and the latency for a HLT decision is absorbed without introducing deadtime beyond the overall design goal of 2%;
- a guarantee for the correctness and integrity of data provided to the HLT processors, with any errors in transmission or generation of any of the data identified and flagged as such in the data stream itself.

2.3.3 DAQ Operational Requirements

- The DAQ should be partitionable into smaller, independent DAQ systems that involve mutually exclusive parts of the readout and other online resources. One clear usage of this property is the simultaneous commissioning and debugging of multiple detector subsystems.

- It should provide the means to control and monitor the entire process of data-taking with the CMS detector in a manner that allows the operation of the system by a wide set of physicists from the CMS collaboration, and not only by experts who know the DAQ in detail.
- It should allow for remote control and monitoring of essentially all elements of the CMS detector.
- There should be ample monitoring of the quality of data, as well as the quality of the data selection processes (i.e. all levels of triggering) on time scales that allow for quick recovery from potential problems with minimal loss of data.
- It should provide means for recording data in isolation for a long enough period (~one week) even if the connection of the experiment to the rest of the CERN site is not functioning properly.
- It should provide safety mechanisms that raise timely alarms that protect the CMS detector, its electronics, and all associated hardware (e.g power supplies) from adverse situations. In emergency situations, the system should automatically take the appropriate action (e.g. turning off electrical power to a subsystem) without human intervention.
- The control and monitoring of numerous detector parameters should be carried out even in the absence of beam in the accelerator.

2.4 Summary

The DAQ system provides, beyond the readout of the information from the CMS detector, the first place where the entire information from the physics collisions can be inspected. As such, it is the one place where complex decisions regarding the fate of each event accepted by the Level-1 Trigger can be made. It is also, then, the one place where the complete picture of the detector response to the collisions can be monitored, thus providing early feedback to physicists running the experiment. The DAQ is therefore the one system of CMS that implements the two crucial functions that eventually determine the reach of the physics program: event selection, and control and monitoring of the CMS detector elements.

The design of the DAQ must therefore address widely different requirements, varying from the fast transfer of large amounts of data, to providing resources for the intelligent filtering of this data, to recording the selected data, to providing intelligent monitoring information, and finally to presenting an intuitive, functional and powerful interface to physicists running the data taking. Above all, the system must be flexible and extendable, both in performance and functionality, for it is clear that not all requirements can be identified at this point, and much remains to be discovered with the advent of the first collisions in the LHC. In the following chapter we describe the architecture that has been chosen in order to achieve these two characteristics, i.e. flexibility and extendability.

2.5 References

2-1 CMS Coll.,“The Trigger and Data Acquisition project, Volume I, The Level-1 Trigger, Technical Design Report”, CERN/LHCC 2000-038, CMS TDR 6.1, 15 December 2000.

2-2 Standard Performance Evaluation Corporation, http://www.spec.org. The Pentium-III figures can be found at http://www.spec.org/osg/cpu95/results/cpu95.html

2-3 CMS Coll.,”The Compact Muon Solenoid, Technical Proposal”, CERN/LHCC 94-38, LHCC/P1, 15 December 1994.

3 DAQ Architecture

The architecture of the CMS DAQ system is shown schematically in Figure 3-1. The detector front-end electronics are read out in parallel by multiple units that format and store the data in deep buffers. These buffers must be connected to the processors in the HLT farm, and this is achieved by the large switch network (Builder Network) shown in Figure 3-1. Two systems complement this flow of data from the Front-ends to the processor farm: the Event Manager, responsible for the actual data flow through the DAQ, and the Control and Monitor System, responsible for the configuration, control and monitor of all the elements. The Computing Services include a host of monitoring services, storage and the interface of the DAQ to the offline environment.

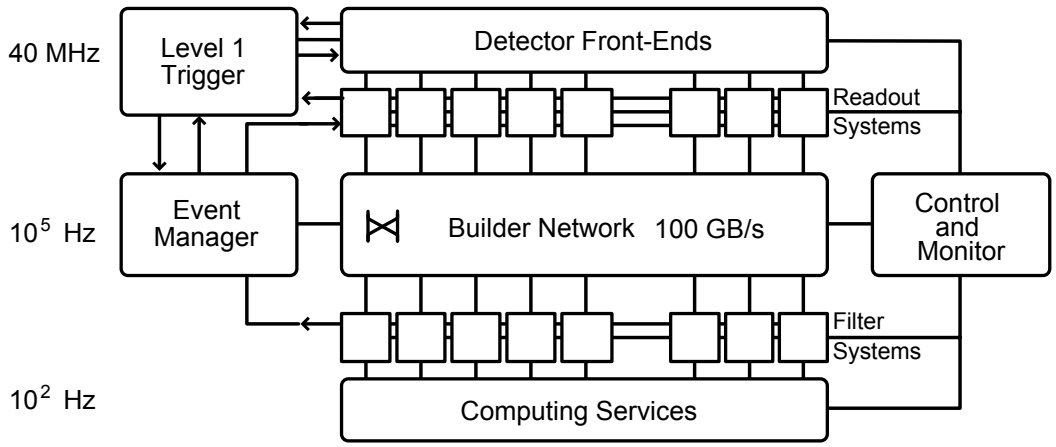


Figure 3-1 Architecture of the CMS DAQ system.

The CMS DAQ system consists of the following elements:

- Detector Front-ends: the modules that store the data from the detector front-end electronics upon the reception of a Level-1 Trigger accept signal. There are approximately 700 such modules in the CMS readout. The Front-ends are the responsibility of the corresponding subdetector.
- Readout Systems: the modules that read the data from the detector Front-End System (FES). The data are then stored, until they are sent to the processor which will analyse the event. There are approximately 500 entities, referred to as “Readout Columns”, in this system. Each Readout Column contains a number of Front-End Drivers and one Readout Unit which is responsible for buffering the event data and for interfacing to the switch.
- Builder Network: the collection of networks that provide the interconnections between the Readout and the Filter Systems. It is a large switching fabric, capable of supplying 800Gb/s sustained throughput to the Filter Systems, as well as all the control information necessary to sustain this data traffic.
- Filter Systems: the processors which are provided with events by the Readout. They execute the High-Level Trigger algorithms to select the events to be kept for offline processing. There are approximately 500 entities, referred to as “Filter Columns” in this system. Each Filter Column consists of one Builder Unit, which receives the incoming data fragments corresponding to a single event and builds them into full event buffers, and a number of Filter Units, which are the actual processing elements for the High-Level Trigger algorithms. A single Filter Unit will contain multiple processors.

- Event Manager: the entity responsible for controlling the flow of data (events) in the DAQ system. Strictly speaking, such an entity is not necessary: the DAQ could be seen as a loosely coupled, fully distributed system in which the status of each event is maintained independently by each Readout Unit. However, the introduction of a centralized intelligence for event management simplifies the overall system synchronization considerably.
- Computing Services: all the processors and networks which receive filtered events, as well as a small fraction of rejected events, from the Filter Farms.
- Controls: all the entities responsible for the user interface and the configuration and monitoring of the DAQ.

Concentrating on the functional breakdown of the system's components, one identifies four stages:

1. a detector readout stage, which results in storing the event data in a collection of ~700 buffers;
2. an event building stage, in which all the data corresponding to a single event are collected from these buffers via the switch;
3. a selection stage, in which the event is processed by the High-Level Trigger in the processor farm;
4. an analysis/storage stage, in which the events selected by the HLT are forwarded to the computing services for further processing either for storage or for further analysis, be it for detector/trigger monitoring or for calibration purposes.

The factorization of the DAQ function into four tasks which can be made almost independent of each other facilitates the design of a modular system which can be developed, tested and installed in parallel. To ensure this factorization, one must decouple the different operational environments of the four functional stages. This, in the CMS design, is achieved via the introduction of buffering of adequate depth in between each of these stages. The primary purpose of these buffers is to match the widely different operating rates of the elements at each stage.

As an example, at a rate of 100,000 events per second, the readout system delivers an event every 10μs. On the other hand, the event building process requires, even assuming a 100% efficiency of link utilization, a time of 1MB/2Gb/s = 4ms to completely read in the event. This is therefore the rate at which the elements of the farm system can operate on events. The two time scales are very different, and this is where the deep buffers present in the readout system serve to decrease the coupling between the stages.

3.1 Event Builder

The Event Builder, the central component in the DAQ system, consists of all the entities involved in collecting the data from the various readout buffers into a single "event buffer", which is then sent to a processor for analysis. Its functions therefore include: part of the readout subsystem, the switch fabric, part of the farm, and the Event Manager.

At a maximum sustained event rate of 100 kHz and a mean event size of 1MB, the total sustained throughput of the switch fabric should be 800 Gb/s. The minimum number of 1Gb/s ports needed would be 1600 (800 ports for the inputs and 800 for the outputs), or 800 for 2Gb/s ports. This is roughly one order of magnitude larger than any fabric commercially available. Furthermore, the commercial demand for very high numbers of ports, on a single switch, is limited. Second, the switch as defined is one that will operate at a very high load. Assuming 1000 ports of 2Gb/s each, the switch will carry a load of 80%. Compared to the networks commonly deployed, and that typically operate on loads close to 20-30%, this

is a very high load indeed. Furthermore, it is not clear how to obtain this high efficiency of operation for such a large switch given the precise traffic pattern of the data in an Event Builder[1].

3.1.1 Choice of Networking Technology

In a system like the one drawn in Figure 3-1 the details of the switch fabric will clearly determine the requirements and thus the functionality of both the readout and filter elements, along with their associated controls. In the initial CMS Event Builder design, the switch fabric was conceived as a large network of commercial switches appropriately interconnected to deliver the total sustained throughput required for the HLT. The details of this network were investigated for various network topologies including a study of their efficiency.

The choice of technology for the switch network is a central one in the design of the DAQ Event Builder. Experience gained from DAQ systems of previous HEP experiments indicates that the costs and logistics associated with the maintenance of large home-grown computing and/or networking systems can be considerable. Moreover, the current explosion of the processing and network technologies indicate that the system should be based, as much as possible, on commercial, off-the-shelf components. Early on, it was decided that the in-house development of a large O(1000) port switch fabric for the CMS DAQ was not an option to be pursued.

Today, the state-of-the art in data-transport technologies includes the following technological options:

- ATM (Asynchronous Transfer Mode): this is the interconnect technology developed by the telecommunications industry. Currently available ATM products include point-to-point links running at 155 Mb/s. At present faster links exist, running at 620 Mb/s, but at a significantly higher cost. Switches accepting both speeds are available from multiple vendors. The ATM standard also includes link speeds of 1.2 Gb/s and 2.4 Gb/s. Today there exist no products (network interface cards for a standard bus like PCI) at these link speeds. Perhaps the greatest advantage of ATM technology is the set of built-in Quality of Service (QoS) capabilities. In the case of Event Builder traffic this can be very beneficial. Unfortunately, the evolution of the cost of ATM technology has not been very encouraging, at least when compared to other technologies like Gigabit Ethernet.
- Ethernet: this is the familiar technology that has contributed much to the expansion of the Internet. Standard Ethernet links run at 10 Mb/s, whereas Fast Ethernet links run at 100 Mb/s. Recently, Gigabit Ethernet, offering links at 1Gb/s, has been introduced, while 10-Gbit Ethernet was introduced in 2002. The main advantage of this technology is its broad deployment (roughly 90% of all desktops with a dedicated network connection use 10 or 100 Mb/s Ethernet), which leads to the expectation that the associated Gigabit version may eventually emerge as the leading technology, with all the implied cost efficiencies, in the years after 2002.
- Fibrechannel: this is the technology developed by the computer industry, and is mainly deployed in association with high-speed storage devices. Currently available products include links running at 1Gb/s. Switches accepting multiple (up to 64) ports are also available. At this point, it is not clear whether Fibrechannel will remain a storage-related technology or whether it will be deployed more widely in generic computer networks.

1. As explained in Chapter 6, a key feature of the data traffic pattern in an EVB is that all data sources must communicate with the same destination for each event. This is not the typical intranet where point-to-point connections are established across the network in a fairly random fashion. The EVB data traffic therefore, has a very systematic pattern which leads to a far more severe bottleneck at the destinations than in a common intranet. This, in turn, in the absence of any additional mechanism for shaping this traffic, typically results in a lower efficiency of usage of the switch fabric.

- Alternative, proprietary networks developed by individual companies: as an example, Myrinet, a product of Myricom, is another Gb/s technology offering a full suite of both network interface cards and switches at relatively low cost. Myrinet has been deployed successfully as the interconnect for large processor farms. The main issue for such proprietary systems is whether they will remain competitive over a timescale of five to ten years.
- New emerging standards that may very well have standard commercial products available in time for the CMS choice of technology. A prime example of this is the new Infiniband architecture which is centred around the concept of switches interconnecting intelligent devices.

The CMS DAQ group has carried out tests of switching systems in an event builder test-bench for all of the above technologies, with the exception of Infiniband. Following a first set of studies, two technologies, Gbit Ethernet and Myrinet, were selected for extended studies. The results of these tests and the associated conclusions and design decisions are reported in Chapter 6, "Event Builder Networks". For the purpose of this discussion on the architecture of the system, only the broad implications of deploying one of these technologies will be considered.

The choice of switching technology will be based on numerous considerations including, but not confined to, the overall efficiency of the switching network, its cost of procurement and its maintainability. This decision should be made as late as possible, constrained only by the time for the DAQ system installation and commissioning. At present estimates, a decision must be made roughly 2-3 years before the deployment of the CMS DAQ. Nevertheless, despite the relatively long (by technology lifetime standards) time between the writing of this document and this decision stage, the above list of technologies can be used as a guideline in planning the CMS DAQ and in establishing a “baseline” architecture. Given the existing technologies, it is fair to assume that the switching network will have individual links operating at either 1Gb/s or 2Gb/s. It is further assumed that if 1Gb/s links are used, then the DAQ elements will be able to use two of them, thus achieving a 2Gb/s effective link speed, irrespective of technology.

3.1.2 Deployment Issues and Event Builder Breakdown

Assuming an Event Builder with ~500 data sources and ~500 data consumers, appropriately interconnected by a large switch network, there are two architectural possibilities as shown in Figure 3-2.

- A system in which the switch network is provided by a multi-stage arrangement of switches with a smaller number of ports. As an example, a 500×500 Event Builder network can be implemented using two stages of 46 Gigabit Ethernet switches with 46 ports each[1]. This is the “all-network” scenario, where right after the basic readout of the Front-ends, the readout depends on a large network to provide the connectivity needed to transport the event to a processor. This corresponds to Configuration A in Figure 3-2.
- A system in which this large network includes an intermediate buffering stage, as in Configuration B in Figure 3-2. The advantages of such an intermediate stage is the resulting increase in the basic data sizes that have to be transported through the switch, as well as another level of decoupling between transport processes with different time scales.

1. The arrangement is a Banyan (delta) network where there are two stages each containing 23 switches. The switches utilized have 46 ports and are used in 23×23 configuration (i.e. 23 inputs and 23 outputs). The routing is as follows: output port *i* from switch *j* in the first stage is connected to input port *j* of switch *i* in the second stage. It is easy to see that this network provides all possible communication paths, and that the path between any two nodes is unique. Clearly, this network can block not only on the outputs of the switches of the second stage, but also on the outputs of the switches of the first stage.

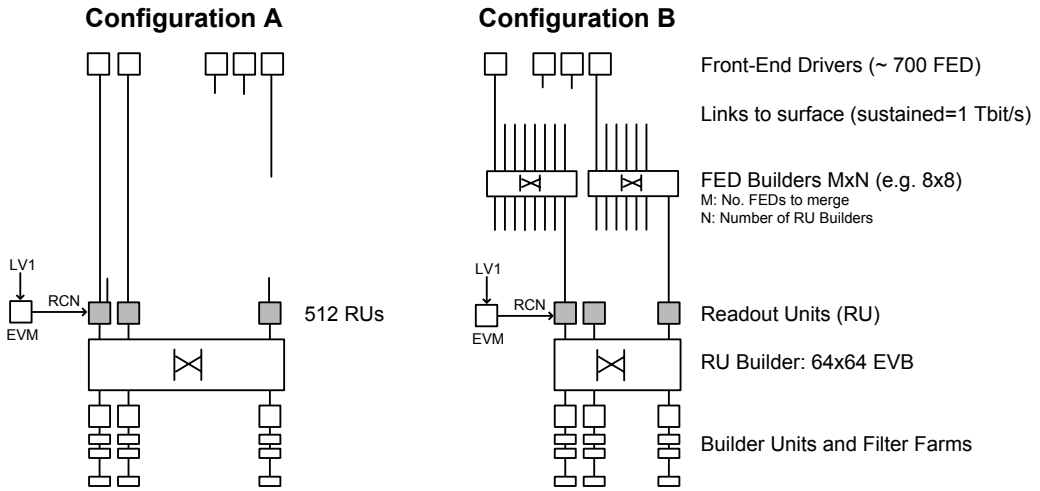


Figure 3-2 Schematic representation of the two main architectures of the switch network. Configuration A contains a single switching stage that connects 500 data sources (RUs) to the filter farm via a single fabric. The fabric may be a composite one, i.e. it may consist of multiple stages of switches. There is, however, no explicit buffering added in between such possible intermediate stages. In configuration B there are two switching stages which are separated by a buffering layer (the RU).

Normally the overheads both in additional costs and in development would tend to favour the first option, in which there is no buffering between the two switch stages, beyond that already provided by the switches themselves. There are, however, a number of considerations that tend to favour Configuration B.

A major issue is the modularity of the system and its effect on the scalability as well as on the installation sequence. An important uncertainty for the DAQ is the exact size of the system to be installed for the first physics run, as well as the evolution of the accelerator and experiment performance. Intricately linked to this are the actual resources that will be made available to the DAQ project near 2004-2005 for the procurement of the bulk of the start-up system. The modularity of the system with an intermediate buffering stage is greater and allows for a phased installation of the DAQ. In addition, this option decouples the choice of technology for the switches used for the two stages.

The system of configuration B can be implemented in steps of eighths of the full system that would connect 500 readout to 500 collections of processors, whereas the system of configuration A has to be implemented either with the full number of inputs and a reduced switching network at the beginning, or the number of inputs from the detector has to increase as the system size increases. The actual number of elements need to achieve the required scaling is now quadratic in this second case. This is shown schematically in Figure 3-3.

Given these features, configuration B has been chosen as the baseline. In what follows, this expansion scenario is referred to as the “three-dimensional” scenario or upgrade path. Similarly, the design itself is referred to as the “three-dimensional”, or 3-D for short, Event Builder.

3.1.3 Event Builder Layout

The conceptual design of the baseline 3-D Event Builder is shown in Figure 3-4. The Front-End Drivers (the modules connected to the front-end electronics, see Chapter 7, "Readout Column") are connected to Readout Units via a small (8×8) switch network - labelled as “FED Builder” in Figure 3-5. For any event in the system, there are eight potential destinations, as far as the Front-ends are concerned. Each one of

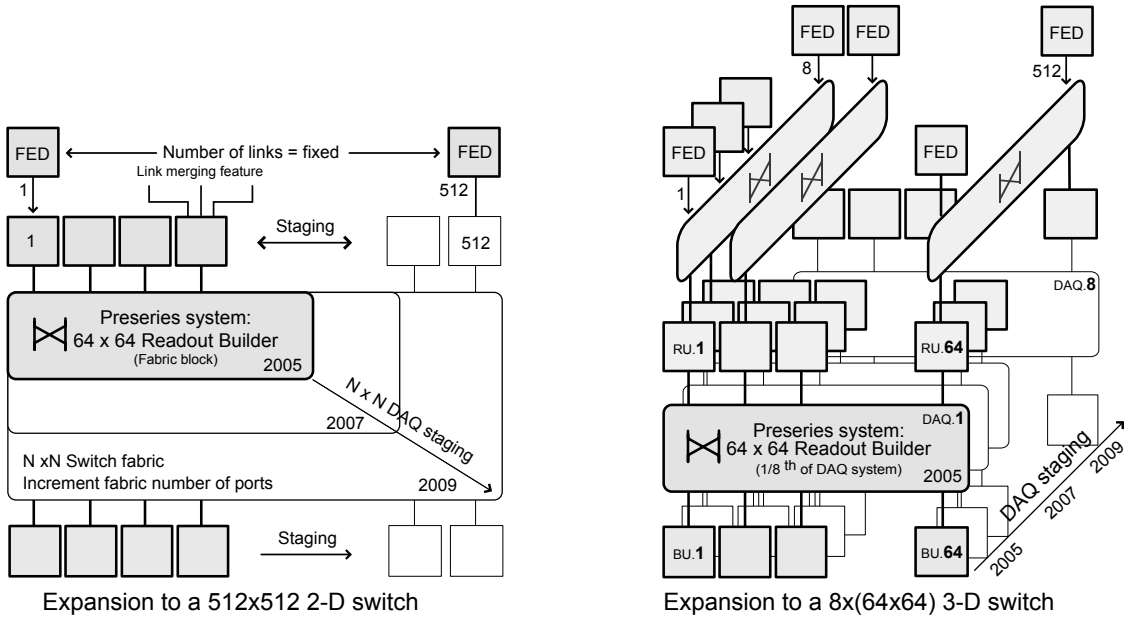


Figure 3-3 The two expansion possibilities for the DAQ Event Builder. The starting point is a 64×64 Event Builder, with 1/8 of the performance of the final system. In one upgrade path the switching network increases in the number of ports. Alternatively, the system is duplicated and obtains data from the same sources as the first system. The dates listed on the figure are purely for illustration purposes.

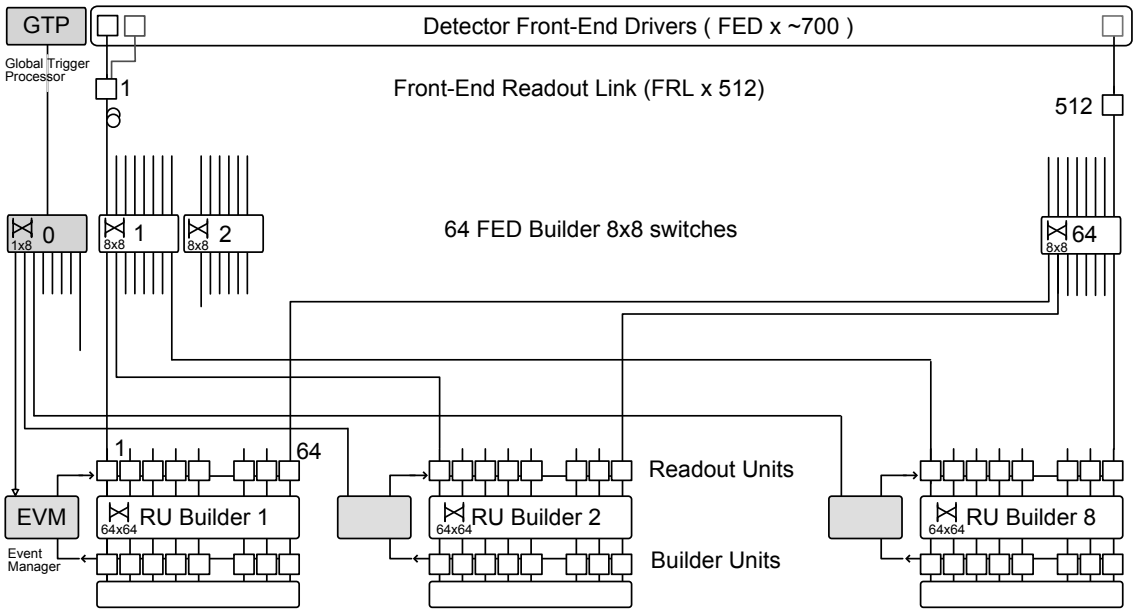


Figure 3-4 Conceptual design of the CMS Event Builder (EVB). There are two stages in the process: in the first, collections of 8 FEDs are multiplexed to one of eight independent collections of Readout Units. There are, therefore, 64 such “FED Builders”. At the end of stage 1, the data from a single event are contained in 64 Readout Units. These are connected, in turn, via a “Readout Builder” switch, to the second stage where the full event is put together. There are 8 RU Builders in the full system.

the Readout Units (RU) is therefore a point of merging eight independent data buffers into mini-events. Given the parameters of the CMS system, each RU will thus build events with a mean size of 8×2 kB=16

kB of data. The set of eight FEDs, the small 8×8 switch and the part of the Readout Units receiving events from the switch constitute a small Event Builder, referred to as the “Front-End Driver Builder” or FED Builder for short.

Once an event has been forwarded from the FEDs to the 64 Readout Units which are connected to the second, larger builder network, the Readout Units proceed to send their fragments for this event to a Builder Unit via the large (64×64) data network also shown in Figure 3-4. The collection of the part of the Readout Units responsible for sending the data to the large switch and the part of the Builder Units that receive the event fragments and collate them into full events is referred to as the Readout Unit Builder or RU Builder for short.

In summary, the Event Builder complex consists of two stages. In the first, the data from up to eight FEDs are combined into a single super-fragment collected by a Readout Unit (RU). There are a total of 64 such systems, one per RU, and thus there are 64 FED Builders. The RUs are in turn connected to an equal number of Builder Units which receive the 64 event fragments and build the final single-event buffer. This complex constitutes the RU Builder. There are eight RU Builders in the full CMS system.

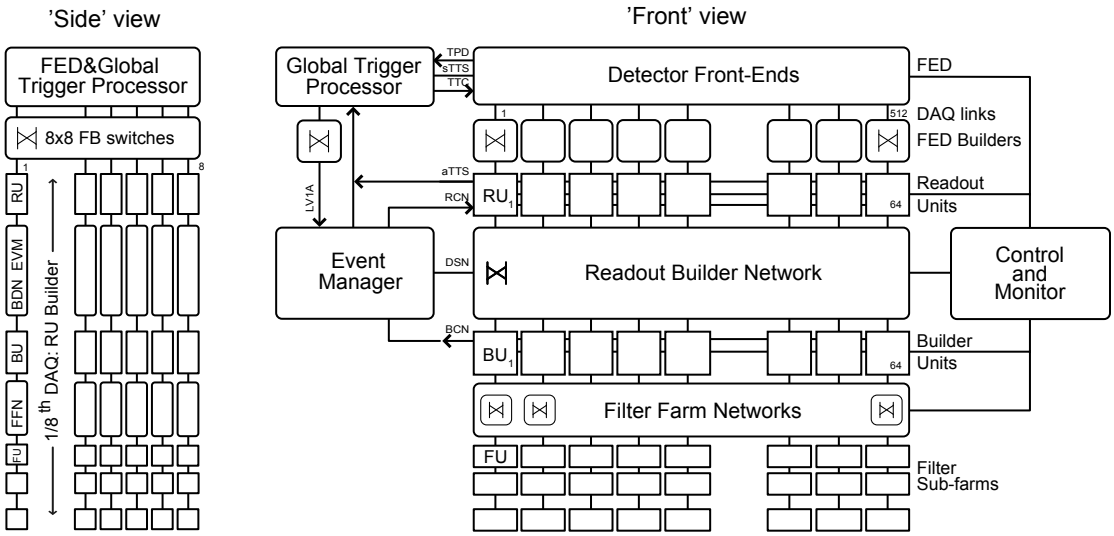


Figure 3-5 The two views of the full Event Builder. On the right-hand side is the standard event builder complex, very similar to the one shown in Figure 3-1. This is the “Readout Unit Builder”. On the left is a schematic of the multiple RU Builders and the way they are connected to the Front-ends via a small switch (8×8) which routes each event to the appropriate RU Builder. This small switch is referred to as the “FED Builder switch”.

An alternate view of the system is given in Figure 3-5, showing the independence between the two stages. As a result, the second stage may consist of a single RU Builder, with the FED Builders of the first stage all sending their data to a single FED Builder Output. This multiplexing of the FEDs to different RU Builders establishes a natural split of the Event Builder into eight independent, near-autonomous Data Acquisition systems with numerous advantages.

- It allows the decoupling, in time, between the decision for a switching technology for the Event Builder from the need to have enough of the DAQ system in place in mid-2004 when the CMS installation will start. With this breakdown, only the small FED Builders need to be determined early (i.e. as of the writing of this TDR), while the technology to be used for the RU Builders can be determined at a later stage. The benefits for the RU Builders, both in cost and in improved technology, are obvious.

- The system is more modular than a collection of switches with 500 inputs and 500 outputs. The topology and number of intermediate switches in such a system would have to change for the performance to increase. Instead, for the selected scheme, the procurement of the upper part of the Event Builder can occur in multiples of one-eighth of the system without any changes to the basic structure of the network. This fraction is small enough to ensure that all available resources at the time of procurement can be usefully deployed, but large enough to ensure that the system's complexity does not increase significantly.
- The system's scalability in multiples of the basic unit, the RU Builder, is built into the design. As long as the FED Builder systems can feed multiple Readout Units at the same rate as a single Readout Unit, i.e. as long as the performance of the FED Builders is linear with the number of Readout Units (1 to 8) actually connected to them, the overall system is obviously scalable as well.
- The system has an increased level of redundancy. An entire RU Builder may cease to operate during data-taking, yet the system can continue to run, albeit with lower (i.e. 7/8 of the full) performance. Furthermore, a RU Builder may be dedicated to testing a new version of the online software, or to commissioning some new element in the DAQ complex. The multiplicity and functional equivalence between the RU Builders enables this type of exercise in parallel with the normal operation of the system.

For all of the above, of the two main architectures displayed in Figure 3-2, the one with the explicit intermediate buffering stage has been selected. This buffering stage allows the possibility of a new interconnect, equivalent to adding a dimension to the propagation of the data, which in turns leads to the advantages listed above. The resulting system is shown, in its "three-dimensional" view, in Figure 3-6.

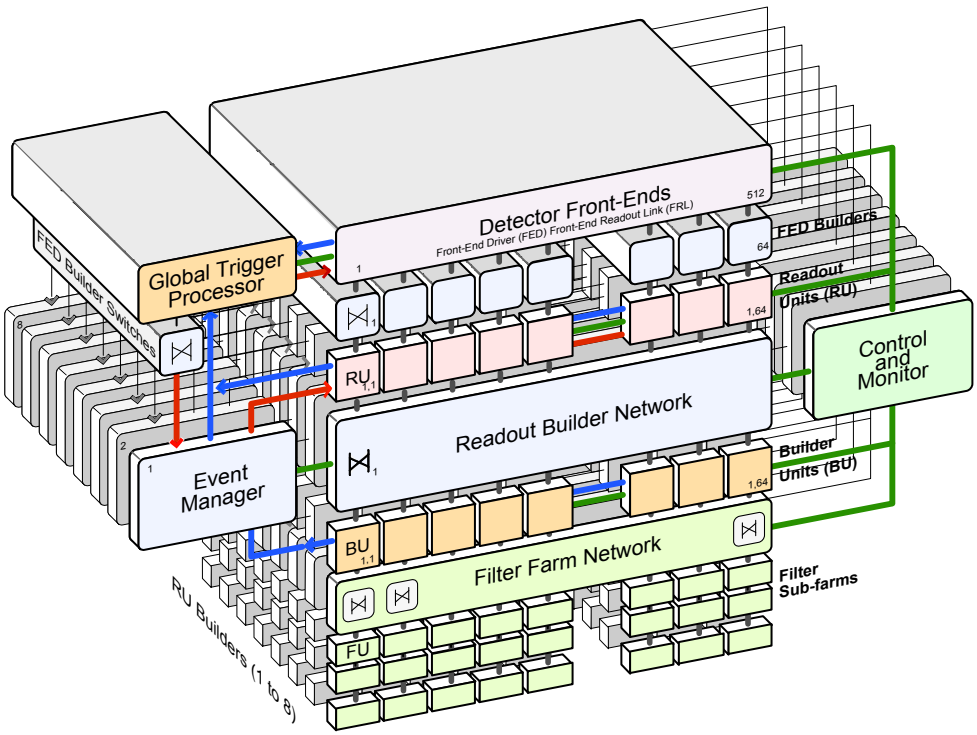


Figure 3-6 Three-dimensional view of the full Event Builder. The EVB corresponds to the same topology as that shown in Figure 3-5, but with the two stages now explicitly combined to make the connection with Figure 3-3.

In summary, the design described here satisfies all of the considerations listed in the Overview, as well as the resulting design principles that have guided the determination of an architecture for CMS. Perhaps above all, the design can accommodate change, be it of a technological or financial nature, or even a modified set of requirements.

3.2 Functional Decomposition of the DAQ

Apart from the Event Builder, the DAQ system has a number of other functions. Following the design principles of Chapter 1, "Overview", these functions are placed within subsystems and/or units, minimizing the dependencies between them. These remaining functions are:

1. the actual readout of the data from the Front-End Systems and into the DAQ FED Builder;
2. the actual selection, every second, of the best 100 events out of the 100,000 events that are read out of the CMS detector, along with the storage of these events;
3. the control and monitoring of all the DAQ elements, guaranteeing the integrity of the events and the trigger decisions.

These functions, along with the subsystems responsible for their execution and the flow of data and signals between them, are listed schematically in Figure 3-7.

3.2.1 Readout System

This function is provided by a large number (500) of Readout Columns which consist of a number of Front-End Drivers (FEDs) connected to a single Readout Unit, via a small 8×8 switch. The switch, as previously explained, is used to connect the FEDs to multiple Readout Units, thus potentially increasing the total throughput of this readout by a factor of up to eight. A subsystem not mentioned so far is the actual front-end electronics which read out the very basic information from each channel of the CMS subdetectors. These are referred to as Front-End Systems and are described, along with the FEDs, in Chapter 7, "Readout Column".

3.2.2 Event Selection

Events are selected by the Filter Farm, alternatively referred to as the processor farm, the HLT (High-Level Trigger) farm, or the HLT system. The DAQ relevant elements are the Builder Units providing the farm processors with complete events for analysis, and the processors themselves, the “Filter Units”. The collection of one Builder Unit connected to a number of Filter Units is referred to as a Filter Column and is the subject of Chapter 8, "Filter Column".

3.2.2.1 Computing Services

The Computing Services System is responsible for the final treatment of the events accepted by the High-Level Trigger system. This includes not only the forwarding of events to mass storage, but also the monitoring of the HLT system and of the CMS detector itself, and the identification of any malfunctioning elements. In addition, the CSS provides numerous displays, user-interface related functions such as an online event display, online physics display of selected physics channels, etc. The CSS is described in Chapter 16, "Computing Services".

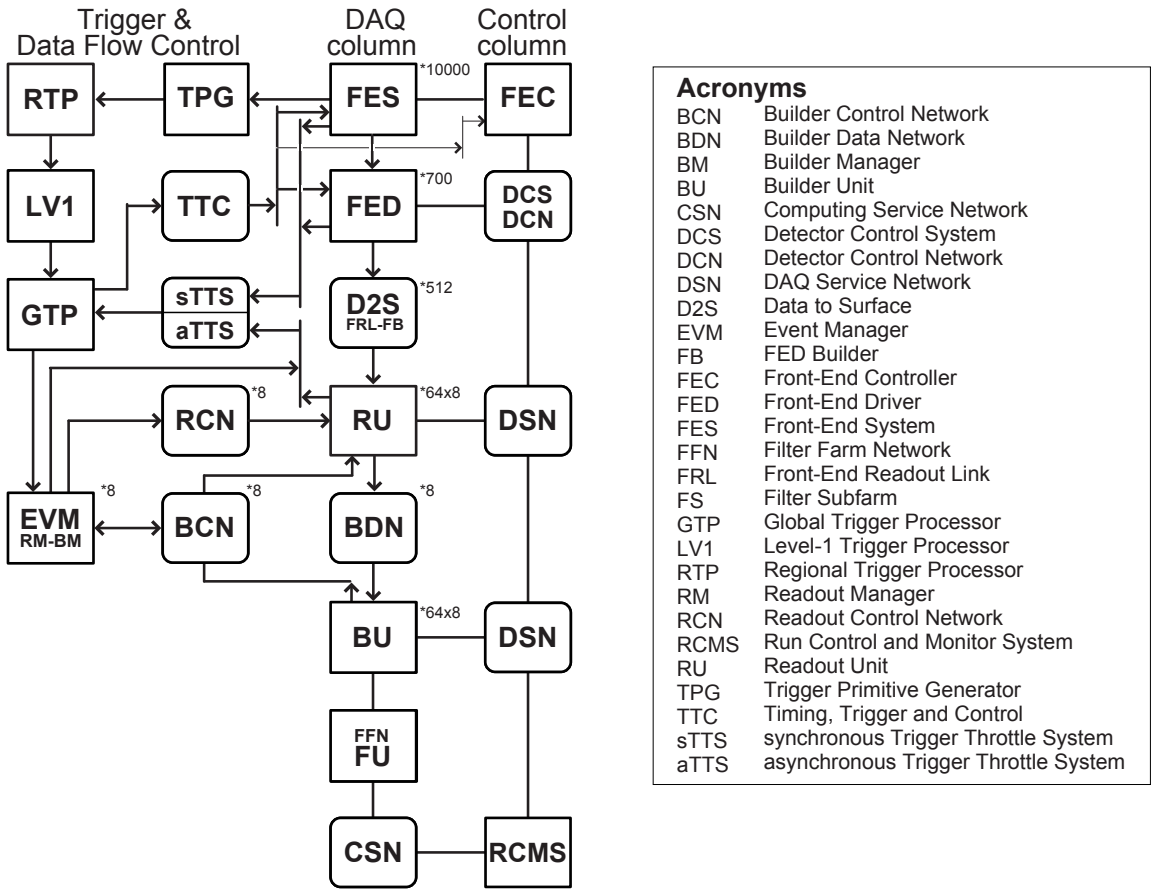


Figure 3-7 Schematic of the Functional Decomposition of the CMS DAQ. The multiplicity of each entity is not shown for clarity. As an example, there are 64 Builder Units for a single RU Builder.

3.2.3 System Control and Monitor

There are three levels at which the CMS Detector and the DAQ system are controlled. The lowest level is provided by a FEC, a front-end Controller charged with controlling and monitoring all aspects of the Front-ends. At the second level, the Detector Control System (DCS) provides the same functionality to the entire CMS detector. Finally, there is a Run Control and Monitor System (RCMS) providing the means to control and monitor all the elements in the DAQ system itself. There are two networks associated with these three levels of control and monitor, the Detector Control Network and DAQ Service, Network shown schematically as DCN and DSN in Figure . The DCN is used by the DCS to communicate with all the devices and subsystems it is responsible for, while the DSN is used by Run Control. The two networks do not need to be physically different, and, depending on the details of the final layout, may be implemented using the same hardware and software.

3.2.3.1 Run Control and Monitor System

The Run Control and Monitor System (RCMS), described in Chapter 12, "Run Control and Monitor System", provides the user interface to the CMS DAQ. Its functionality includes configuring all the elements in the system, monitoring their performance, displaying their status, identifying malfunctioning or un-

der-performing elements, and providing recovery mechanisms (for instance, resetting and restarting after a failure). The RCMS also interfaces to the Detector Control System (DCS).

3.2.3.2 Detector Control System

The Detector Control System, described in Chapter 13, "Detector Control System", is responsible for the traditional control and monitoring of all the detector services and other elements, from high and low voltage supplies to temperature sensors, as well as with the configuration of the front-end electronics. It is expected that during data-taking the DCS, which is nearly autonomous, will be driven by the RCMS.

3.3 Software Architecture

The architecture of the online-software infrastructure follows a layered middleware approach for distributed systems, designed according to an object-oriented model and implemented using the C++ language. Each computing node runs a copy of an executive program that exposes two sets of interfaces. The first lies between the middleware and the core plugins, components providing basic access to system services and communication hardware. The second lies between the middleware and the applications components and provides the access to the functions offered by the core plugins as shown in Figure 3-8. The middleware services include data and signal dispatching to applications, data transmission, exception handling facilities, access to configuration parameters, application and services location lookup (address resolution) and basic system services. The latter include locking, synchronization, task execution and memory management facilities. Applications communicate with each other through the services provided by the executive according to a peer-to-peer message-passing model. Each application can act both as a client and as a server. Messages are exchanged according to an event-driven processing scheme [3-1]. Messages are sent asynchronously and trigger the activation of user supplied procedures when they arrive at the receiver side.

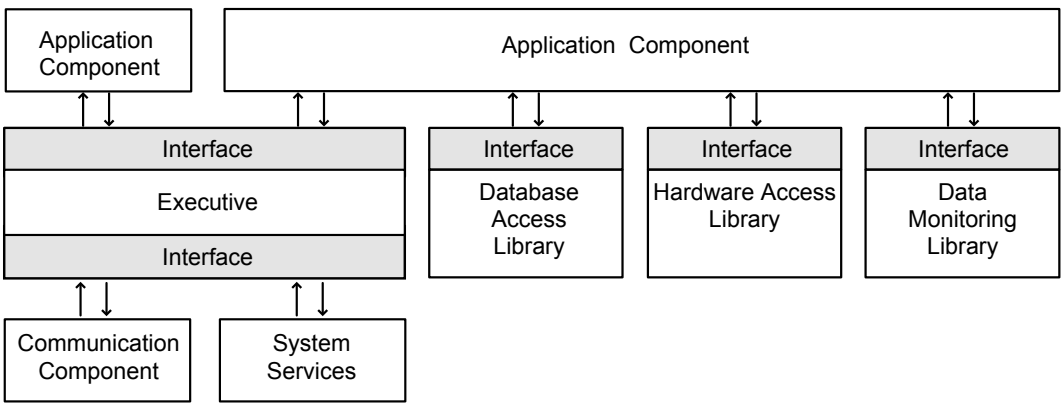


Figure 3-8 Architecture of the online software.

In addition to the software services that support distributed computing, the online software includes components to support local tasks such as hardware access, database access and local data monitoring. The Hardware Access Library (HAL) is used to directly manipulate hardware devices for configuration purposes. It provides access to registers and memory areas for various bus systems through named objects that can be defined dynamically.

The Database Access Library (DAL) exhibits an interface to read information from persistent data stores and presents it to the user through a standardized interface, the Document Object Model (DOM) [3-2]. DOM renders information as trees that can be traversed by the application code. Each node can contain a named data element or may be a link to other information that is accessible through one of the provided protocols. The library provides the required authentication and access regulation procedures for each supported persistent data store.

A library is used to perform basic statistical analysis for monitoring the applications' and system's operation on a local processing node. It allows the results to be rendered in a standard description, suitable for storage with database systems and for visualization using common display facilities supported by Web browsers.

The above software packages can be combined to retrieve parameters for a particular device from a database to configure, monitor and control the hardware at run-time.

The online-software is based on various industry standards for data-exchange to reduce development and maintenance efforts as well as to achieve interoperability with external systems. The selected technologies, through their wide adoption, increase the confidence to reach working solutions whether functions, services or whole subsystems are to be replaced. As an example of such a standard, the interface to external systems, such as run-control and monitoring facilities has adopted the Simple Object Access Protocol (SOAP)[3-3]. This message-passing protocol relies on the HTTP Web protocol [3-4] and encapsulates data using the eXtensible Markup Language (XML) [3-5]. More details are provided in Chapter 10, "Online Software".

3.4 High-Level Trigger

The algorithms used for the event selection, along with the details of the software and the detector simulation used to evaluate the performance of these algorithms, are described in Chapter 14, "Detector Simulation and Reconstruction" and in Chapter 15, "Physics Object Selection and HLT Performance".

As explained in the introduction, the CMS processor farm executes algorithms that combine the traditional "Level-2" and "Level-3" trigger steps. The wish to offer as much physics selection flexibility as possible, given the financial and technical constraints of the DAQ project, was one of the guiding principles, and one that results in the current architecture. The basic thesis of the CMS architecture is the creation of a system that can provide the following to the High-Level Trigger process:

- data which, except for the calibration constants used by the HLT, is essentially of "offline" quality;
- data originating from any part of the detector, without any restrictions from the physical connection of this detector element to some part of the readout or other electronics;
- full algorithmic flexibility, without any limitations arising from programming languages, instruction sets, or other technical limitations often encountered with special-purpose, in-house designs.

The first requirement is obviously satisfied, since the data given to the HLT processors are the full raw data contained in the front-end electronics, without any loss of information from further digitization and/or merging of the information. The second requirement is also clearly satisfied, since each processor is connected, by design, to all the detector elements of CMS, and can therefore access any data it deems valuable for the selection of any particular event. The third requirement is satisfied since the DAQ provides standard commercial processors as the platform on which the HLT algorithms are run.

As discussed already, while the above three features result in a HLT system which is only limited by the total computing resources that can be procured, a remaining issue is whether the system can provide enough throughput for the entire set of events accepted by the Level-1 Trigger to be forwarded to a given processor in the HLT farm. For the case of the 100 kHz scenario considered so far, the architecture described in this chapter, using 2Gb/s links can indeed provide this throughput. The demonstration of this fact is the subject of Part 2 “Data Flow” of this Technical Design Report.

3.5 Summary

The architecture of the CMS DAQ is motivated by the considerations put forth in Chapter 1, "Overview". In summary, it is conceived for a system that is expected to change with time, accelerator and experiment conditions. It is also conceived to provide the maximum possible flexibility to executing a full physics selection online. Finally, the two-stage switching design explicitly addresses the varying timescales of the experiment’s installation and of technological progress over the next few years. The remaining chapters of this Technical Design Report are aimed at providing a full blueprint of those elements of the DAQ that must be present in the middle of 2004, and a full design and proof of principle for those elements that only have to be present shortly before the actual start of data taking, currently planned for 2007.

3.6 References

3-1 B. N. Bershad et al., “Extensibility, Safety and Performance in the SPIN Operating System”, in Proceedings of the Fifteenth ACM Symposium on Operating System Principles, pp. 267-284, 1995.

3-2 A. Le Hors et al.,” Document Object Model (DOM) Level 3 Core Specification, Version 1.0, W3C Working Draft 13, September 2001”, http://www.w3.org/TR/2001/WD-DOM-Level-3-Core-20010913

3-3 D. Box et al.,” Simple Object Access Protocol (SOAP) 1.1, W3C Note 08 May 2000”, http://www.w3.org/TR/SOAP

3-4 R. Fielding et al., “Hypertext Transfer Protocol - HTTP/1.1”, IETF RFC 2616, June 1999. http://www.w3.org/Protocols/rfc2616/rfc2616.html

3-5 J. Boyer, “Canonical XML Version 1.0, W3C Recommendation 15 March 2001”, http://www.w3.org/TR/xml-c14n

Part 2
Data Flow

4 Event Builder

One of the main tasks of the Data Acquisition (DAQ) system is to read the data corresponding to each event selected by the Level-1 Trigger out of the detector Front-End Drivers, to concatenate it into a single data structure, the “physics event”, and to then forward this event to a filter farm for the final decision on whether the event should be discarded or directed to permanent storage. The complex that provides these functions, i.e. the detector readout, the merging of the data from the physically and logically distinct data sources, and the forwarding of complete events to a set of processors is referred to as the “Event Builder”.

This chapter provides a functional description of the Event Builder of the DAQ. The emphasis is on the architecture, the overall functionality and the interfaces between the individual components of the Event Builder. The protocols used, as well as the control of the data flow are the subject of Chapter 5, "Event Flow Control". The design of the networks involved in the Event Builder is the subject of Chapter 6, "Event Builder Networks". Finally, the elements comprising the full readout and filtering chain of the DAQ are described in Chapter 7, "Readout Column" and Chapter 8, "Filter Column".

4.1 Introduction

The three main requirements on the Event Building process in the DAQ are summarised in Table 4-1. These are “system-level” parameters, in the sense that they are essentially independent of any design choices. Thus, the maximum Level-1 Trigger rate of 100,000 Hz as well as the $O(10^2)$ Hz are dictated by the physics program of the experiment as well as the expected offline computing resources that will be made available. Similarly, the average event size is an “external” parameter that is estimated based on the expected occupancy of the CMS detector channels.

Table 4-1 Requirements of the CMS Event Builder.

Parameter	Value
Maximum Level-1 Trigger rate	100,000 Hz
Maximum rate after High-Level Trigger	$O(10^2)$ Hz
Average event size	1 MB

The Event Builder is implemented as a two-stage process as outlined in Chapter 3, "DAQ Architecture". In the first stage, the data for each event in the Front-End Drivers (FEDs) are merged into Readout Units (RUs). The RUs read the data from 8 FEDs, and perform a first concatenation of the event data fragments into larger data blocks, referred to as “super-fragments”, or “s-fragments”. This stage is implemented by 64 “FED Builders”, where a FED Builder comprises 8 FEDs, a 8×8 switch and 8 RUs. Data from different events are sent, in turn, to different RUs, thus maximizing the bandwidth of the readout. At the end of this first stage, the data from a single event are contained in 64 RUs. These RUs are in turn read out by Builder Units (BUs) which combine the 64 super-fragments into a single event. The collection of 64 Readout Units, the 64×64 switch that connects them to the Builder Units and the actual 64 Builder Units is referred to as a “RU Builder”.

The full DAQ system comprises 64 FED Builders and 8 RU Builders as listed in Table 4-2 and displayed in Figure 4-1. The rest of this chapter describes the functionality of these two Builders.

Table 4-2 Event Builder parameters.

Parameter	Value	Derived from	Scope
Number of inputs to the Event Builder	512	Assumed link speed: 2Gb/s	System
Number of Readout Unit Builders	8	Assumed switch size: 64×64	Full Event Builder
Number of FED Builders	64	Number of RU Builders	
Inputs (FEDs) per FED Builder	8 – 16	Desired mean size at output	FED Builder
Average FED data fragment size	0.1 – 3 kB	Detector readout	
Outputs (RUIs) per FED Builder	1 – 8	Number of RU Builders installed	
FED Builder output: s-fragment size	16 kB	Balancing of data sizes across RUs	
Inputs (RUs) per RU Builder	64	Assumed switch size	RU Builder
Average s-fragment size	16 kB	Balancing of inputs: 1MB/64	
Outputs (BUs) per RU Builder	64	Number of inputs	
Maximum event rate in RU Builder	12.5 kHz	Number of inputs in RU Builder, link speed	

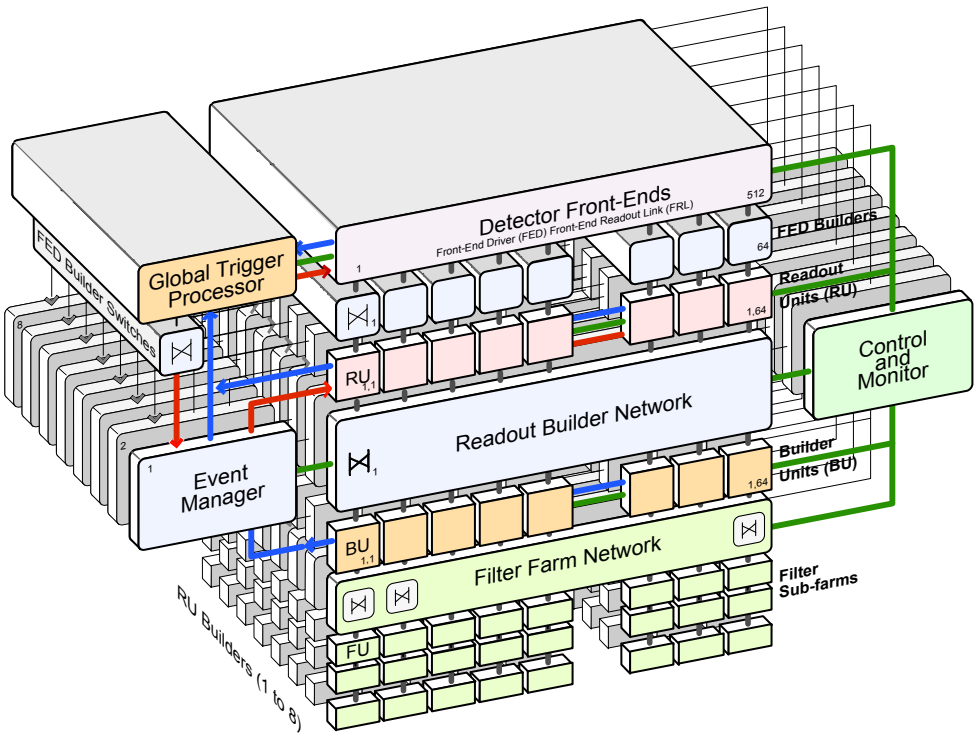


Figure 4-1 Conceptual Design of the Event Builder. This is a “three-dimensional” view of the system. On the front-side is one Readout Unit (RU) Builder. On the left and down the side is a schematic of the multiple RU Builders and the way they are connected to the Front-ends via a small switch (8×8) which routes each event to the appropriate RU Builder. The interface to the Front-ends, the small switch and the input to the RU Builder constitute the Front-End Driver (FED) Builder.

4.2 FED Builder

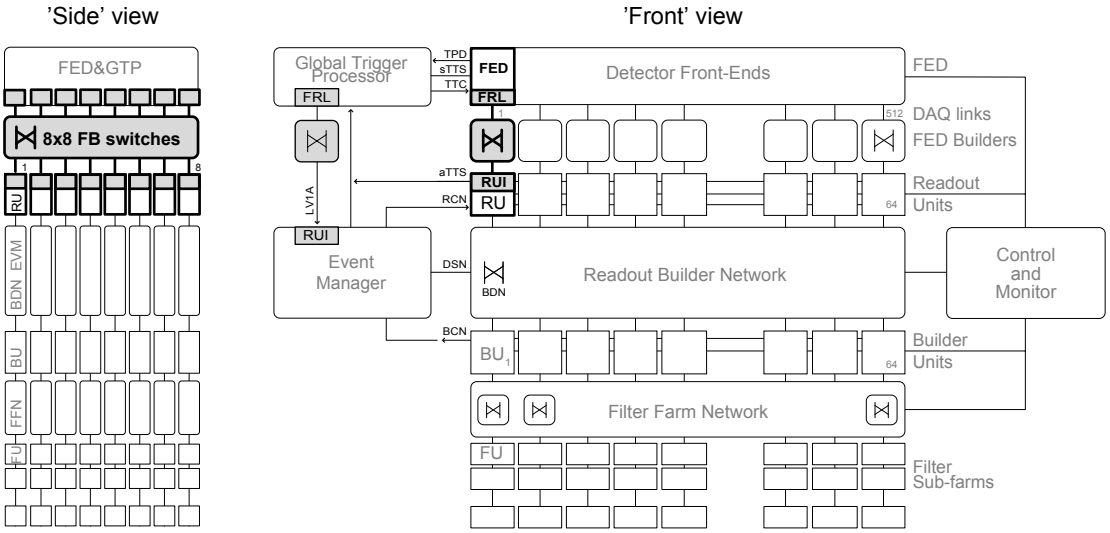


Figure 4-2 FED Builder elements from FED to RUI plus the special FED Builder connecting the Global Trigger Processor to the Event manager.

The main functions of the FED Builder are (see Figure 4-2):

1. transport the fragments from the underground area to the surface,
2. assemble event fragments of variable size from some ~700 FEDs into 64 super-fragments (s-fragments) of roughly equal size (16kB each),
3. multiplex these s-fragments on an event by event basis between N ($1 \leq N \leq 8$) Readout Unit Builders.

The implementation of the second and third functions necessitates the connection of a number of FEDs (typically 8) to a number of RUs. The full system has 8 RUs, but in general, only one per RU Builder need be present in the system. The connection is achieved by switches with typically 16 ports, 8 for input and 8 for output. Each input port obtains data, corresponding to a single event, from one or two FEDs. Each output port is connected to a Readout Unit which concatenates the 8 fragments into a super-fragment. When the system is equipped with more than one RU Builder, successive events are multiplexed, via the switch, to different Readout Units and hence different RU Builders. The full system has 64 FED Builders, which connect the FEDs to the RUs, plus an additional FED Builder, which connects the Global Trigger Processor to the Event Manager of each RU Builder.

Each FED Builder consists of an interface to the FEDs, a Fed Builder Input buffer (FBI), a switch and an output buffer. The interface to the FEDs is called the "Front-end Readout Link" (FRL) and the output buffer is called the "Readout Unit Input" (RUI) because it also serves as input to the Readout Unit.

To maintain the data rate of the CMS DAQ system, each FED Builder input must sustain the data from a single FED at a rate of up to 100 kHz, i.e. 2 kB·100 kHz = 200 MB/s. Clearly, the switch output ports must be capable of sustaining this data rate as well, and therefore the basic FED Builder switch consists of 16 ports each one providing a sustained 200 MB/s.

4.2.1 Front-end Readout Link (FRL)

With 64 switches and 8 input ports per switch, there are only 512 input ports available to connect more than 600 FEDs. The FRL, whose primary function is to provide the link between the FEDs and the FED Builder Input buffer (FBI) can optionally merge data from two FEDs so that the maximum number of FEDs that can be read out is 1024

The FRL also checks the event fragment size and discards event fragments whose size exceeds a predefined maximum. This is necessary in order to protect the FED Builder from problematic events.

The FRL constitutes the interface between the FEDs and the FED Builder and is not considered to be part of the FED Builder.

4.2.2 FED Builder Input Buffer (FBI)

The FBI sends the data received from the FRL to the FED Builder switch and must therefore select the correct output port for each event. The algorithm used to determine the destination for each event may vary from a simple round-robin across all outputs, implemented as modulo 8 of the trigger number, to more general cases designed to make efficient use of the DAQ when the RU Builders are not identical in performance. If the RU Builders are installed in stages, it is possible and likely that the newer ones can operate at higher performance than those installed in the beginning. The algorithm for selecting a destination can be used to send more events to the faster RU Builders. The algorithm can also be used to redirect data traffic away from a RU Builder that has developed a fault.

4.2.3 FED Builder Switch

The FED Builder switch provides the inter-connectivity between the FBIs and the RUIs in the CMS Readout. In its simplest configuration, it is a 16-port switch capable of sustaining 200 MB/s per port. This 8×8 switch[1] has eight ports connected to FBIs and eight ports connected to RUIs. In reality, the actual number of FBIs connected to a single FED Builder will not be fixed to eight, but will be determined by the average size of the data fragments in the corresponding FEDs. In particular, the FED event fragment size, as discussed in Section 7.2, "Detector Readout Requirements", varies significantly across FEDs, with some FEDs providing an average fragment size as low as 100 B. Each FED Builder switch will connect to FEDs with both small and large event fragment sizes selected such that the sum of the fragment sizes in each switch is approximately equal across the RUs.

In addition to differences in the average amount of data from different FEDs, the actual amount of data from a single FED fluctuates from event to event. As explained in Section 6.2.3, "Traffic-shaping", this variability of the data size leads to a lower utilization efficiency of the switch bandwidth. For this reason, one must either apply some traffic-shaping policy or provide switches with port capacity which is sufficiently larger than the desired throughput. To avoid complicated traffic patterns at this early stage of the DAQ, the FED switches are required to have ports running at twice the desired average of 200 MB/s, i.e. the FED Builder switch should have 400 MB/s per port.

A final requirement on the FED Builder switch is that the overall network (FRL-Switch-RU) it belongs to should provide back pressure so that data are not lost in case of congestion in the system downstream.

1. In general, it is an "$n \times m$" switch where $1 \le n \le 16$ and $1 \le m \le 16$. The final values of n, m are an implementation issue. At present, the most likely option, used throughout this TDR in order to simplify the description, is the case $n=m=8$.

4.2.4 The Readout Unit Input (RUI)

The output of the FED Builder serves as input to the Readout Unit and is thus given the name “Readout Unit Input” (RUI). As mentioned above, the basic function of the RUI is to build super-fragments from the collection of event fragments from the FEDs and to transmit these s-fragments to the RU. This is described in more detail in “4.3.7 RU Builder Data Flow”. The RUI consists of a processor with sufficient memory to build and store super-fragments. It has high bandwidth data connections to both the FED Builder switch and the Readout Unit.

4.2.5 Data Flow in the FED Builder

The protocol used by the FED to send event fragments to the FRL is made as simple as possible: upon receipt of a Level-1 Trigger accept, the FED encapsulates the event data corresponding to the bunch crossing in question in a common CMS data format[1] and stores it in its FIFO output buffer. The encapsulation header includes the event number which is used by the FBI to route the fragment through the FED switch. Event data flow from the output buffer of the FED through the FRL to the FBI, whenever data are present in the FIFO and the FBI is ready to receive. The protocol used includes control words that are used by the FRL to recognize the start and end of each event fragment. The data flow includes backpressure so data will stop flowing if the FRL and FBI buffers fill up. Note that this will in turn cause the FIFO in the FED to fill up. To prevent new triggers while the DAQ has no room for more data, the FED will issue a fast trigger-throttling signal when the FIFO of the FED is filled beyond a pre-defined high level mark.

In the simplest and most common case the FRL receives input from only one FED and passes it through to the FBI. The FBI then extracts the event number from the FED data and determines, based on this number[2], the destination RUI (i.e. the switch output port number). It then transmits the fragment to that output port. The algorithm used for calculating the output port number is identical in all the 512 FBIs. Output ports corresponding to the same event number in all 64 FED Builder switches belong to one RU Builder. In the more complicated case where the FRL receives data from more than one FED, these fragments are merged in the FRL before they are passed through to the FBI.

4.2.5.1 “Super-fragment” Building

For a given trigger the data fragments from the 8 input ports of a FED Builder switch arrive at the output port (RUI) in an arbitrary order, depending on the switch load, the timing of the start of sending and the sizes of the FRL fragments. The RUI has a processor with memory and the fragments are stored in this memory in the order in which they arrive. Once all fragments have arrived they are assembled into a “super-fragment” (s-fragment). A header containing the event number and the s-fragment size is then added to each s-fragment. The s-fragments are transferred to the memory of the RU in order of increasing event number as soon as they have been assembled. This is done in order to free space for new event fragments in the RUI which has limited memory. In this scheme, it is the RUI that is responsible for the transfer of the s-fragments to the RU in an ascending event number order.

The FED Builder switch supports backpressure. This in turn causes backpressure from the FBI to the FRL to the FEDs and data stops flowing. This can happen even if only one output port, i.e. any one of the 8 Re-

1. See Section 7.3.3, "FED Data Format".
2. As previously mentioned, the algorithm used to select the destination RUI can be anything from a simple round-robin to a more sophisticated algorithm that allows for un-equal data distributions among the switch outputs. This latter case can be implemented using, e.g. “destination tables” prefabricated by the Run Control system.

adout Units, blocks the data flow. If one RU Builder develops problems that cause it to block the data flow frequently or permanently, then the system is reconfigured to exclude this RU Builder from the list of outputs of all FED Builders and data taking is then resumed with the reduced capacity provided by the remaining RU Builders

The fragment size from a given FED varies from event to event. These statistical fluctuations can be absorbed by the FED Builder within certain limits imposed by the link speed and the amount of memory available for s-fragment building. A truncated version of the fragment, flagged as an error is transmitted when a fragment must be discarded because it is too big.

4.2.6 FED Builder for the Global Trigger Processor

A few entities from the global trigger, such as event number, bunch crossing number and trigger pattern, must be read by the Event Manager at each Level-1 Trigger, very much in the same way that event fragments are read from the FEDs at each trigger. This data must be routed to the Event Manager in the same RU Builder as the FED data for the event and must therefore also go through a FED Builder.

The FED Builder for the Global Trigger is special in the sense that it does no merging of data. Only one input port, connected to the Global Trigger, is used. There are up to 8 output ports, one for each RU Builder in the DAQ system. Each output port is connected to the corresponding Event Manager.

The Global Trigger also provides trigger data in three additional data streams (GT data, CalTrig Data, MuTrig Data). These data are considered to part of the event data and are thus read like data from any other FED.

4.3 RU Builder

The second stage of the Event Building process is carried out by up to eight RU Builders (see Figure 4-3). The function of the RU Builder is to assemble the event super-fragments from the 64 FED Builders into full events ready for processing in the filter farm. The main functional elements of the RU Builder are:

- 64 Readout Units (RUs) with large buffers that serve to de-couple the flow of data out of the FEDs from the flow of data in to the RU Builder.
- 64 Builder Units (BUs) where full events are assembled and then sent off to the filter farm
- 1 Event Manager, whose primary function is to adapt the traffic in the RU Builder to the throughput capacity of the BUs and their associated filter farms.
- 3 logical networks connecting the storage and processing elements
    a. Builder Control Network (BCN). Connects all units in the system and whose main function is to enable the flow of control information.
    b. Builder Data Network (BDN). The high-throughput network used for the transfer of the event data. It connects all the RUs to all the BUs.
    c. Readout Control Network (RCN).Connects the Event Manager to all the RUs. It serves to distribute common Readout information to all the RUs

Each of these elements are briefly described in the following sections. The detailed description of the data flow in Section 4.3.7, "RU Builder Data Flow" should help explain the interconnection between these elements.

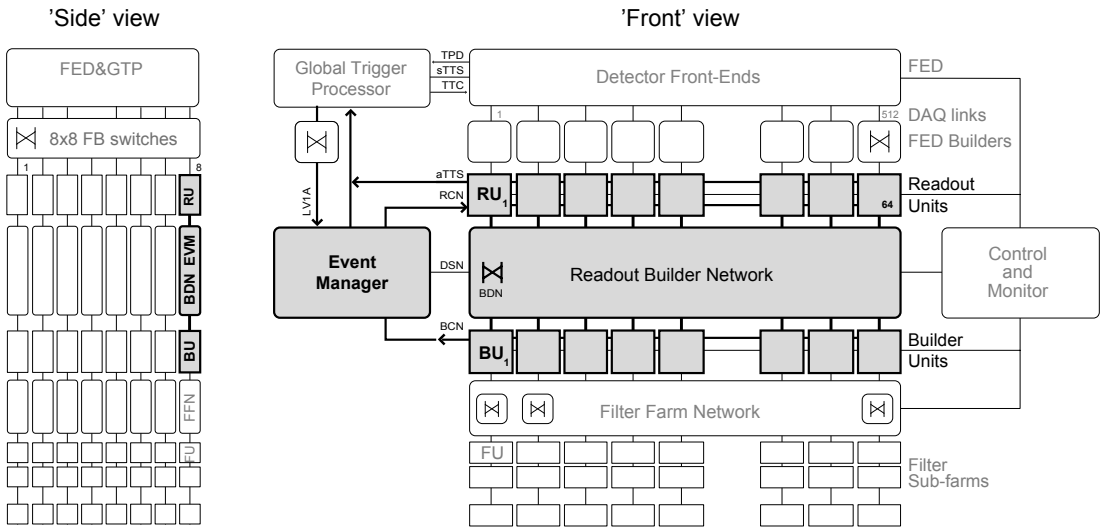


Figure 4-3 Event builder with the RU Builder highlighted.

4.3.1 Requirements

Table 4-2 shows how the global event building parameters, along with a basic assumption for the size of the switch with 2Gb/s links that will be available, translate into requirements for the RU Builder. These requirements need not be met until a few years after the start of LHC operations since it is expected that at the beginning of data taking in 2007, the accelerator will operate at a reduced luminosity with potentially some detector elements to be installed at a later time. In 2007 the average data rate is anticipated to be approximately a factor 3-5 smaller than the design requirement of 100 GB/s (100 kHz·1 MB).

Since the final system consists of up to eight RU Builders, each RU Builder must sustain an event rate of up to 12.5 kHz. This corresponds to a data throughput of 12.5 kHz · 16 kB = 200 MB/s for each RU. At the start of data taking in 2007 when only a few RU Builders will be needed, the maximum throughput per RU will still be 200 MB/s.

4.3.2 Readout Unit (RU)

The RU performs three functions: it reads super-fragments (s-fragments) from the RUI, buffers them in its memory until a request for this data is made and finally transmits the data to the Builder Units (BU) upon request. In Figure 4-4 the RUs are shown in the context of the RU Builder.

The primary function of the RU is to provide sufficient buffer space to store s-fragments until the full event data corresponding to each event are assembled in the corresponding Builder Unit. A secondary function is to allow for an efficient use of the Builder Data Network via the implementation of traffic shaping and optimal data organization in memory. A final function of the RU is to check the correctness of the data concentration performed by the FED Builder.

Up to the input buffer of the RUs all buffering in the readout is carried out in hardware or software FIFOs which contain, in the header information of each fragment, the bunch crossing number and the event number. After this point, random access is required because the BUs operate in parallel. The RU must

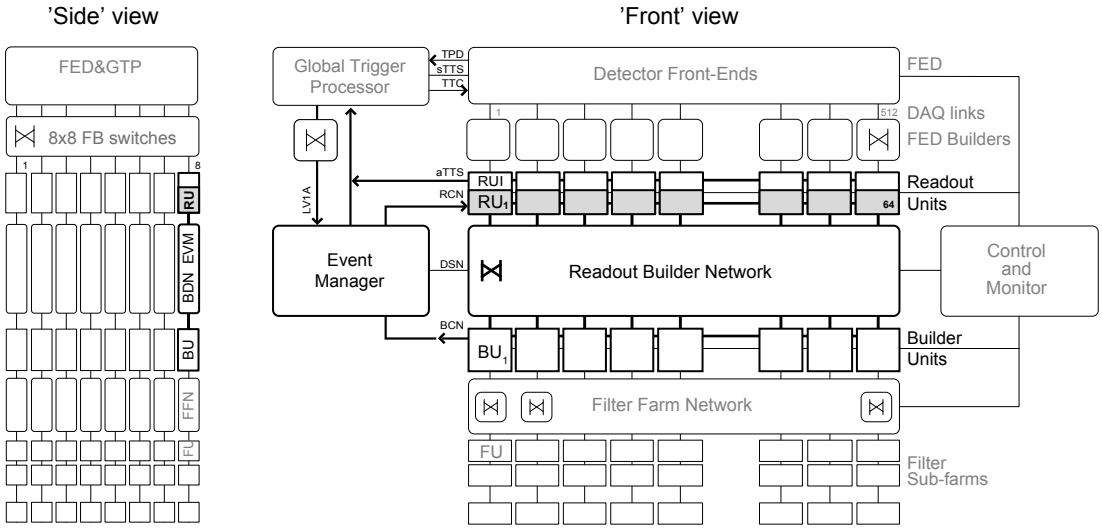


Figure 4-4 RU Builder with RU highlighted.

therefore store event s-fragments in memory in a way that permits efficient random access by the BUs. The RU must also sustain a s-fragment rate of 12.5 kHz, reading s-fragments of 16 kB average size. This translates into a data input rate of 200 MB/s (12.5 kHz · 16 kB).

After storage in the RU's memory, the event s-fragment is assigned a temporary Event Identifier (Event_ID), which is the label used for all further references to this event throughout the Readout Builder.

4.3.2.1 Event_ID

This is a logical entity introduced to facilitate efficient random access to event s-fragments in the RU's memory. It is a unique label assigned to each event after the s-fragments are stored in the RU. The label remains valid as long as the event is being processed by the event builder (and filter farm). The number of Event_IDs is limited and reflects the number of events that can be processed concurrently in the event builder. When an Event_ID becomes free it can be reused for a new event. The use of the Event_ID is described in Section 4.3.7, "RU Builder Data Flow".

4.3.3 Builder Unit (BU)

The primary event building function of the BU is to build full events from the event s-fragments residing in the 64 RUs. A secondary function is to buffer these events until they are processed by the filter farm. The functionality of the BU with respect to the filter farm is described in Chapter 8.3, "Builder Unit (BU)". This chapter does not deal with the filter farm. It just assumes that the events reside in the BU until they have been released by the filter farm. In Figure 4-5 the BUs are shown in the context of the RU Builder.

Each BU requests event fragments from all the RUs to form one full event. All BUs are active at the same time, each reading event fragments from the RUs in order to keep the switching fabric operating at full capacity. Each BU is actually building multiple events concurrently in order to be able to overlap data input with processing. This is described in more detail in Section 4.3.7, "RU Builder Data Flow".

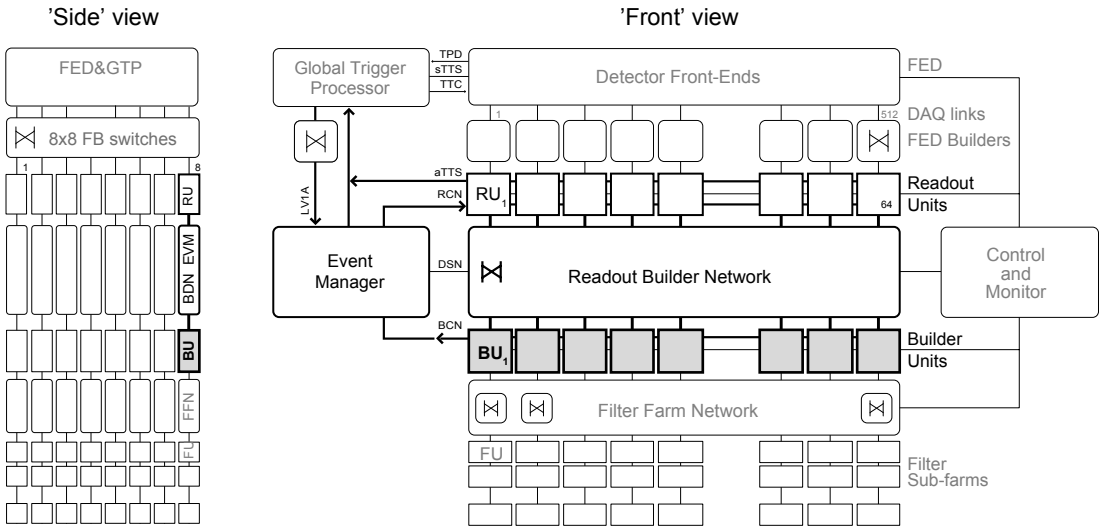


Figure 4-5 RU Builder with BU highlighted.

The BU must be able to receive s-fragments of average size of 16 kB from 64 RUs at a rate of 12,500/64 = 200 Hz. With an average event size of 1 MB this translates into an input rate of 200 MB/s.

4.3.4 Event Manager (EVM)

The Event Manager (EVM) is the entity, present in each RU Builder, which determines the flow of data in the RU Builder. Its main function is the assignment of Event_IDs to the events entering the RU Builder, as well as the management of these Event_IDs for the duration of the event's presence in the RU Builder. The EVM is thus the single entity in the RU Builder that contains the status of all the events in the RU Builder at any point in time. It is thus the natural place to accumulate system-level statistics related to the rate of input events, rate of absorption of events by the filter farm, etc. The EVM communicates with all RUs and BUs in the RU Builder. As mentioned previously (Section 4.2.6, "FED Builder for the Global Trigger Processor"), the EVM communicates with the Global Trigger via a standard RUI which serves as the interface to the FED Builder of the Global Trigger Processor. In Figure 4-6 the EVM is shown in the context of the RU Builder.

The EVM consists of two tightly coupled functional units: the Readout Manager (RM) and the Builder Manager (BM). The Readout manager distributes event numbers with an associated Event_ID to all the RUs, while the Builder manager communicates with all the BU's, distributing valid Event_ID's on request from the BUs. The two parts share access to a common pool of Event_ID's.

4.3.5 Builder Network

The Builder Network covers the two logical networks namely the Builder Data Network (BDN) and the Builder Control Network (BCN). It is a 64×64 port switching fabric connecting the 64 RUs to the 64 BUs in a RU Builder. The switch must be capable of 200 MB/s throughput on each port. Event building is performed over this network by sending s-fragments, corresponding to a single event, from all 64 RUs to a single BU. High performance is achieved by having all 64 BUs collect event fragments from multiple events concurrently. Inefficiencies may occur in the switching fabric because of congestion at either the output ports or in the switch fabric. However with well balanced input and careful traffic shaping it is pos-

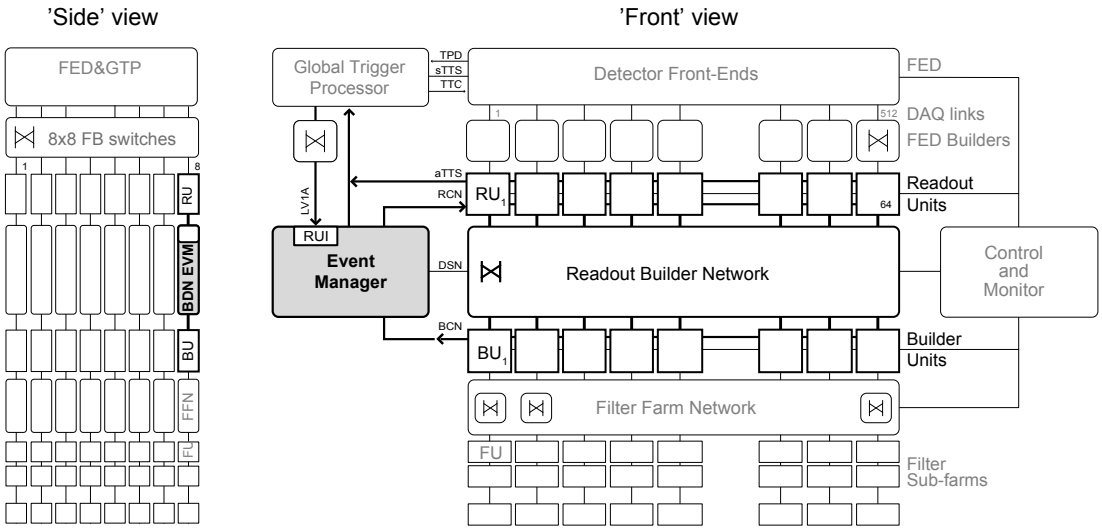


Figure 4-6 RU Builder with EVM highlighted.

sible to operate the switch at almost 100% efficiency. This is discussed extensively in Chapter 6, "Event Builder Networks". Assuming that such a high utilization of the switch is possible, the switch utilized need therefore have a capacity of 200 MB/s per port.

4.3.6 Readout Control Network (RCN)

This is the logical network that connects the EVM to all the RUs in a RU Builder. It may be implemented as an independent network or it may use the same switching fabric as the Builder Network. For each trigger the EVM must distribute the Event_ID-event number association to all the RUs. This is not a high volume of traffic (16 B · 12.500 Hz = 200 kB/s).

4.3.7 RU Builder Data Flow

The association of Event_IDs with event numbers is performed by the EVM, which manages a pool of Event_IDs and associates free ones with event- and bunch crossing numbers, read from the Global Trigger at each Level-1 Trigger.

The EVM obtains the trigger information including the event and bunch crossing number from the Global Trigger at each Level-1 Trigger directed to the RU Builder in question. The EVM then assigns an Event_ID to each of these triggers and sends this assignment to all RUs in the RU Builder via the “Readout Control Network” (RCN). Several assignments may be packed into one physical message over the RCN in order to reduce transmission overheads.

The super-fragments are initially stored in order of arrival in the RU memory, which is also in increasing event number order, because that is how they are delivered by the RUI. The RU will change this to store the event fragments indexed by Event_ID as soon as the Event_ID assignments arrive.

The RU buffer memory imposes an upper limit on the maximum latency of the RCN. However, with an average data rate into the RU of 200 MB/s and a RU buffer size of O(100) MB, the latency requirement is not critical. See Chapter 6.6, "Readout Control Network" for a quantitative description.

The BU must have access to a list of Event_IDs corresponding to events that are buffered in the RUs awaiting event building. The BU obtains this information from the Event Manager over the “Builder Control Network” (BCN). This ensures that each BU gets allocated a different set of Event_IDs, so that the same event is not built by two different BUs. The BU can then use an Event_ID to request all the RUs to send the super-fragment corresponding to this Event_ID. For efficiency reasons the request is not made for a single Event_ID, but for several Event_IDs that were already allocated to the BU.

All the RUs then start to send s-fragments corresponding to the requested Event_IDs to the BU requesting them. This can potentially lead to inefficiencies in the BDN network because of congestion at the BU port since this port is now the target of numerous RUs. Traffic shaping can minimise this congestion and achieve a very high utilization efficiency of the BDN, provided the input traffic is well balanced. The details of this are discussed in Section 6.2, "Event Building with Switching Networks".

The procedure outlined above is followed by all the BUs concurrently, so the BDN has data flowing from all input ports (RUs) to all output ports (BUs) concurrently.

When a BU (and the associated filter farm) has finished processing an event it requests the EVM to purge the corresponding Event_ID from all EVB components. The EVM is then free to reuse that Event_ID for a new event accepted by the Level-1 Trigger. The EVM throttles the Level-1 Trigger if the number of free Event_IDs falls below some number deemed necessary for the operation of the system in the following second or so.

The EVM communicates with each BU over the “Builder Control Network” (BCN) in a peer-to-peer fashion. Each BU requests a group of Event_IDs and gets a list of Event_IDs in return. The communication between the EVM and the RUs over the RCN may be one-to-N, or peer-to-peer, depending on switch technology.

The BU has enough memory to concurrently build several events from all RUs plus sufficient memory to buffer events that are currently in use in the filter farm. Whenever buffer space is available in the BU for more events, the BU requests new Event_IDs from the Event Manager. As already mentioned, this is not done on an event by event basis, but when a given fraction of the available buffer space has been released. This grouping of requests reduces the load on the Event Manager, which must communicate with all BUs individually.

4.4 Partitioning

In normal physics data acquisition mode the event builder works as described so far in this chapter. However other modes must also be possible in order to cover the following cases

- Some detectors need to take data independently of the others for the purpose of
    - Calibration
    - Detector or trigger commissioning
    - DAQ debugging
- All detectors participate in normal physics data taking except for one which may perform any of the tasks listed above.

The division of the DAQ into smaller subsets capable of operating (more or less) independently of each other is called partitioning. Partitions used for calibration or commissioning have less stringent event rate requirements than physics data taking so efficient event building is not a requirement.

The CMS trigger system must also be partitioned to provide independent triggers to each of the DAQ partitions. The partitioning of the CMS trigger is described in reference [4-1]**.** The partitioning of the trigger imposes constraints on the partitioning of the DAQ system as well and for this reason it is described briefly in the following section.

4.4.1 Trigger Partitioning

The CMS trigger system is organised in 32 partitions, with each partition corresponding to a major component of a subdetector to which triggers are distributed and from which Trigger-Throttling System (TTS) signals are collected. These partitions are referred to as Trigger Timing and Control (TTC) partitions in this document. The TTC partitions can be grouped in up to 8 TTC partition groups, which can operate independently.

The configuration and operation of these TTC partition groups is carried out under the control of the Global Trigger Processor. Each TTC partition group operates with its own trigger conditions, receiving triggers that correspond to these conditions only. The triggers are distributed to just one TTC partition group at a time.

TTC partitions from different subdetectors may be combined in one TTC partition group. All TTC partitions are integrated into a single TTC partition group when the DAQ system is running in normal physics conditions.

The Global Trigger Processor receives triggers from all the TTC partition groups and sends them to the Event Managers in the RU Builders. The DAQ must be configured to have one DAQ partition corresponding to each TTC partition group and the DAQ design must allow independent operation of each DAQ partition.

4.4.2 DAQ Partitioning

The DAQ system is designed to have one DAQ partition for each TTC partition group of the Global Trigger. Each DAQ partition can operate independently of the other DAQ partitions in terms of data flow and run control. That is to say each DAQ partition can start and stop runs without forcing the other DAQ partitions to start and stop at the same time and a data flow problem in one DAQ partition will not prevent data from flowing in the other DAQ partitions.

The required independence of the DAQ partitions imposes some limitations on their granularity.

- A RU can not belong to more than one partition
- It is preferable that each partition has its own EVM

The DAQ partitioning can be implemented either at the FED Builder level, assigning RU Builders to be partitions, or at the RU Builder level dividing each RU Builder into partitions. Both methods are described in the following sections.

4.4.2.1 FED Builder Partitioning

The distribution of events to RU Builders is performed by programmable logic in the FED Builder as describe in Section 4.2, "FED Builder". Events are normally distributed to different RU Builders as determined by the event number modulo the number of RU Builders active in the system. However, a group of

FEDs belonging to a TTC partition group can be allocated a RU Builder dedicated to its readout. This is achieved by programming all the FED Builders belonging to these FEDs such that all events are sent to this single RU Builder (see Figure 4-7). All other FED Builders must be programmed not to use the RU Builder dedicated to that TTC partition group. This scheme allows the data flow from one TTC partition group to operate independently with a dedicated RU Builder, while all other sub detectors participate in normal physics DAQ mode utilizing the remaining RU Builders.

The EVM of the RU Builder belonging to the DAQ partition is dedicated to this partition. It operates like any other EVM, except that it must deal with only the subset of the RUs that belong to the DAQ partition.

The scheme can be extended to as many TTC partition groups as there are RU Builders with, at the limit, each TTC partition group operating independently. This mode of operation is well suited to commissioning of detectors and their DAQ systems in periods without beam in the accelerator.

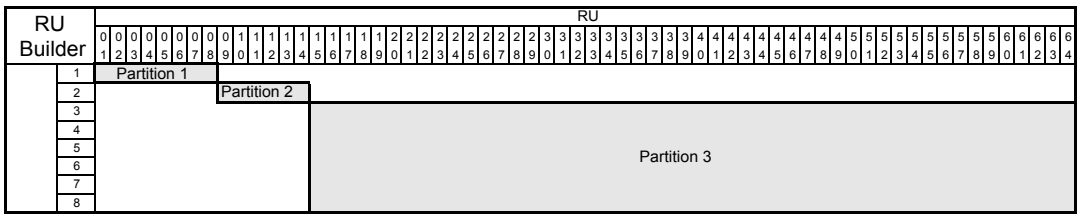


Figure 4-7 Example of partitioning in the FED Builder. Partition 1, using RUs 1 to 8 is allocated to RU Builder 1 and Partition 2, using RUs 9 to 14 is allocated to RU Builder 2. Partition 3 uses the remaining RUs and is allocated RU Builders 3 to 8.

The maximum number of possible DAQ partitions is limited by the number of RU Builders installed and the maximum number of TTC partition groups being 8. In the final implementation, the number of RU Builders installed will be 8, however the initial number will be dictated by the performance of the accelerator and is thus expected to be less than 8.

The FED Builder for the Global Trigger, which transfers the “trigger summary block” containing event number and trigger type must be programmed to direct the trigger summary block to the RU Builders that correspond to the TTC partition group which generated the trigger.

The only trigger data that all partitions can see is limited to the trigger summary block transmitted to the EVM.

Restrictions

- The event builder can not be used to its full capacity because each partition can not use all the RUs connected to the FED Builders. Only the RUs in the allocated RU Builder(s) are used, the rest are idle.
- The number of DAQ partitions is limited by the number of RU Builders.

4.4.2.2 RU Builder Partitioning

FED Builder partitioning can not be used when only one RU Builder is present, which may be the case in the initial running of the DAQ. Partitioning can then instead be implemented in the RU Builder. In this scheme the Event manager is provided with the association of sub-detectors i.e. RUs to different DAQ partitions. The EVM must then send the event number for the DAQ partition in question and its associat-

ed Event_ID only to the RUs associated with that DAQ partition and not to the RUs associated with other DAQ partitions.

The BUs are configured to know which DAQ partition they belong to and which RUs belong to the partition in question. BU requests to the EVM for Event_IDs include the DAQ partition identifier, so that the EVM can return only Event_IDs that have been allocated to RU's belonging to the same partition. The physical EVM contains multiple logical EVM tasks, i.e. one EVM task per DAQ partition. (See Figure 4-8 for an example.)

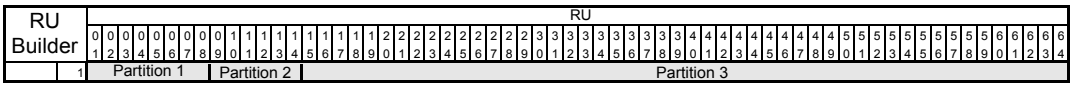


Figure 4-8 Example of partitioning in the RU Builder. Partition 1 uses RUs 1 to 8, Partition 2 uses RUs 9 to 14 and Partition 3 uses the remaining RUs. The system has only one RU Builder used by all partitions.

If more than one RU Builder is installed, then in principle all of them can be used. Each DAQ partition must then use the event number of the DAQ partition to multiplex events between the RU Builders exactly as in normal physics mode.

Restrictions

- The number of DAQ partitions is limited only by the number of TTC partition groups (8) allowed in the trigger. No RU can participate in more than one DAQ partition and no RU may read data from more than one TTC partition group[1]. Each FED Builder may therefore only be connected to FEDs from a single TTC partition, so as to not restrict the possible grouping of TTC partitions.
- The three additional data streams from the trigger can be assigned to different TTC partition groups and should ideally not share a FED Builder. However with only 64 FED Builders available, it may be necessary to connect all three to the same FED Builder, which would mean that the three data streams must all be assigned to the same TTC partition group, so that they end up in only one DAQ partition.
- The allocation of BUs to partitions must take into account the restrictions imposed by RU Builder traffic shaping if high efficiency is required.
- DAQ partitions share equipment, such as the EVM and the RU Builder network, which augments the risk for interference between the DAQ partitions.

4.4.2.3 Comparison of FED Builder and RU Builder Partitioning

The advantage of the RU Builder partitioning is that it allows up to 8 DAQ partitions in a single RU Builder. The disadvantage is that some elements like the EVM and the RU Builder switch are shared between partitions. This can be a problem if one of the partitions misbehaves and generates too high trigger rates or too large data volumes. In case of multiple RU Builders this sharing has to be replicated in each RU Builder.

The advantage of the FED Builder partitioning is that it minimises the amount of equipment shared between DAQ partitions and it requires no complications in the EVM implementation. It also has the advantage of not applying any multiplexing based on data coming from the FEDs, which makes it resistant to

1. Without this restriction the DAQ software in the RU would need to implement multiple logical RUs, one per partition. This would reduce the independence of partitions and increase the complexity of the DAQ software.

incorrect event counting in the FEDs. The disadvantages are that only few RU Builders may be installed at the start of LHC operation, where the need for partitioning may be highest, and resources maybe wasted because each partition can only use the RUs in its allocated RU Builder(s).

The FED Builder partitioning is the favoured scheme because it minimises the amount of shared equipment and the design is simpler. Not only will this minimise the risk of interference between partitions, it should also make it easier operationally to allow one partition to start and stop runs while others continue running. The FED Builder partitioning also allows a finer granularity of the DAQ partitions because it does not impose the restriction of only one TTC partition per FED Builder.

4.4.2.4 Combining the two Partitioning Schemes

The RU Builder partitioning is necessary, when the DAQ has only one RU Builder, but it does not lend itself well to a multi-RU Builder configuration. The FED Builder partitioning only works with multiple RU Builders, but may be too limited if the number of RU Builders is small.

It is possible to combine the two schemes in several ways (see Figure 4-9). If for instance only four RU Builders were implemented and it were desired to run with 5 DAQ partitions, then the FED Builder partitioning could be used to assign 1 DAQ partition to each of the four RU Builders plus one additional DAQ partition to one of the four RU Builders. The RU Builder with two DAQ partitions would then need RU partitioning to cope with the two partitions.

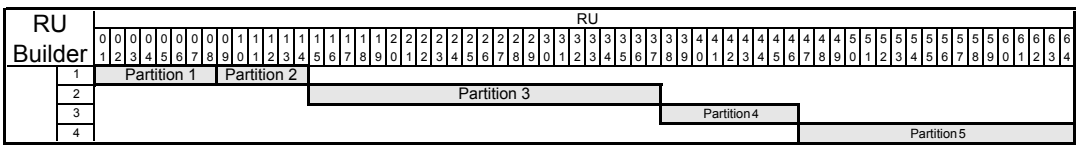


Figure 4-9 Combining the two partitioning schemes. Partitions 1 and 2 in RU Builder 1 require the RU partitioning scheme to share RU Builder 1, while the other DAQ partitions get assigned their own RU Builders in the FED Builder partitioning scheme.

4.5 Summary

This chapter has described the two-stage event builder at the functional level. It has defined the requirements on the FED and RU switches and demonstrated that the performance scales with the number of RU Builders. Not all RU Builders will be needed at start-up, when the detector and accelerator will operate well below the nominal values assumed for a full-scale event builder.

It has been demonstrated that the event builder can be partitioned to match the partitioning scheme of the trigger allowing for independent data flows in each partition.

The chapter on event builder networks will demonstrate that this design can be implemented with current technology and that it can also take advantage of future improvements in switching technology.

4.6 References

4-1 CMS Trigger/DAQ group, ed. J. Varela, “CMS L1 Trigger Control System”, CMS Note 2002/033.

5 Event Flow Control

Event Flow Control serves to regulate the dataflow from the trigger driven readout of the front-end systems to the demand driven delivery of full events to the Filter farms. It also serves to protect against overflow of intermediate buffers which might cause loss of synchronisation.

This chapter first describes the event flow control issues from Front-End Systems to the Builder Units and then concentrates on the implementation of the Event Manager, which performs a central role in regulating the event flow in the RU Builder.

5.1 Introduction

The two stages of the CMS Event Builder, i.e. the Front-End Driver (FED) Builder that creates super-fragments (“s-fragments”) for up to eight outputs and the Readout Unit (RU) Builder that combines these s-fragments into complete events, have very different event flow control requirements. FED Builders “push” s-fragments into the big buffer memories of RUs and the only flow control is a “backpressure” mechanism, to reduce or eventually stop the rate of Level-1 Triggers. The routing of an event through a FED Builder is a static function of its event number and the number of DAQ RU Builders available (per partition).

RU Builders on the other hand contain Builder Units (BUs) with free resources that “pull” events from the RUs. This allows RU Builder traffic to dynamically adapt to the throughput of each BU and its associated Filter farm. Each RU Builder has an Event Manager (EVM) that manages the dataflow in the RU Builder and keeps track of the memory occupancy of the RUs. The Event Manager is able to request a reduced trigger rate if one of the RUs is running out memory for buffering incoming s-fragments.

The Event Manager deals with three logically separate networks (see Chapter 6). The Readout Control Network (RCN) is used to give each s-fragment a unique Event_ID label to be used in subsequent operations as described below. The Builder Control Network (BCN) is used by the BUs to request Event_IDs from the EVM, to request the corresponding s-fragments from the RUs, and to release Event_IDs back to the EVM when they are considered fully processed. The Builder Data Network (BDN) is used by the RUs to send their s-fragments to the BUs. An important issue in the event flow control is the efficient use of the BDN. Care must be taken to avoid some BUs not receiving s-fragments while other BUs are overloaded. The Event Manager is neither concerned with the communication between the BUs and the filter farm, nor with the downstream computing services. When a Filter Unit does not keep up with its share of the event rate, the EVM will observe the corresponding BU requesting events at a lower rate. When too many BUs fall behind, the EVM will observe a steady increase in the occupancies of all the RUs and will request a reduced trigger rate as soon as a critical level is reached. Whenever a RU or BU is found to behave anomalously beyond a given threshold, the EVM will signal the Run Control and Monitor System (see Chapter 12, "Run Control and Monitor System").

5.2 Trigger-Throttling System (TTS)

The first element of event flow control is implemented in the last stage of the detector readout hardware, i.e. before the central data acquisition. It is a hardwired system, designed to act on the dataflow in time scales of a few hundred nanoseconds.

All the Front-End Drivers (FEDs) and their associated Front-End Systems operate independently and synchronously with the LHC clock (see Section 7.3.1, "Front-End Driver Overview").They respond to Level-1 Triggers and store the data corresponding to the trigger in local FIFO buffers together with the content of their local Event and Bunch Crossing counters. The content of the Event Counter, the “event number”, is used throughout the Event Builder as a key to building events

An error in the event number of any Front-End System (FES) or FED will mean a loss (or an incorrect assignment) of data from that Front-end during event building. If not detected and corrected such a synchronisation error will be present in all following events, until a resynchronisation has been applied.

A recovery procedure involving the start of a new run is a long process and may lead to unacceptable DAQ inefficiency unless the rate of these errors is extremely low. A hardware-based recovery procedure involving only the trigger system has therefore been designed. It is based on the Trigger Timing and Control System and the Trigger-Throttling System. A brief description of these systems is given here. More details can be found in Section 7.6, "Flow Control and Front-end Synchronization".

5.2.1 Trigger Timing and Control (TTC) System

The TTC is the fast opto-electronic system used to distribute the Level-1 Trigger as well as some commands from the Global Trigger Processor (GTP) to all the Front-End Drivers.The commands include resets of various counters (Event counter, Bunch crossing counter, Orbit counter), start and stop of data taking plus a resynchronisation (Resynch) command.

5.2.2 Synchronous TTS (sTTS)

The sTTS is the system that provides the feedback from all the FEDs to the Global Trigger Processor (GTP). The sTTS consists of 4 signals: Ready, Busy, Synch failure and Overflow warning. Each FED reports the state of these signals to the sTTS, which then presents a summary of the state of all the FEDs to the GTP following the logic in Table 5-1.:

Table 5-1 sTTS inputs to Global Trigger Processor.

GTP input from sTTS	Function of all FED outputs
T3 (Ready)	Logical AND of all Ready
T2 (Busy)	Logical OR of all Busy
T1 (Synch failure)	Programmable majority algorithm of all Synch failures
T0 (Overflow warning)	Logical OR of all Overflow warning

The sTTS is organized by trigger partition group so that a state set by a FED will apply only to the trigger partition group that the FED belongs to. The four signals are encoded to represent up to 16 different system states (see Section 7.6.2, "Synchronous TTS (sTTS) Implementation").

5.2.2.1 FIFO Overflow Handling

An overflow of a FIFO in a FED or FES will not only cause loss of data, but may also lead to synchronisation errors. Precautions are therefore taken at the level of sTTS to reduce the risk of these overflows. If

any FED FIFO buffer fills beyond a predefined high water mark during data taking, then the FED will assert the Warning signal, which will be removed again once the FIFO content falls below a low water mark. The GTP interprets the warning as a request for a reduction of the trigger rate.

If data stops flowing for whatever reason in any of the FED Builders, then the reduced trigger rate will cause some FED buffers becoming full. The affected FEDs will then assert the Busy signal, which will stop all further Level-1 Triggers until the blocking has been cleared or the failing FED Builder has been taken out of the Data acquisition system.

5.2.2.2 Synchronisation Recovery

Whenever any FED detects a synchronisation error it will assert the “Synch failure” signal. The GTP interprets the “Synch failure” to be a request for a Resynch command, which it will then execute. Note that all event counters will be reset to zero during the resynch command. The Resynch procedure includes the inhibit of all Level-1 Triggers followed by a wait to allow all pending events to flow out of the FIFO buffers of the FEDs. Then the Event Counters in all FEDs are reset and data acquisition resumes once all FEDs report Ready.

5.2.3 Asynchronous TTS (aTTS)

The purpose of the aTTS is to give processors that are handling the event data flow a possibility for interacting with sTTS to enable or disable Level-1 Triggers, request reduced trigger rates or a resynchronisation of all the FEDs in the DAQ partition. The processors that handle the data flow have large memories capable of buffering several hundred milliseconds worth of data, so the timing requirements are not as tight as those of sTTS. Reaction times of tens of milliseconds are sufficient for aTTS. Synchronisation errors can only be detected when event and bunch crossing counters from several FEDs can be compared. The most complete check can therefore be made in the BU after the full event has been built. The RUs can check synchronisation errors only within a super-fragment, but can catch the errors before the potentially long wait for final event building.

The aTTS is a computer system which interfaces to the sTTS and GTP. Processors in the DAQ system can exchange messages with aTTS and request it to act on the sTTS of their own partition on their behalf. The aTTS can also read the status of the GTP for each trigger partition group and can thus use this status to decide which command to send to the GTP over sTTS[1].

The Event Manager of each RU Builder can interact with aTTS to request a reduction of the Level-1 Trigger rate when the RU Builder comes close to saturation. The aTTS will need to obey some rules on how to analyse and act on multiple, possibly conflicting, requests received within a given time window. These rules will be simpler when fewer elements are allowed to send requests to the aTTS.

Figure 5-1 illustrates how sTTS and aTTS are integrated with the data flow components.

1. If, as an example, a RU Builder reports a synchronisation error to aTTS, then the aTTS system will initiate a Resynch through setting the sTTS “Synch failure” line for the corresponding DAQ partition. This will cause the GTP to take this partition out of the data taking state. When the same RU Builder reports more synchronisation errors from the following events, which were buffered before the resynch was issued, then the aTTS will not request a resynch for each of these, because it can see from the GTP status that this partition is no longer in the data taking state.

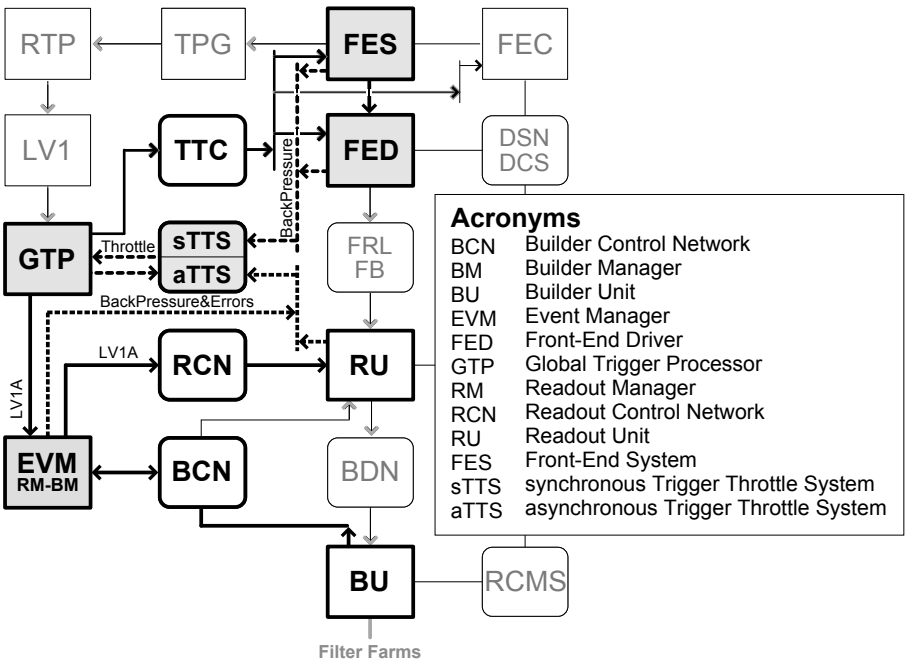


Figure 5-1 sTTS and aTTS integration with major data flow components.

5.3 Event Builder Protocols and Data Flow

5.3.1 FED Builder

The Event builder has 64 FED Builders, reading data from the subdetector Front-End Drivers (FED), plus one special FED Builder, which reads trigger information from the Global Trigger Processor (GTP). The event flow control in the FED Builder for the GTP is described separately in Section 5.3.1.3.

The FED Builder is the most time critical element of the Event builder as it must read all the events from the FEDs with a minimum of delay. The operation is time critical because the FEDs have limited buffer capacity and DAQ inefficiencies will occur if any FED buffer fills up. The required performance is achieved at the cost of some loss of flexibility. The FED Builder has thus no automatic load balancing, because this would require a central controller for all the FED Builders operating at 100 kHz. The latency requirement on the interconnect between such a central controller and the 64 FED Builders would be difficult to satisfy.

5.3.1.1 Flow Control in the FED Builder

The dataflow in the FED Builder has already been described in Section 4.2.5, "Data Flow in the FED Builder", so only the flow control handling of exceptional circumstances is described here. Two options are possible, with or without back pressure from the RU.

5.3.1.1.1 Flow Control through RU Back Pressure

Congestion in one RU in any of the 8 RU Builders will cause back pressure to be asserted to the corresponding FED Builder. The FED Builder Input (FBI) and FRL buffers of this FED Builder will then start to fill up and finally the FIFO buffer in of one of the FEDs connected to this FED Builder will reach the high water mark and issue an sTTS warning signal to reduce the Level-1 Trigger rate. If the cause of congestion is transient, then data will start to flow to the RU again, the buffers will be freed and the FED will release the sTTS warning signal after the FED buffer has been emptied below a predefined low water mark.

If the congestion is permanent, then the FIFO buffer of the FED will become full and the FED will issue an sTTS busy level, which will inhibit any further triggers. All the FED Builders then need to have their routing changed to exclude the RU Builder that blocks the data flow.

One RU Builder may develop a reduced throughput capacity through failing BUs or FUs. This may lead to congestion in several RUs, causing assertion of the sTTS warning signal by several FEDs. The trigger rate will be reduced and data will keep flowing into the RU Builder. If the input rate is low enough then the buffers will be freed and FEDs will release the sTTS signals. The triggers will resume with normal rate and continue until the FED buffers associated with the underperforming RU Builder fill up again and reassert the sTTS Warning signal. All DAQ RU Builders will thus have the throughput reduced to the level of the slowest RU Builder.

It may then be more efficient to remove the underperforming RU Builder or possibly send it a smaller fraction of the events. In either case all the FED Builders, including the GTP FED Builder, will need to have their routing changed.

5.3.1.1.2 Data Flow without RU Back Pressure

The FED Builders can also be designed to always accept all data from the FEDs even if the receiving RUs are not able to accept the data. All that is required is that the RUI simply discards super-fragments that cannot be delivered to the memory of the RU.

In this case the data will continue to flow normally to all the RU Builders that have no throughput problems and only the RU Builder which has RUs with full memory buffers will have some missing super-fragments. This error will be detected in the RU Builders.

5.3.1.2 Change of FED Builder Routing

A change of routing must be decided at a higher level in the DAQ system, possibly through operator intervention. The triggers are paused and a purge of all buffers in FEDs, FBIs and RUIs must be made in order to start the new routing cleanly. The FBIs, which perform the routing may have several routing tables stored in memory, so a change of table in all FBIs can be performed by simply changing the number of the routing table to use. The change can in principle be made at the same time in all FED Builders by requesting the FED Builders to perform the change at a given (future) event number.

A detailed design for this procedure has not yet been made.

5.3.1.3 FED Builder for the GTP

The FED Builder for the GTP has only one input port, which is connected to the GTP through an interface that is identical to the FED interface used for all the detectors. The 8 output ports are connected to the Event Managers (EVM) of the 8 RU Builders.

Triggers arrive in the EVMs naturally in increasing event number order. The GTP trigger data block is much smaller than the average fragment size transmitted from the FEDs, so the total traffic through the GTP FED Builder is much smaller than the traffic through the other FED Builders.

5.3.2 RU Builder

The Event Builder has up to 8 RU Builders, each consisting of 64 RUs, 64 BUs and one Event Manager. The data flows through the RU Builder on request from each of the BUs. This ensures that the data flow adjusts to the throughput capacity of each BU and its associated FU farm.

5.3.2.1 Event Flow Protocols

The data flow has already been described in Section 4.3.7, "RU Builder Data Flow". It is governed by a set of protocols described in Appendix C, "Event Flow Control Protocols in RU Builder". These protocols were designed with the following aims:

- Provide load balancing of the traffic through the RU Builder
- Be resistant to a reset or “wrap-around” of the event number
- Optimise access to event fragments in the RUs
- Ensure that each event normally is built by only one BU
- Allow for the possibility of having more than one BU building an event for debugging purposes.

The Event Manager and a logical Event identifier, the “Event_ID”, were introduced to achieve these aims. The Event Manager is a central entity that controls and monitors the flow of events through the RU Builder, and the Event_ID is a unique logical identifier assigned to each event currently being processed by the RU Builder.

The Event Manager maintains a pool of Event_IDs, each of which is either free or associated[1] with an event that is currently present in the RU Builder. The Event Manager associates every incoming event with a free Event_ID and distributes this association to all RUs in the RU Builder over the Readout Control Network. Once an Event_ID has been associated with an event number, it cannot be assigned to any other event. If the event counters are reset, then new events entering the RU Builder may have the same event number as those of events already being processed by the RU Builder. This scenario does not pose a problem to the RU Builder, as the Event_ID is unique. An Event_ID becomes free again once the relevant BU and the associated FUs have finished processing the corresponding event data.

The Event Manager controls the flow of data through the RU Builder through the distribution of Event_IDs. A BU with free resources asks for new events by requesting a group of Event_IDs from the

1. When associated with an event, the Event_ID can be in either the “open” or the “allocated” state as described in Section 9.2.2, "Fault Tolerance of the Event Building Nodes"

Event Manager. It is up to the discretion of the Event Manager to allocate an Event_ID to only one BU for normal operation or to more than one for debugging purposes.

The protocols described in Appendix C, "Event Flow Control Protocols in RU Builder", were designed for the original 512×512 Event Builder operating at 100 kHz where the RUs and BUs were expected to be implemented in hardware with deterministic timing behaviour. This protocol is still the baseline protocol for the RU Builder, but the more relaxed timing requirements that govern the RU Builder now make some extensions to this protocol possible. Protocol extensions may be introduced later to implement new features, such as more extensive monitoring and control in the EVM. A brief summary of the baseline protocol is outlined in Table 5-2, Table 5-3 and Figure 5-2.

Table 5-2 Baseline protocol between BU, EVM and RU.

Sender	Receiver	Message
BU	EVM	A BU requests a new event by sending an “allocate” message to the EVM
EVM	BU	EVM replies with a message, that includes the Event_ID of the event allocated to the BU
BU	RU	BU issues a “retrieve fragment” request with the Event_ID to all the RUs in this partition
RUs	BU	Each RU replies with a message, which includes the Event_ID and the associated s-fragment
		Once all RUs have replied, the full event is resident in the BU, ready for processing by the Filter Farm
BU	EVM	When the filter farm has finished processing this Event_ID, the BU sends a “clear Event_ID” message to the EVM

Table 5-3 EVM to RU protocol.

Sender	Message
EVM	The EVM sends a “readout” message containing a free Event_ID and the associated event number to all RUs in the partition. This message also serves as an implicit “clear” of the previous fragment data associated with the Event_ID

5.4 Event Manager

The Event Manager reads the trigger summary block from the GTP FED Builder for every event addressed to its RU Builder. This information is formatted with the header and trailer defined in the Common data encapsulation format (See Section 7.3.3, "FED Data Format" and Figure 5-4).

The trigger summary block contains both a trigger number and an event number. The difference is that the trigger number counts all the Level-1 Triggers received by the GTP since it was initialized, while the Event number counts all the Level-1 Triggers received by this partition since the last trigger “Resynch” in this trigger partition group. The event number follows the same logic as the Level-1 Trigger counter in all the FEDs and is used to identify all the data fragments belonging to the same trigger.

The GTP has two connections to the DAQ; one is to the EVM and the other is to a normal RU. The trigger summary block described here is a subset of the more complete trigger data sent to the RU port. When operating in partition mode (see Section 5.4.1, "DAQ Partitioning in the Event Manager"), the trigger data from the RU port of the GTP will belong to one partition only, while the trigger summary block received by the EVM will be available to all partitions. The trigger summary block, which uniquely defines the

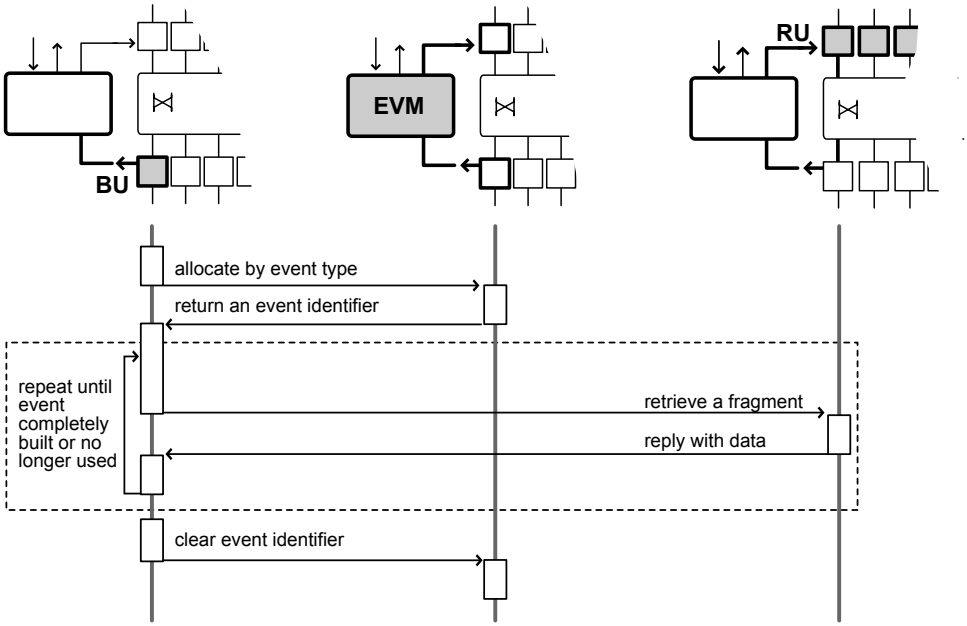


Figure 5-2 Graphical description of RU Builder protocol for event building.

Table 5-4 Trigger summary block from Global Trigger Processor. Note that the shaded rows correspond to the header and trailer of the Common Data Encapsulation.

Content	Comment	Size in bits
Event number	Matches event counters in FEDs of this DAQ partition	24
Bunch crossing number	Bunch crossing counter. max value=3564	12
Orbit number	Orbit counter for 12h	32
Trigger number	Number of all L1A since GTP initialized	24
DAQ partition number Trigger type	Bits 6...4 Bits 3...0	7
Algo bits_3	Physics trigger algorithm bits 127...96	32
Algo bits_2	Physics trigger algorithm bits 95...64.	32
Algo bits_1	Physics trigger algorithm bits 63...32	32
Algo bits_0	Physics trigger algorithm bits 31...0	32
Technical Trigger bits	bits 31...0	32
GPS Absolute Time	Bytes 1 to 4 of BST GPS time	32
GPS Absolute Time	Bytes 5 to 8 of BST GPS time	32

event should therefore be recorded with each event.This can be done by transmitting this data block together with the Event_ID to the BUs. (This is an extension of the baseline protocol.)

The EVM supervises all event traffic in the RU Builder. It consists of two functional parts, the Readout Manager (RM) and the Builder Manager (BM). The primary task of the RM is to propagate the Event number and bunch crossing number together with the Event_ID to all the RUs. Note that if the DAQ is operating in Partitioning mode, then only a subset of the RUs must be addressed. See Section 5.4.1, "DAQ Partitioning in the Event Manager".

The primary task of the Builder Manager is to communicate with all the BUs in a peer-to-peer fashion, sending on request Event_IDs and any associated information, such as the trigger summary block to the requesting BU.

5.4.1 DAQ Partitioning in the Event Manager

DAQ partitioning is described in Section 4.4.2, "DAQ Partitioning". Two modes of partitioning are envisaged. The so-called FED Builder partitioning, where the FED Builders are used to route different DAQ partitions to different RU Builders, and the so-called RU Builder partitioning (Figure 5-3), where a single RU Builder must handle several DAQ partitions concurrently.

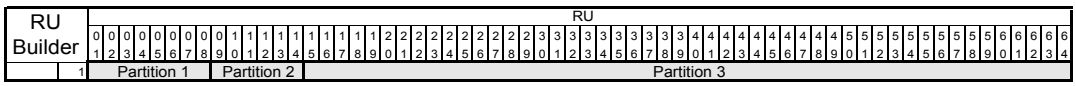


Figure 5-3 RU Builder partitioning, with 3 partitions sharing one RU Builder.

The RU Builder partitioning is likely to be needed initially, when only a few RU Builders will be available. This is the most complex operating mode for the EVM because one physical EVM must act as multiple logical and independent EVMs, one per partition.The trigger summary block contains the DAQ partition number, which the Event Manager must use to determine which RUs to address. The Event Manager sends the Event_ID-event number association only to the RUs that belong to this DAQ partition.

The association of TTC partitions to global trigger partitions (which equal DAQ partitions) is configured by software in the GTP. The same information must also be sent to the EVM so that it can associate RUs with DAQ partition numbers. The EVM must also be configured to know which BUs belong to each partition and each BU must be configured to know which RUs are in the same partition as itself.[1]

5.4.2 Event Manager Networks

The Event Manager communicates with the RUs over the RCN logical network and with the BUs over the BCN logical network. Experience from the Event builder test-benches has shown that, for the two network technologies considered for the RU Builder (Myrinet and Gigabit Ethernet), both the RCN and the BCN logical networks can be implemented on the same physical network as the BDN. The combined network is referenced as the Readout Builder Network.

The EVM has two connections to the Readout Builder Network, one for the RM and one for the BM (see Figure 5-4). The two connections replace one BU and one RU connection respectively. The total number of RUs and BUs is therefore reduced from 64 to 63, thus reducing the maximum switch throughput capacity by 1/64 or 1.6%.

1. Alternatively this can be done dynamically by transmitting the pattern of participating RUs together with the Event_ID information in response to a BU “allocate” message.

The advantage of this arrangement is that only one physical network is needed. The network load generated by the RM and BM traffic is orders of magnitude smaller than the RU and BU traffic, and has a negligible effect on the event data flow.

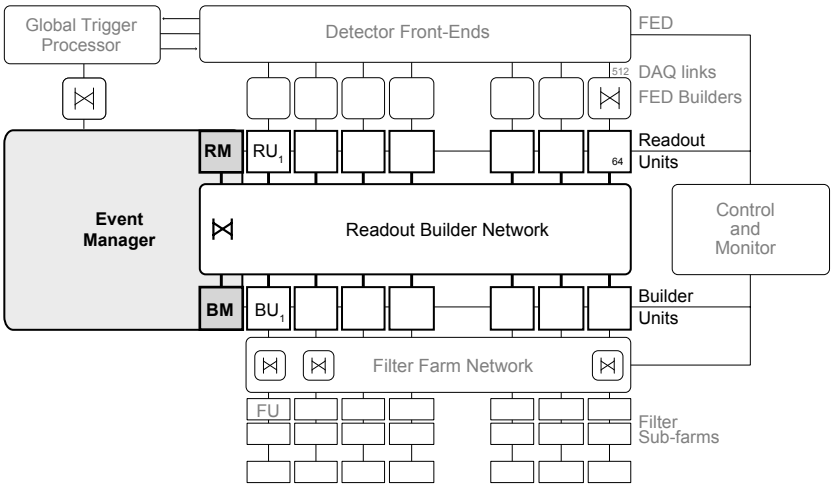


Figure 5-4 Event Manager interconnection with Readout Builder Network.

5.4.3 Interface to the Run Control and Monitor System

The Event Manager must be configured at the beginning of each run with the list of Readout Units and Builder Units in that run. The configuration will be done through the DAQ run control.

Partitioning of the RU Builder must be defined by Run Control before any start of run as each partition can start and stop runs independently.

5.4.4 Monitoring

The EVM is the central place from which the operation of the RU Builder is controlled and monitored. It is a convenient place to gather information about the RU Builder because a large fraction of this information is already there. If the EVM has the full knowledge of the state of the RU Builder it can better take the right corrective action, when some part of the system does not perform correctly.

The quantities that should be monitored, pertinent to the correct functioning of the RU Builder include:

- Data throughput at each RU. (So that the EVM can compare and report the data rate of all RUs in the RU Builder)
- Memory occupation in each RU. (So that the EVM can request a reduction of the trigger rate before a RU runs out of memory)
- Distribution of s-fragment and fragment sizes sent by each RU. (Serves to indicate if one of the FED Builders has too high a frequency of abnormal event sizes)
- Event rate at each BU (will be a function of the processing power of the FUs attached to each BU). Variations in time can indicate problems with the BU or the associated Filter units.

The RUs and BUs can be kept simple, dealing with data flow only, if all monitoring and control is concentrated in the EVM.

Note that the monitoring must be done independently for each of the DAQ partitions.

5.4.4.1 Test Patterns

In order to test the integrity of the system, on request of a Filter Unit, the EVM should be able to issue a "send test pattern" command on the RCN, that has the effect that every RU creates a pseudo event fragment containing a test pattern uniquely identifying that RU. The Event_ID corresponding to this special event, should then be given to the BU requesting this event, so that the BU can build this event. Occasionally each Filter Unit might request such a test event.

5.4.5 Prototypes for the Event Manager

There are two prototype systems to develop and test the EVM: the hardware test-bench and the computer simulation of the event builder system. The two approaches are complementary. The EVM hardware and software parameters, protocols and throughput are verified in both.

5.4.5.1 Hardware Test-benches and Schedule

A small test-bench system which is flexible enough to emulate all possible solutions of the DAQ design is necessary for the EVM and the RCN development. A system of several network switches with generic PCs is a natural solution because of its flexibility. It is easy to change network arrangements in such a system. Incremental upgrades of the system keep the system up-to-date and facilitate a gradual shift to the final configuration. This hardware and software arrangement also provides various network parameter measurements as input to the simulation.

The test-bench system was first built in 1999 and has been upgraded several times since.

5.4.5.2 Test-bench Hardware

The current test-bench system for event manager studies has nine PCs, one VME CPU, one 16-port Myrinet switch, two Fast Ethernet switches and two IEEE1394 6-port hubs. The network connections can be changed physically and logically to meet various test requirements.

All PCs are Intel x86 based, running the Linux OS. With nine PCs, a system of up to 4×4+1 (4 RUs, 4 BUs and an EVM) can be configured. The EVM PC has two independent PCI buses to minimize interference between the RCN and the BCN traffic. The VME CPU board can be used either as a simple GTP or as a hardware RM. The Myrinet network (currently 1280 Mbps with LANai7 chip based interface cards) is used for the BCN/BDN. The VME CPU and the dual-PCI PC are directly connected with Myrinet-2000 cards (based on LANai9) in order to test the GTP-EVM connection, which is point-to-point. Myrinet, Ethernet and IEEE1394 networks may be configured as the RCN.

5.4.5.3 Test-bench Software

In order to support a wide spectrum of technologies in the test-bench the operating systems used are Linux and VxWorks, which support the most popular network devices. In addition, there is a wide variety

of application software and development environments supported by these OSs. Depending on the real-time performance test results, RT-Linux and/or other OSs can also be tested on the test-bench.The EVM software is implemented in the CMS DAQ software framework, XDAQ.

5.4.5.4 Benchmark Tests

The hardware and software environments described above enable benchmark testing with the following aims:

- Network technology evaluation: One can choose from several network technologies for the RCN based on their performance and availability. FastEthernet, IEEE1394 and Myrinet are considered. For each of them, latency and bandwidth are checked to see if they meet the RCN requirements.
- Parameters for simulation: Network performance numbers measured with benchmark tests are put into the system simulation to get more realistic results.
- Study of the network configuration: The logical network design is already fixed, but the mapping to the physical network is not yet decided.Throughput measurements of the entire system will give the optimal design of the RCN configuration.
- System integrity tests: The EVM code is tested on the test-bench to make sure that the code, including error handling, works correctly.
- Performance check: The final EVM system is checked to see if it satisfies all requirements.

In the course of these tests, robust EVM and RCN systems will be developed.

5.4.6 Simulation of the Event Manager

The operation of the Event Manager is simulated as part of the simulation of the event builder system, involving RUs, BUs and the network fabrics connecting all entities.The simulation tools used are described in Appendix B.4.1, "Simulation Package".

Since there was concern about the ability of the Myrinet switch to handle the event traffic efficiently, the Myrinet switch and network cards have been modelled in great detail. For other entities, e.g. RU, BU and EVM, the focus has rather been on the simulation of the message protocols and the memory usage vs time. Each BU emulates a subfarm of filter nodes. A filter node takes a random amount of reconstruction time per event, according to a distribution that may have been determined from reconstructing Monte Carlo events offline. The input distributions of the event fragment size per RU and of the times between subsequent triggers can be determined similarly.

The simulation is only concerned with a single RU Builder of the complete event builder system, assuming that all RU Builders operate with negligible mutual interference. The EVM is logically split into Readout Manager and Builder Manager, which are presumed to be implemented as threads in a single process on a reasonably powerful PC. In the nominal model the BM is connected to the Myrinet switch on the RU side, thereby leaving room for 63 RUs. The RM is connected to the BU side, leaving room for 63 BUs. This configuration allows the Myrinet switch to implement the three logical networks RCN, BCN and BDN. The simulation results show that such a configuration is viable, but the code also allows the RCN to be implemented as an external network (Ethernet, Firewire), in which case the last slot on the BU side can be used for an extra BU.

The simulation of the Myrinet switch has been shown to be in excellent agreement with test-bench “demonstrator” measurements at CERN. The Ptolemy results for the whole event builder system are in good

agreement with results obtained in an earlier, simpler simulation based on the CNCL C++ simulation framework [5-1]. In that simulation the Myrinet switch is considered a perfect crossbar, in which blocking can only occur at the endpoints of a route through the switch. That assumption is reasonable when the vast majority of the traffic, viz. the event data traffic over the BDN, is made nearly 100% efficient by means of “barrel-shifting”, as is the case in the CMS DAQ. The CNCL-based simulation has shown fair agreement with the EVM test-bench measurements done at Fermilab, given its limitations.

Direct comparisons of the Ptolemy event builder simulation with the real system will be made when the test-benches at CERN and Fermilab have a complete set of hardware and software (XDAQ) to implement a full RU Builder (in 2003).

5.5 Summary

The control of the event flow from the Front-End Drivers to the Builder Units has been described. It has been demonstrated how the flow-control in the two-stage Event Builder meets the dual requirement of a trigger-driven input and a demand-driven output. The “push” protocol with backpressure in the FED Builder satisfies the event flow requirement of the trigger-driven data collection while the “pull” protocol regulated by the Event Manager in the RU Builder satisfies the requirement for demand-driven delivery of full events to the Filter Units. The Readout Units with their large buffer memories serve to decouple the two data flows. The possibility to throttle the trigger from the flow control logic provides the tool for limiting the trigger rate if the event flow momentarily exceeds the capacity of the system.

The function of the Event Manager and its integration in the RU Builder is well understood. Initial benchmarks, prototyping and simulation have been performed. Test systems and simulation models have been prepared for evaluating different possible implementations of the EVM.

5.6 References

5-1 ComNets Class Library and Tools, see http://www.comnets.rwth-aachen.de/doc/cncl.html.

6 Event Builder Networks

The switching networks used to interconnect the entities in the DAQ are important elements of the Event Builder (EVB). This chapter describes the design and possible implementations of these networks. The actual description is preceded by an introduction and two review sections that aim at giving the reader the understanding of the networking issues needed to follow the description of the implementation studies. The layout of this chapter is as follows:

1. Review of some basic properties of switching networks along with the additional requirements that event-building traffic imposes on the architecture of a switching network. (Section 6.2)
2. Brief description of the actual network technologies, switched Ethernet and Myrinet, that are currently considered as leading candidates for the Event Builder networks. (Section 6.3)
3. Description of implementation studies of the FED Builder and the RU Builder networks. The requirements, the design, the results from prototypes and simulation, as well as prospects for the future are discussed. Finally, issues related to the full EVB with both stages combined, i.e. the FED Builder and the RU Builder, are discussed. (Section 6.4 to Section 6.9).

The chapter provides a summary of the main results and conclusions obtained over the past several years in the subject of networks for the Event Builder. When a more detailed discussion was deemed useful, the corresponding material was placed in the Appendix.

6.1 Introduction

The central feature of the DAQ design is the factorization of the event building process into logically and possibly physically distinct steps:

- front-end readout up to the Front-end Readout Link (FRL)
- FED-building consisting of the transfer of data from multiple FRLs to a single Readout Unit (RU)
- RU-building consisting of the transfer of data from multiple RUs to a single Builder Unit (BU)
- Transfer of the event to the Filter Farm, i.e. from a Builder Unit (BU) to a Filter Unit (FU)

These stages are decoupled from each other via memory buffers. This decoupling allows the collection of FED Builders and RU Builders to be considered as a system whose function is almost entirely independent from the other two functions of the DAQ, namely the front-end readout and the feeding of processors in the filter farm with new events.

In the two-stage EVB design, the large (about 500×500) switching network is divided into two stages comprising a number of smaller switching networks. The two stages are decoupled by intermediate buffers (the Readout Units). From the network point of a view, a major advantage of this architecture is that the performance requirements are more easily attainable than in a single-step event building process involving a large monolithic switching network without intermediate buffers. The disadvantage of this solution is that the two-stage process requires more network interfaces. At the time of writing of this Technical Design Report, it is assumed that the two-stage design will use 8×8 switches in the FED Builder and 64×64 switches in the RU Builder. Depending on the details of the technologies selected, a different topology can be chosen.

In the context of the Event Builder, a M×N switch means a switching network that connects M data sources to N data sinks. In the FED Builder, the sources are the FRLs and the sinks are the Readout Units

(RUs). In the RU Builder, the sources are the RUs and the sinks are the BUs. The ports of all nodes (sources and sinks) must operate at an effective speed of about 2 Gb/s[1].

Over the past several years, CMS has performed numerous studies of various event building architectures, switching technologies and communication protocols. Some of this work has been reported in [6-1] and [6-2]. The scale of the switching network test-benches utilized has varied from 4×4 to 32×32. Clearly, a plan of work in which a full prototype of the final system is implemented is not cost effective. For this reason, the performance of the full final system has to be predicted on the basis of simulation.

An important measure of performance in the system, given a particular network technology, is the utilization efficiency, i.e. the ratio of the throughput of the event builder and the theoretical bandwidth of the switch. The cost effectiveness of any particular implementation of the architecture clearly depends on the efficiency with which the switches in the network can be used. The measurements performed, along with the results obtained from the simulation programs, have included the overall sustained throughput of the switching network, the event rate, and the latency of the data in the system. Measurements have typically been carried out for various switch sizes.

An important parameter for any Event Builder is the distribution of data sizes across the inputs and across events. In the studies presented here, event fragments of variable sizes are generated to mimic the expected data from the FEDs. At this stage, the precise distribution of the fragment sizes delivered by the FEDs is not known. The sizes are generated according to the log-normal distribution[2] with different parameter settings. As an example, the frequency distribution is shown in Figure 6-1 for an average size of 2 kB and an rms of 2 kB, which is expected to be a good model of the event fragment size distribution. Note that the distribution is asymmetric, with the most probable value (about 1 kB) significantly different from the average value.

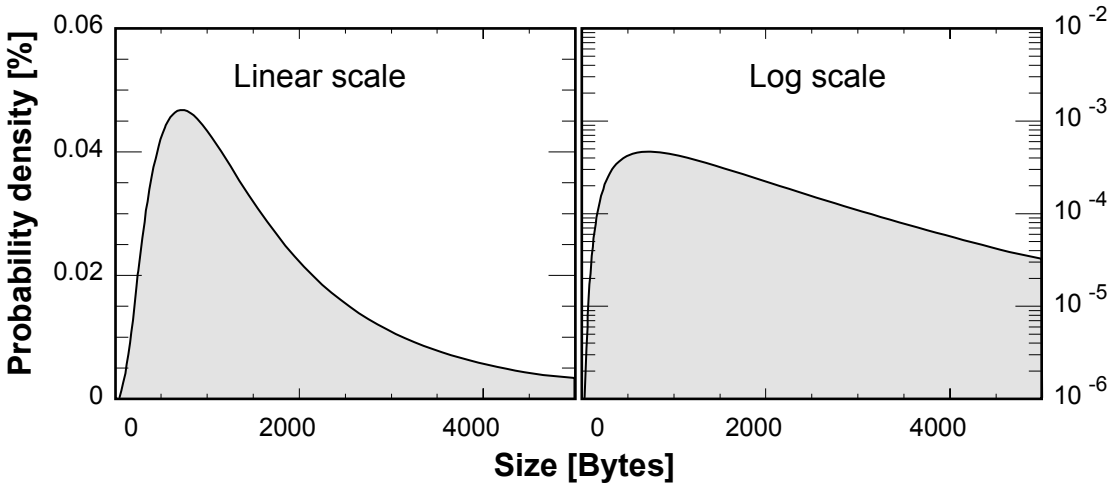


Figure 6-1 Fragment size distribution for variable size test (linear and log scale) for parameter values set to an average of 2 kB and an rms of 2 kB.

1. If the switching network does not support this effective speed, it is still possible to achieve the 2 Gb/s by using multiple parallel switching networks, referred to as ‘rails’ in the literature. In this case, each node accommodates one port per “rail”.
2. The log-normal distribution is defined in Appendix B.4.2.1, Table B-4

In the real system there will be correlations among the amounts of data in the FEDs in any single event. As an example, the amount of data in the tracker system is essentially proportional to the number of pp interactions in the beam crossing in question. Therefore, an increased amount of data in one FED should in general be accompanied by similar increased amounts in other tracker FEDs. Correlations in data sizes are also expected for FEDs belonging to different detectors (e.g. the occupancy in both the calorimeters and the tracker are expected to be higher than average in the core of a hadronic jet). These effects have been studied in the FED Builder prototypes described in Section 6.4.

Finally, an important design decision affecting the parameters of the Event Builder is the choice of communication protocol. This protocol can be “lossless” or it can be “unreliable”. TCP/IP is an example of a lossless protocol, while UDP is an example of a non-guaranteed delivery protocol. This is not an issue if the network hardware is inherently lossless, as is the case for one of the technologies studied. However, if the chosen network hardware is not lossless, then a protocol like TCP/IP is desirable as it results in a system with fewer possible error conditions and overall simpler operation.

6.2 Event Building with Switching Networks

This section reviews some fundamental properties of switching networks along with the additional requirements EVB traffic imposes on them. Methods to increase the utilisation efficiency are outlined.

6.2.1 Switching Architectures

Switched networks are widely used in Local Area Networks (LANs) and in High Performance Computing (HPC) clusters. The switches consist of a combination of ideal switching elements (“crossbars”) with buffering (queues) on the input and/or output ports. The role of the queues is to absorb instantaneous fluctuations of traffic bursts. Limitations in the queue size or bandwidth reduce the efficiency of the switching network. Appendix A, "Overview of Switching Techniques and Technologies" gives an overview of available switching techniques explaining the relative merits of the different queuing techniques employed in switches.

6.2.2 Event Building Traffic

There are two major differences between the data traffic in an Event Builder and in a typical LAN and HPC cluster:

- In an Event Builder, all data sources and destinations operate continuously at approximately the same data rate which, to maximize cost performance, should be as close as possible to 100% of the switching network bandwidth. In a typical campus LAN, however, the load is not constant and its mean value is often below 20% of the capacity available.
- In an Event Builder, all sources must send data (fragments from a single event) to the same destination. The process is shown schematically in Figure 6-2. The traffic pattern of a typical LAN is more random and most nodes communicate only with a few other nodes.

HPC clusters running fine-grain parallel applications, also have stringent requirements to the switching network; but unlike Event Builder applications, it is not throughput but latency, that is critical for this type of application.

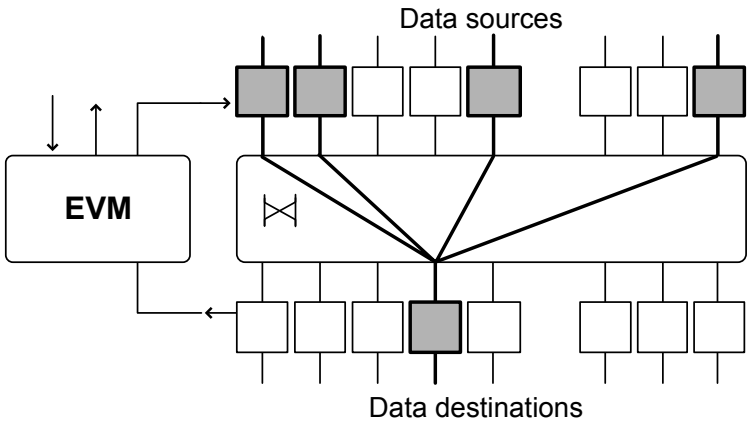


Figure 6-2 Schematic display of the traffic implied in Event Building. All sources must communicate, eventually, the data belonging to the same event to the same destination.

6.2.3 Traffic-shaping

Even assuming the existence of a N×N switch with crossbar like connectivity, the data traffic pattern in the Event Builder implies blocking or loss of data unless the switch provides output queues that are large enough to store the equivalent of an entire DAQ event[1]. For an efficient use of the switch, the switching capability of the cross-bar must be supplemented with some packet-level algorithm that provides an arbitration mechanism for the output link. The procedure used to share this link, as well as to ensure that all the other switch links are used concurrently, is referred to as “Traffic-shaping”.

6.2.3.1 Barrel-shifter

Assuming a balanced system, where on average each source sends data to all destinations at the same rate, one can exploit the characteristics of event building traffic with the “barrel-shifter” technique [6-3], shown schematically in Figure 6-3. The basic idea is that the sequence of sends from each source to each destination follows the cyclic permutations of the sinks. Concretely, if at a certain point in time source i is sending event j to sink k, then, during this same time period, source $(i+1)$ sends event $(j+1)$ to sink $(k+1)$[2]. Assuming that event data are always available for sending to the sinks, the barrel-shifter uses 100% of the bandwidth of a non-blocking switch, after the first N-1 transfers from the first source are complete.

In practice, the simple barrel-shifter (BS) scheme just described needs to be augmented with additional procedures to account for the varying fragment size in each source. Without these procedures the entire system can lose its synchronization once two sources have very different amounts of data. So, while the BS traffic-shaping scheme is 100% efficient for constant event sizes, it can be very inefficient when applied to events of variable sizes.

A solution is to segment the variable size fragments into fixed-sized units, or carriers, before transmitting them to the switch. The order in which carriers are sent to different destinations follows the cyclic permutation order of the barrel-shifter. The carriers are reassembled back into variable-size fragments at the out-

1. This issue is discussed in detail in Appendix A.1, "Queuing Architectures".
2. If any of these integers exceeds N, the number N is subtracted from them.

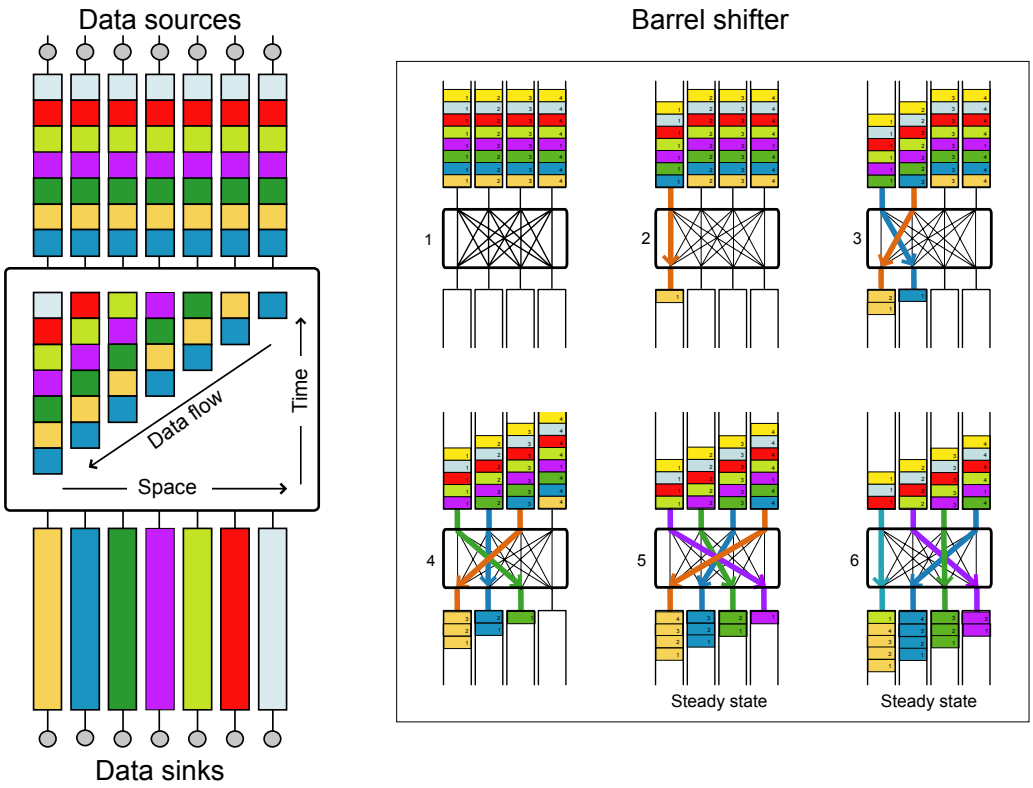


Figure 6-3 Schematic of the Barrel-Shifter Traffic-Shaping scheme. During the first time slot, only source 1 is sending data (to destination 1). During time slot 2, source 1 is sending to dest. 2, while source 2 is sending to destination 1. During time slot m, source 1 is sending to dest. m, source 2 to m-1 and so on. After N time slots (where N is the number of sources) all sources are sending data to mutually exclusive destinations.

put. This is equivalent to having source-driven Time-Division Multiplexing of the data to the various sinks.

In the simplest case, fragments are split into carriers without packing data from two adjacent fragments into the same carrier, i.e. the end of the first fragment and the beginning of the next. This can nevertheless be reasonably efficient, provided the carrier size is small compared to the typical fragment size. If, on the other hand, the carrier size is relatively large compared to the fragment size, high efficiency can only be obtained by packing data from different fragments into a carrier. The disadvantage here is that the segmentation and reassembly process are more complicated.

6.2.3.2 Destination-driven Traffic-shaping

In this scheme, one relies on the destinations to send their requests for data to the sources in an order that inherently avoids clashes. In practice, the RU Builder (described Section 6.5, "RU Builder") is driven by the destinations, i.e. the Builder Units (BUs) which send requests for data only once they have buffers available. The simplest way to shape the traffic at the destination is to assume that the destination sends a request to N RUs sequentially: a request to RU 1 (e.g. “send event 45”) is then followed by the actual data transfer of the data from event 45, at which point the request for the same event is sent to RU 2, and so on, as shown in Figure 6-4. No data is transmitted into the BU link during the time the request message is sent and processed. This results in a reduced utilisation of the switching network, depending on the latency of the network and processing times.

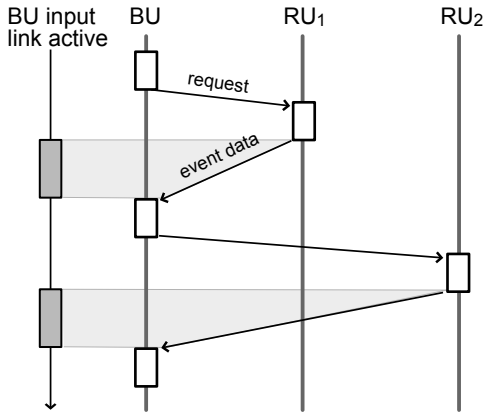


Figure 6-4 Principle of destination based traffic-shaping.

6.2.3.3 Switch-based Traffic-shaping

In this scheme, the switch employed uses internal buffering as intermediate storage for data that cannot momentarily be sent to the desired destination. If this buffering capability is large enough (e.g. greater than at least N events in the DAQ) then the sources need no knowledge of time-sharing among the destinations, and can simply send their data, assuming a cross-bar like connectivity, at any point in time.

Switches of this kind with a large number of 1 Gb/s ports already exist, albeit at relatively high cost. Nevertheless, the dependence of system performance, as well as its overall balance, on the internal characteristics of a specific proprietary switch is not a desirable feature. For this reason, this type of traffic-shaping, while the easiest one to implement, remains the least attractive.

6.3 Network Technologies Considered for EVB Networks

This section discusses the network technologies, namely Switched Ethernet and Myrinet, which are considered as possible implementations of the Event Builder networks. A comparison of the main features of these technologies is given in Table 6-1. The following subsections give a brief overview the two technologies and describe their use in an Event Builder network.

6.3.1 Switched Ethernet

The term Ethernet refers to the family of LAN products covered by the IEEE 802.3 standard protocol. Invented in 1976, Ethernet became a de-facto standard in 1980. Four data rates are currently defined for operation over twisted-pair cables or optical fiber (see Table 6-2).

Most of the LANs in use today are based on Ethernet. Ethernet started as a shared medium technology. Fast Ethernet introduced full duplex point to point links, which allowed a fast rise of data rates for switched Ethernet. The success of this standard is attributed to its support of ever-higher data rate capabilities combined with high flexibility in network installations and full inter operability between products from different manufacturers.

Table 6-1 Network technologies comparison.

	Switched Ethernet	Myrinet
focus	Local Area Network	cluster I/O
bandwidth	100 Mbps (Fast Ethernet) 1 Gbps (Gigabit Ethernet) 10 Gbps emerging	2.5 Gbps (8b/10b encoded[a])
latency (MPI application[b])	100 μs	10 μs
routing method[c]	destination-based	source-based
switching algorithm	store-and-forward (typically)	wormhole
switch type	shared memory (typically)	crossbar
flow control	Xon/Xoff pause frames (optional)	link level Xon/Xoff
medium	UTP5 up to 200 m, fiber	copper up to 3 m (SAN), fiber
MTU size	1500 bytes (user payload) jumbo frames (optional)	unlimited
multicast/broadcast	yes	no
processor on NIC	typically not	yes
market	multi-vendor	single vendor

a. Due to the 8b/10b encoding of the data the 2.5 Gbps baud rate results in a 2 Gbps effective bandwidth.

b. MPI (Message Passage Interface) is an industry standard employed widely in high-performance computing.

c. the routing method refers to whether the determination of the path to follow between source and destination is done at the source (source routing) or at each switching node along the way (destination based).

Table 6-2 Ethernet standard.

Bandwidth	Standard	Year of IEEE standardisation
10 Mbps	10 Base-T Ethernet	1991
100 Mbps	Fast Ethernet	1995
1 Gbps	Gigabit Ethernet	1999
10 Gbps	10 Gigabit Ethernet (fiber only)	2002

6.3.1.1 Switched Ethernet Technology

An Ethernet network is composed of switching elements and NICs connected by point-to-point bidirectional links. Store-and-forward switches permit interconnection of links with different data rates. Packet size is limited to 1500 bytes of data (plus header and trailer). The NIC is in general not programmable; but the switches are configurable and support extensions to the standard such as VLAN, which effectively divides a switch into several smaller switches.

The switches are in general non-blocking, i.e. the crossbar can support the aggregate rate of all the switch ports; packets can still get lost because of capacity limitations in the store and forward memories. Flow

control is optional and is implemented through the use of pause frames generated by the Media Access Control layer of the receiving NIC. Flow control is generally implemented between NIC and switch port, but not always through the switch back to the source NIC.

6.3.1.2 Switched Ethernet for Event Building

The bandwidth of Gigabit Ethernet is a factor two below the bandwidth required by a single data source (be it a FRL or a RU) in the Event Builder, so a Gigabit Ethernet solution requires at least two rails. 10-Gigabit Ethernet is not an option at present as it only supports products for switch-to-switch communication and not for the host-to-switch communication needed in the Event Builder.

The small frame size of Ethernet is a drawback because it increases the load on the hosts. A number of companies have equipment that support bigger packet sizes, the so-called ‘Jumbo frames’. Although these are not part of the IEEE standard they are becoming a de-facto standard supported by many Ethernet equipment manufacturers

The IEEE 802.3 physical layer does not guarantee error-free transmission. Packets may be lost if buffer space is not available in any of the switches between the sender and the receiver. The protocols needed for event building with Ethernet must be implemented on programmable hosts as long as the availability of programmable NICs is not guaranteed[1].

A lightweight application level protocol has been tested in a system with 64 hosts (see Section 6.5.2). The performance depends on the characteristics of the switch. A simulation of this setup is under way and it is hoped that this can be used to evaluate how the system scales with a doubling of the number of ports and how performance depends on the memory size of the switch.

The TCP/IP protocol would be more convenient as it guarantees error free transmission over Ethernet, but it is demanding on the resources of the host. Preliminary results of event building using TCP/IP over Ethernet are given in Appendix B.3, "Gigabit Ethernet TCP/IP Test-benches". The basic conclusion is that, in spite of the overheads, it is now possible with TCP/IP to drive 2 GbE links near the full wire rate using a 2 GHz Pentium-IV Xeon based PC. With the ever increasing PC performance, TCP/IP over GbE could become an attractive candidate for the RU Builder over the next few years.

Overall, switched Ethernet is a valid candidate for the RU Builder, but it is not the leading candidate for the switches in the FED Builder (See Section 6.4.3).

6.3.2 Myrinet

Myrinet is a high-speed interconnect for clusters, based on networking principles previously used in massive parallel processors [6-4].

6.3.2.1 Myrinet Technology

A Myrinet network is composed of switching elements and network interface cards (NICs) connected by point-to-point bidirectional links. The effective link speed is currently 2 Gb/s. The core of the switching chip is a pipelined 16-port crossbar that supports non-blocking wormhole (also known as cut-through)

1. As an example, some programmable NICs that were available at the time of the prototypes, described later in this chapter, have had their software support withdrawn.

routing of packets. Packets can be of arbitrary size. Network link-level flow control guarantees the delivery of packets at the expense of an increased potential for blocking in the switches. The NIC has a RISC processor whose firmware can be programmed to interact directly with the host processor for low-latency communications and with the network links to send, receive and buffer packets.

Myrinet is specified at the data link and physical level as an ANSI standard [6-5]. Myrinet products are manufactured by Myricom [6-6]. Details can be found in Appendix A.2, "Myrinet Technology and Products".

6.3.2.2 Myrinet for Event Building

The implemention of the Event Builder networks using Myrinet has numerous attractive features:

- large network fabrics can be constructed with ease
- data packets can be of arbitrary size
- there is no packet loss inside the switching fabric, while the bit error rate is very low
- the RISC processor on the NIC is programmable. This feature can be useful for implementing traffic-shaping, event building, or for optimized protocols. This is particularly relevant if the host is not a general-purpose fully programmable CPU but an FPGA-based device.

Even with a lossless switching fabric data can get lost due to overruns in the receiver buffers on the NIC or on the host. These losses can be avoided by flow control at the application level or by propagating pressure from the receiver host and NIC backwards to the sender. With a bit error rate below 10^{-15}, discarding those packets will lead to an insignificant efficiency loss and therefore there is no need for a mechanism for the recovery from such errors on an event-by-event basis. Only an error detection mechanism is still required.

A Myrinet network is essentially an input-queued switching fabric (see Appendix A, "Overview of Switching Techniques and Technologies"). Hence, depending on the traffic pattern, the throughput can be limited by head-of-line blocking. For random traffic, the maximum utilisation has an asymptotic theoretical limit of 59% for a single-stage network. For multi-stage networks the maximum utilisation is further reduced, as there is essentially no buffering in the switches.

Event building can be more demanding than random traffic since it creates a high degree of congestion at the switch output nodes. Simulation studies of event building with variable-size fragments show that, depending on details of the fragment size distributions and the number of stages, a ~50% utilisation of the bisection bandwidth is obtained. These studies are detailed in Appendix B.1, "EVB Performance with Myrinet Switches".On the other hand, a 100% utilisation of the network can be reached, in principle, with barrel-shifter traffic-shaping, assuming balanced inputs.

Overall, Myrinet is a valid candidate for the RU Builder switches as well as the FED Builder switches.

6.4 FED Builder

This section discusses the first stage of the Event Builder; the FED Builder. It starts with a brief review of its main functions. A possible configuration of the FED Builder based on the expected data rates from all CMS subdetectors is outlined. This is followed by a discussion on the network implementations for the FED Builder. Implementations with Ethernet and with Myrinet are considered. Measurements made on

prototypes and performance estimates made through simulation show that the networking requirements of the FED Builder can be met with the networking technology available in 2003.

6.4.1 Introduction

The main functions of the FED Builder are (see Section 4.2)

- to read the FED data and act as a buffer to smooth the bursty arrival of data.
- to assemble event fragments of variable sizes from some 600 FEDs into 64 super-fragments of roughly equal size (16 kB each)
- to multiplex these super-fragments on an event-by-event basis between N ($N \leq 8$) RU Builders, i.e. the second stage of the Event Builder.

These functions are achieved by 64 FED Builder switches, each with 8 input and 8 output ports. The input ports are connected to the FRL, whereas the output ports are connected to the RUI.

A 8×8 switch is the current design choice for the FED Builder, but other configurations can be implemented easily as well. In particular, a 16×8 switch configuration can be used to merge more FRLs as an alternative to merging low rate FEDs in the FRL.

6.4.2 FED Builder Configuration

At the time of writing, it is foreseen that the CMS readout will contain a total of ~630 FEDs which are concentrated into ~460 FRLs in order to reduce the number of EVB switch inputs. The details of this scheme are described in Section 7.2.3, "Sub-detector Readout Parameters". The FRLs are the inputs of the FED Builders. A possible configuration of the FED Builder is given in Table 6-3. The breakdown for the tracker-silicon detector will be discussed in more detail below. The assignment of FEDs to FED Builder (FEDB) switches takes into account the following considerations:

- FEDs providing a relatively small volume of data can be concentrated into a single FRL. It is assumed that up to 2 FEDs can be merged. Merging by the FRLs is used for the Silicon-Tracker FEDs.
- Detectors are not mixed on the same FEDB switch, so that they can be operated independently
- FEDs (and hence FRLs) with a relatively high data volume are combined with FEDs with a relatively small data volume on the same switch in order to achieve super-fragments of about equal size. This applies to the tracker-silicon.
- Partitioning considerations add the constraint that FEDs from different TTC partitions may not be mixed on the same FEDB switch (see Section 4.4.2, "DAQ Partitioning"). This restriction holds for RU Builder partitioning. It ensures that no FEDB switch and associated RUI and RU will have to be shared between two DAQ partitions which always follow TTC partition boundaries. The present assignment takes partition constraints only into account for the tracker-pixel and tracker-silicon detectors. Strict adherence to these rules would require a substantial number of additional FED Builders for the other detectors with a small number of FEDs and a relatively large number of TTC partitions. However, it is probably acceptable in practice, that the DAQ partitioning is more restrictive than the constraints imposed by the TTC partitioning.

About half of the total data rate is expected to come from the tracker-silicon detector. The data rate expected from each individual FED of the silicon tracker has been estimated (see [6-7]). FED-to-FED vari-

Table 6-3 FED Builder configuration.

Detector	TTC partition	# of FEDs	FEDs per FRL	# of FRLs	# of FEDBs
Tracker - pixel	barrel	32	1	32	4
	both endcaps	6	1	6	1
Tracker - silicon	inner	114	1 (54) or 2 (30)	84	11
	outer	134	1 (2) or 2 (66)	68	9
	endcap+	96	1 (24) or 2 (36)	60	8
	endcap-	96	1 (24) or 2 (36)	60	8
Preshower		47	1	47	6
ECAL		54	1	54	7
HCAL		32	1	32	4
Muon CSC		8	1	8	1
Muon RPC		6	1	6	1
Muon DT		5	1	5	1
Trigger[a]		6	1	6	1
Total		**636**		**468**	**63**

a. Trigger comprises 4 FEDs for global trigger, and 1 FED for each of the Muon track finders (CSC and DT).

ations are about a factor of three. The expected data rate is estimated to have an uncertainty of about 40%. A proposal on which FED data streams should be merged using the FRLs is summarised in Table 6-4. A possible assignment of FED Builders to these FRLs is given in Table 6-5. The resulting average super-fragment size varies between 6.5 and 9.3 kB. This assignment provides reasonably balanced inputs to the second stage of the EVB and the value is well below the nominal 16 kB size. This margin is probably necessary in view of the uncertainty on the data rates.

The current design choice is to use 8×8 switches in the FED Builder and 64×64 switches in the RU Builder. Hence, there are 63 FED Builders available, if one port of the RU Builder input switch is reserved for connection to the Event Manager. The current assignment uses 63 FED Builders, which leaves one free for contingency. However, the architecture is flexible, because more aggressive merging can be provided at the FRL level and by using N×8 switches for the FED Builder.

Table 6-4 Estimated data rate from the FEDs of the tracker silicon-detector and a proposed FED-FRL merging (from [6-7]). The FRLs have been sub-divided into categories depending on the number of FEDs they read from each layer[a]. An estimated average size of the fragments is also included. The sizes are for high luminosity pp collisions with FED zero-suppression. Note that the numbers are for one endcap only.

	FRL category	# of FRLs of this category	Reads FEDs from layers	average size (B)
inner	1	24	TIB-1	1030
	2	16	TIB(2+2)	1380
	3	12	TIB(3+4)	1250
	4	2	TIB(4+4)	1040
	10	10	TID1	810
	11	10	TID2	810
	12	10	TID3	810
outer	5	22	TOB(1+5)	1080
	6	24	TOB(2+6)	960
	7	14	TOB(3+4)	910
	8	2	TOB4	410
	9	6	TOB(5+6)	620
each endcap	13	32	TEC(1-3)+TEC(4-6)	1380
	14	4	TEC(1-3)+TEC(1-3)	1320
	15	16	TEC(7-8)	1010
	16	8	TEC9	1040

a. The tracker consists of 20 barrel layers and 24 end-cap disks. The inner and outer barrel layers are named TIB1-4 and TOB1-6, respectively. The end-cap disks consist of 3 mini-disks named TID1-3 and 9 larger disks named TEC1-9.

Table 6-5 FED Builder configuration for the tracker-silicon detector. The FED Builders (FEDB) have been sub-divided into categories depending on the FRLs connected to their 8 input ports. As an example, the first row means that there are 4 FED Builders of category 1, where each FED Builder has 2 inputs connected to FRLs of FRL category 1, 2 inputs connected to category 2, 1 input connected to category 3, 2 inputs connected to category 10 and 1 input port is free. The estimated average size of the resulting super-fragments is also included. The sizes are for high luminosity pp collisions with FED zero-suppression.

	FEDB category	# of FEDBs in this category	FEDB inputs from FRL category	average super-fragment size (B)
inner	1	4	2*FRL-1 + 2*FRL-2 + 1*FRL-3 + 2*FRL-10 + 1 free	7690
	2	1	2*FRL-1 + 2*FRL-2 + 1*FRL-3 + 2*FRL-10 + 1*FRL-11	8500
	3	1	2*FRL-1 + 1*FRL-2 + 2*FRL-3 + 3*FRL-11	8370
	4	2	2*FRL-1 + 1*FRL-2 + 1*FRL-3 + 1*FRL-4+ 3*FRL-11	8160
	5	2	3*FRL-1 + 1*FRL-2 + 1*FRL-3 + 3*FRL-12	8150
	6	1	2*FRL-1 + 1*FRL-2 + 1*FRL-3 + 4*FRL-12	7930
outer	7	3	8*FRL-6	7680
	8	2	7*FRL-7 + 1*FRL-8	6780
	9	3	5*FRL-5 + 2*FRL-9 + 1 free	6640
	10	1	7*FRL-5 + 1 free	7560
each endcap	11	4	4*FRL-13 + 1*FRL-14 + 2*FRL-16 + 1 free	8920
	12	4	4*FRL-13 + 4*FRL-15	9560

6.4.3 FED Builder implementation with Ethernet

Ethernet has been considered for the FED Builder switch, however a full prototyping effort has not been launched because of the following shortcomings of Ethernet with respect to Myrinet for this application:

- Bandwidth. The bandwidth limitation of Gigabit Ethernet would require a three-rail implementation.
- NICs. Gigabit Ethernet NICs supporting this number of rails are starting to become available, but are not, at least currently, programmable.
- FRL complexity. Ethernet NICs are not programmable, so the FED Builder protocol would have to be implemented in the FPGA of the FRL. The current FRL design uses a simple FIFO buffer between FED input and NIC output, whereas Ethernet NICs typically assume packet buffers to be in random access memory.
- Ethernet does not provide a lossless packet transport. (Packets may be lost if the switch momentarily has insufficient buffer space). A transmission protocol that includes detection and retransmission of lost packets is more complex and difficult to implement in FPGAs. This drawback does not hold when using switches implementing flow control.
- In order to implement back pressure from RU to FED, we need to restrict ourselves to switches which implement flow control.

- RU load. Without programmable NICs, the super-fragments would have to be built by the RU, which requires packet chaining or data copying in the RU memory. This is critical, because I/O and memory bandwidth is currently the most severe bottleneck in the use of PCs for implementation of the RU.

The advent of programmable multi-rail Gigabit Ethernet NICs or programmable 10 Gigabit NICs would remove most of these obstacles.

6.4.4 FED Builder Implementation with Myrinet

Myrinet supports reliable transport with back pressure at the hardware level, the NICs are programmable and have sufficient memory to smooth bursty arrival of data from the FRL. Myrinet supports a bandwidth per port of 2 Gb/s (or 250 MB/s). The switch utilisation will be roughly 50% when transporting variable size fragments (as discussed above in Section 6.3.2.2). The switch capacity therefore has to be doubled to absorb an average of 200 MB/s per port. LANai10 based NICs with two links per NIC are expected to be available in 2003. These can be used to construct a two-rail network, where each link of the NIC is connected to two independent 8×8 crossbar switches (see Figure 6-5).

The Clos-128 switches from Myricom provide 128 external ports with a three stage Clos network using 8×8 crossbars. It is therefore possible to configure these switches either as eight 8×8 switches or as four 16×8 switches by just reprogramming the routes of the wormhole packets. This could be used to enable the FED Builder to provide data concentration.

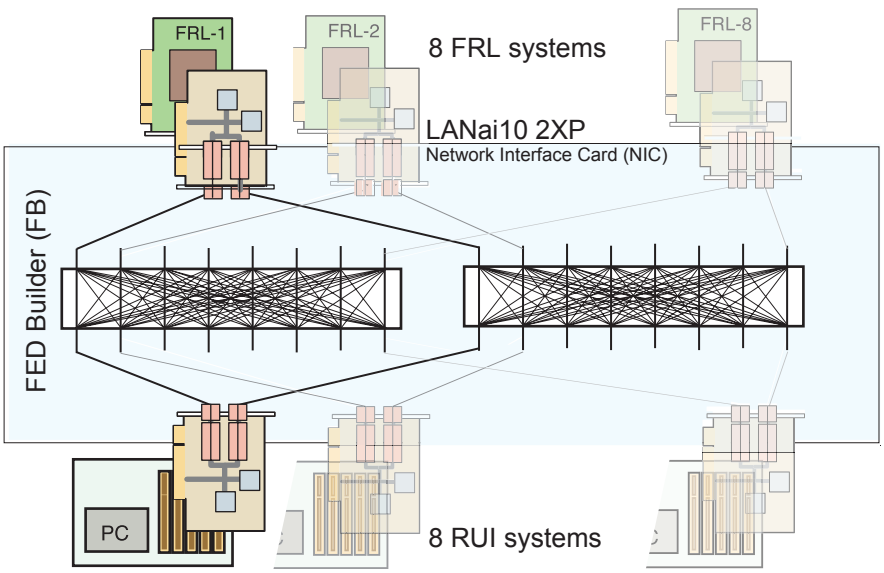


Figure 6-5 FED Builder based on Myrinet.

The FRLs with their NICs are located in the cavern close to the FED crates, whereas the switches are located at the surface in the DAQ room. The Myrinet links from the FRL NICs to the switches are standard optical fibers (multi mode 50/125 mm diameter) with a maximum length of 200 m.

6.4.4.1 Design

As described in Section 4.2, "FED Builder" the NICs at the FRL side correspond to the FED Builder Inputs (FBI) and the NICs at the RU side correspond to the Readout Unit Inputs (RUI).

The FBI is programmed to read event fragments from the FRL at the highest possible rate and buffer them in its memory in FIFO order. Event fragments in this buffer are transmitted to an output port at the rate that can be accepted by the link. The destination port for each fragment is derived by the FBI from the fragment event number. A lookup table addressed through the lowest 8 bits of the event number is used for this purpose. In the simplest case event number modulo 8 is used, but more complex algorithms can be stored in the lookup table.

The RUI, acts as a small Event Builder, concatenating event fragments with the same event number from all the FBIs and building super-fragments. The super-fragments may become available out of sequence since fragments arrive by two different switches (rails). It is the task of the RUI program to deliver the super-fragments to the RU in order of increasing event number. The RUI builds super-fragments directly in the memory of its host RU by using DMA chaining of all the fragments from one event as soon as they are all ready in the RUI memory.

The buffer memories in the FBI and RUI are organised in fixed sized blocks. All transfers of event fragments from the FRL to the FBI, and from FBI to the RUI are performed in units of blocks. The blocksize is called the FEDB block size. The FBI sends all blocks belonging to one fragment along the same rail through the FED Builder switch so that all blocks belonging to a fragment arrive in sequential order at the RUI. The buffer memory in the RU is also organised in fixed size blocks, but not necessarily of the same size as in the FED Builder.

The FBIs send the fragment blocks asynchronously with respect to each other. Hence, a deadlock situation can occur in the event building process in the RUI if the amount of buffer memory is not sufficient to store all the data corresponding to the range of triggers being processed. In order to avoid this, it is foreseen to implement a credit scheme, where the number of blocks sent by the FBI is limited by the credits received from the RUI.

6.4.4.2 Error Handling

The Myrinet switching fabric does not lose packets and the credit scheme protocol ensures that the receiving NIC (RUI) has buffer space available for incoming packets. Therefore, packets will not be lost in the FED Builder. However, packets can be corrupted due to transmission errors. These errors are detected via a CRC. All packets with CRC errors are simply dropped, but this leads to an insignificant event loss probability of about 10^{-8} with a Myrinet bit error rate below 10^{-15} and an event size of 10^{7} bits.

It is foreseen to impose an upper limit on the fragment size in order to protect against pathological cases. If the FBI receives a fragment which is too big, it will read the full fragment from the FRL, discard the data and insert an error flag in the header. It will then transmit only the fragment header to the RUI.

The RUI has a limited buffer space and can not wait indefinitely for data from the FEDs. The NIC releases the incomplete event for transfer to the RU if data from one of the FEDs does not arrive within a predefined time limit.

6.4.4.3 Test-benches

Different aspects of the FED Builder have been studied using two test-benches, both based on LANai9.

6.4.4.3.1 Switch Utilisation

The fraction of the available bandwidth of the switch which is actually used has been measured under event building conditions with a dedicated system (see Figure 6-6). As the LANai9 has a single port, only a one-rail configuration could be tested. Here, 8 sources are connected via a Xbar16 switch to 8 destinations. The sources and destinations are LANai9 based NICs with custom firmware. The data is moved from NIC-source to NIC-destination buffer memory, and then immediately discarded without event building. The sources can generate fragments of fixed and variable size. To study the effect of event-to-event fluctuations, fragment sizes are generated according to log-normal distributions with a number of different average and rms values. Fragment sizes are uncorrelated between all 8 sources. For these tests, each fragment was sent as a single Myrinet packet. An implementation with fragments cut into fixed size blocks is expected to give a better performance.

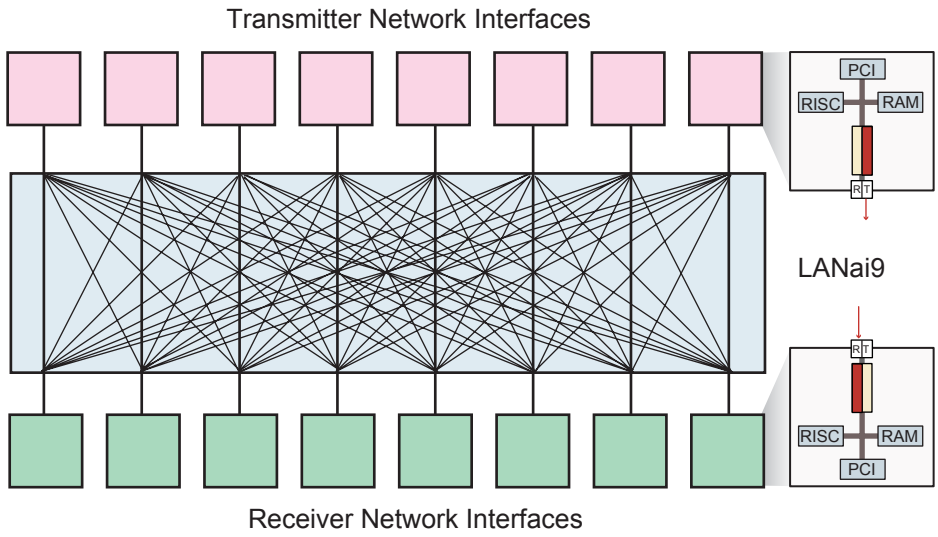


Figure 6-6 Layout of the 8×8 EVB used to study switch utilisation.

The throughput obtained is shown in Figure 6-7. The ‘balanced’ measurement series corresponds to the case where all sources send, on average, the same amount of data per event, namely 2 kB. It can be seen that the maximum utilisation drops with increasing rms/average, i.e. larger fragment-to-fragment size fluctuations, as expected.

In order to study the sensitivity to an unbalanced distribution of fragment sizes, the 8 inputs are divided in two groups, one group with fragment sizes above a 2 kB average and the other group with fragments of an average below 2 kB, while the sum of all inputs averages to 16kB. The imbalance is parametrized with the parameter *R* defined as

$R = \frac{\overline{L_1}}{\overline{L_2}}$ with $\overline{L_1}$ and $\overline{L_2}$ the average of the two groups with $\overline{L_1} \geq \overline{L_2}$.

A configuration with a division of 4-4 has been measured. The two sets of unbalanced series correspond to (a) the case where 4 sources generate on average 1.33 kB and the other 4 sources generate 2.66 kB, i.e. R=3 and (b) the case where 4 sources send 1 kB and 4 sources send 3 kB, i.e. R=3. In both cases, the total volume of data is 16 kB, and thus the attempted total data rate is the same. It can be seen that there is a small degradation for unbalanced inputs compared to balanced inputs. In general, the maximum utilisation is roughly 50% of available bandwidth.

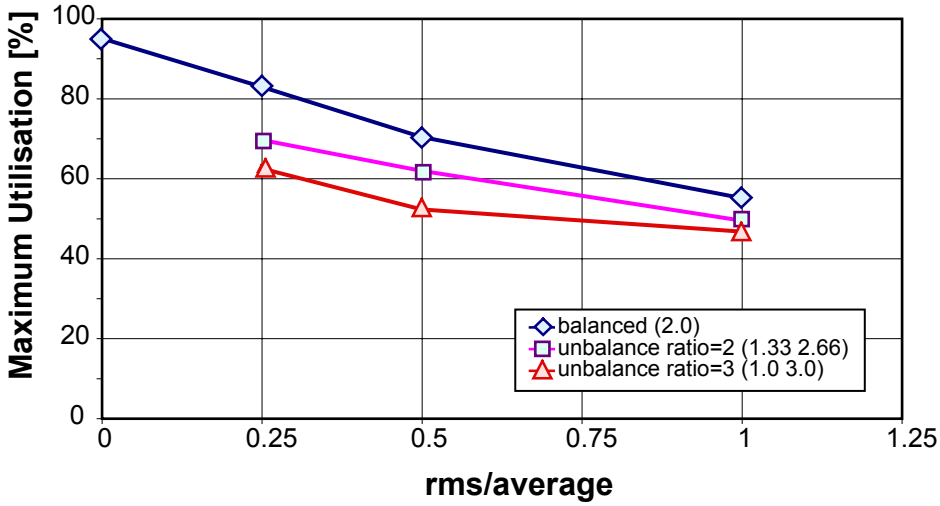


Figure 6-7 Switch utilisation for 8×8 EVB.

6.4.4.3.2 Building of Super-fragments

A FED Builder test-bench including a prototype FRL, Myrinet NICs and a RU is shown schematically in Figure 6-8. The FRL is implemented with the general purpose GIII PCI card (see Section 7.7.1, "The Generic Prototyping Platform GIII"). There is no input into the FRL, but the on-board FPGA logic includes a data generator to emulate the input. The FRL communicates with the NIC to push blocks of event fragments into the NIC buffer memory. Subsequently, the NIC, programmed as FBI sends these blocks to the receiving NIC, programmed as RUI. The RUI builds super-fragments and pushes them into the PC acting as RU. Although there is a single physical source and destination, the fragments are generated as coming from 8 different sources. This way the process of super-fragment building can be tested at full rate, without the rate penalty imposed by the 50% switch utilisation.

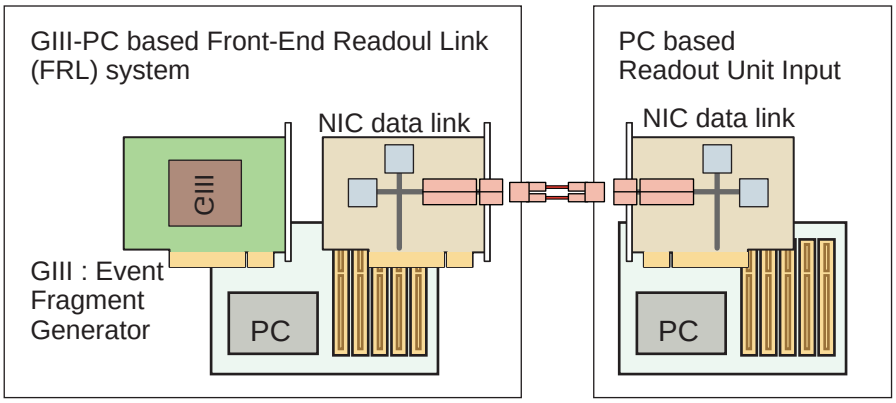


Figure 6-8 FED Builder prototype.

The achieved throughput is shown in Figure 6-9, compared to direct forwarding of fragments without any building, discussed in Section 7.7.4.1, "Protocol Between the FRL and the Network Interface Card". It can be seen that the additional overhead for super-fragment building is small. The ‘sawtooth’ structure is due to the FEDB block size being set to 4 kB[1]. For the nominal 2 kB fragment sizes, the throughput is

230 MB/s, corresponding to a trigger rate of 115 kHz. With a super-fragment size of 16 kB at the output of the FED Builder, this corresponds to building super-fragments at the rate of 14 kHz. It should be noted that this prototype version does not yet implement a credit scheme and error detection.

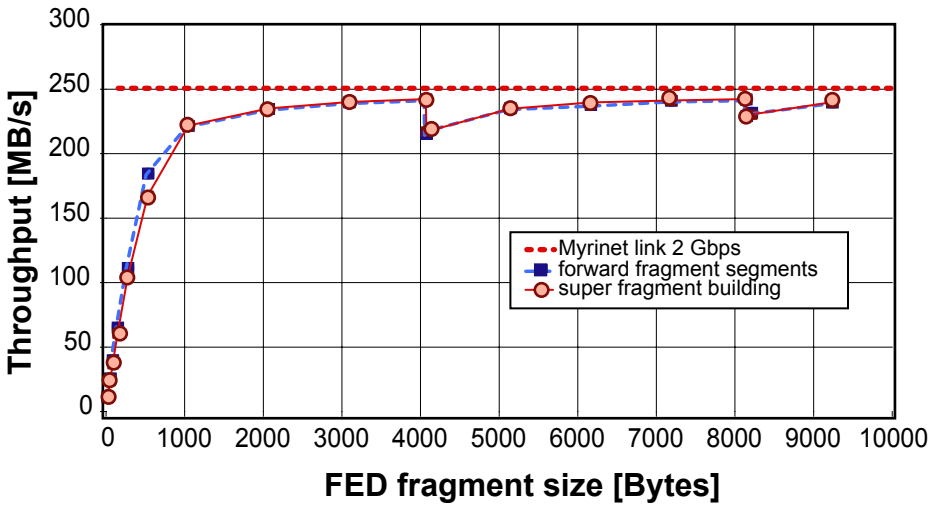


Figure 6-9 FED Builder prototype. Throughput versus fragment size.

6.4.4.4 Simulation

A detailed simulation model of the Event Builder complex has been implemented and is described in Appendix B.4, "Simulation of Myrinet Event Builder". In the following, results of the simulation are compared to data obtained with the FED Builder prototype described above. A prediction is made for the performance of the FED Builder based on LANai10 hardware and a two-rail network.

The data sizes in the CMS DAQ system are approximated by a log-normal distribution. However, it is likely, that the different data sources will be correlated and that the distributions will be different for different sources. To study the sensitivity of the throughput to these issues, the correlation of fragment sizes between data sources as well as unbalanced input distributions are studied. In rare cases a very big fragment might be emitted by one data source. The impact of these “jumbo-fragments” on the system performance is also tested.

6.4.4.4.1 Comparison of the 8×8 EVB with Test-bench Measurements

The results of the simulation for the switch utilisation of the 8×8 EVB are compared with the measurements presented in Section 6.4.4.3.1 in Figure 6-10. Reasonable agreement is obtained. The asymptotic value for large rms/average and imbalance ratio of 3 is about 45% (not shown).

6.4.4.4.2 Prediction for Two-rail FED Builder Based on LANai10

The performance of the FED Builder based on a two-rail LANai10 configuration has been estimated. The system operates in saturation mode, where the data sources always assume an event available. The

1. The RU blocksize was set to 80 kB, i.e. larger than the maximum super-fragment tested.

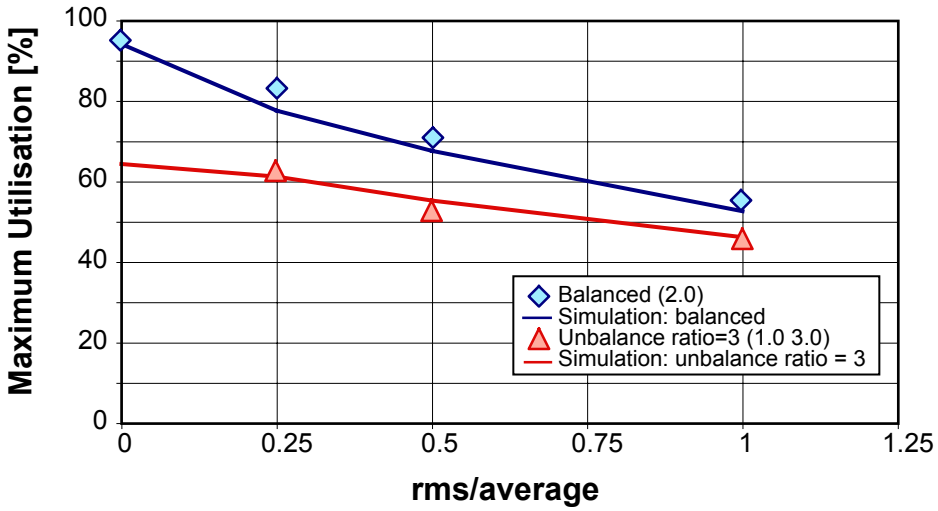


Figure 6-10 EVB 8×8. Simulation compared with LANai9 prototype measurements.

throughput per node for a number of fragment size distributions is shown in Figure 6-11. The average and rms of the log-normal distribution were set equal. For the nominal fragment size of 2 kB, the expected throughput is about 300 MB/s, corresponding to a 150 kHz trigger rate.

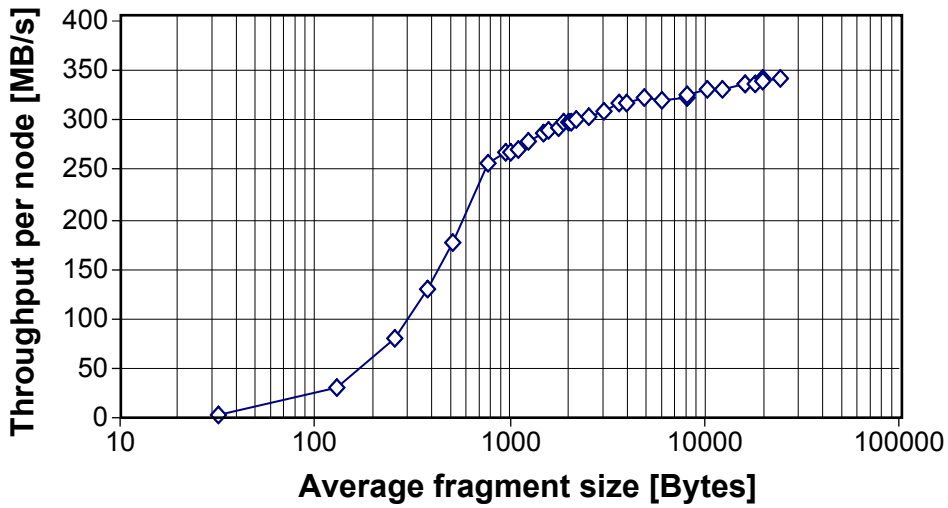


Figure 6-11 Estimated FED Builder performance for two-rail LANai10.

The trigger efficiency has been studied by implementing a trigger with poissonian distributed trigger-events and backpressure in the FED Builder. Here, a buffer capacity of 64 events in the FED and FRL combined was assumed. The efficiency of the trigger is defined as $\varepsilon = N_{acq} / N_{tot}$ with N_{acq} the number of acquired triggers and N_{tot} the total number of issued triggers. The trigger efficiency versus the trigger rate is shown in Figure 6-12. With LANai10 hardware, the FED Builder is capable of handling trigger rates up to 150 kHz for uncorrelated inputs at maximum efficiency. It is noted that the saturation throughput analysis gives a very good estimate for the maximum trigger rate.

The full system comprising 64 FED Builders based on LANai10 hardware has been studied as well. The trigger efficiency obtained is plotted in Figure 6-13 as a function of the trigger rate for LANai10. It is ob-

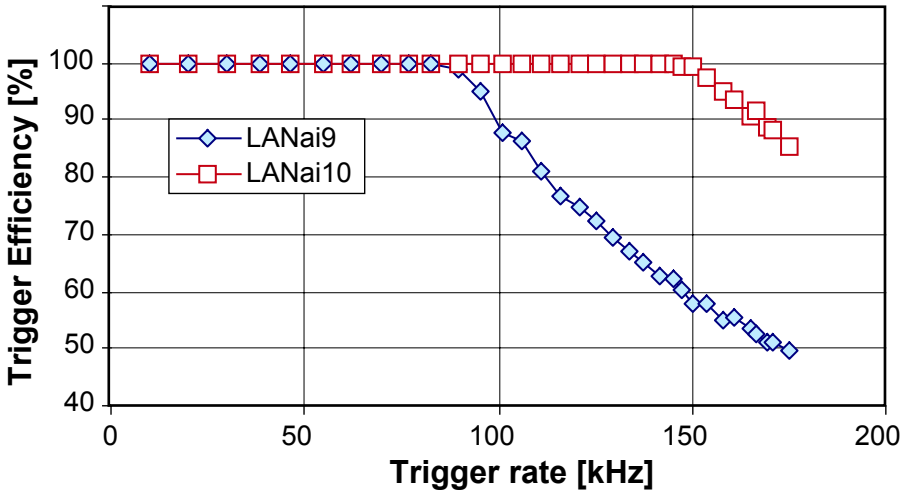


Figure 6-12 Trigger efficiency versus trigger rate for one 8×8 FED Builder and a log-normal fragment size distribution with average and rms set to 2 kB.

served that up to a trigger rate of about 150 kHz the system is fully efficient. This shows that the maximum trigger rate is not very sensitive to the number of FEDs in the system. The trigger efficiency of a system of multiple FED Builders is not simply the product of the single FED Builder efficiencies, because the throttling of the trigger affects all FED Builders, and hence the systems are coupled. This can be understood by the following argument. If one FED Builder is close to saturation and throttles the trigger, the other FED Builders will also receive no further triggers and hence empty their buffers as well.

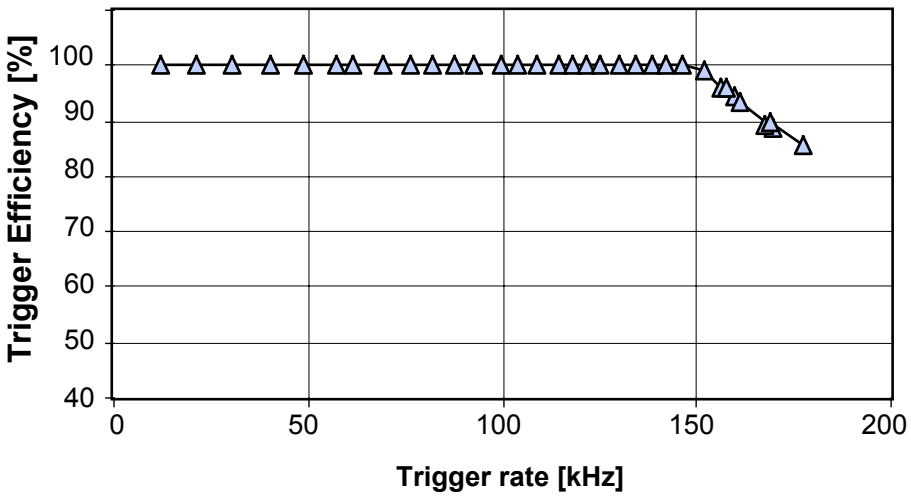


Figure 6-13 Trigger efficiency versus trigger rate for 64 8×8 FED Builders and a log-normal fragment size distribution with average and rms set to 2kB.

6.4.4.4.3 Data Conditions

The fragment sizes in CMS will be correlated between the FEDs. Hence, it is important to study also the influence of correlated fragment sizes on the utilization of the switch. To this end, fragment sizes are generated by a log-normal distribution with varying values for the linear correlation *r* between the different

sources, where r is defined as

$$r = \frac{S_{xy}}{\sqrt{S_{xx}\,S_{yy}}} \quad \text{with} \qquad S_{xy} = \sum(x-\bar{x})(y-\bar{y})$$

There are two limiting cases; a value of r=0 corresponds to uncorrelated sizes, the case considered so far, whereas with r=1 the sizes are identical in all sources (but with fragment-to-fragment fluctuations). The throughput as a function of r is shown in Figure 6-14. It is observed that the throughput is not sensitive to the correlation between variable fragment sizes for both LANai9 and LANai10 hardware.

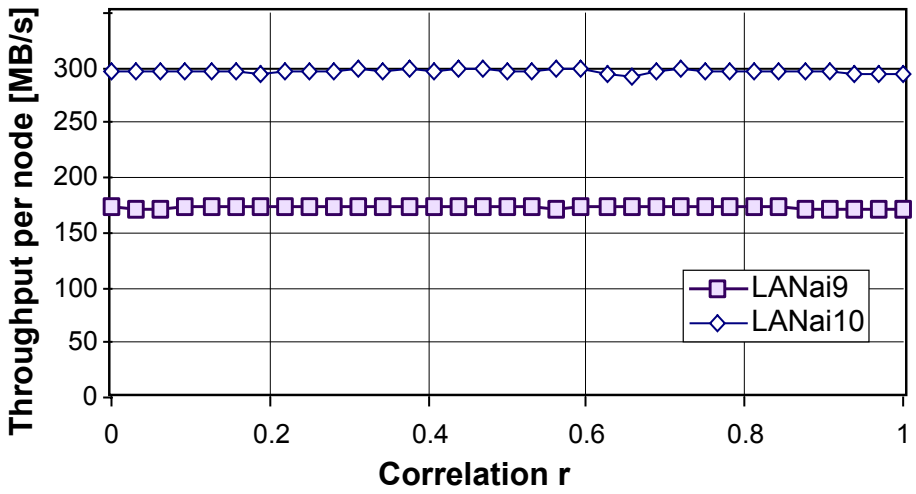


Figure 6-14 Throughput as function of correlation parameter between inputs.

The sensitivity to an unbalanced distribution of fragment sizes has been studied for two configurations; a division of 4-4 and a division of 1-7. The throughput has been simulated within a range of $1 < R < 6$ for LANai9 and LANai10 hardware (see Figure 6-15).

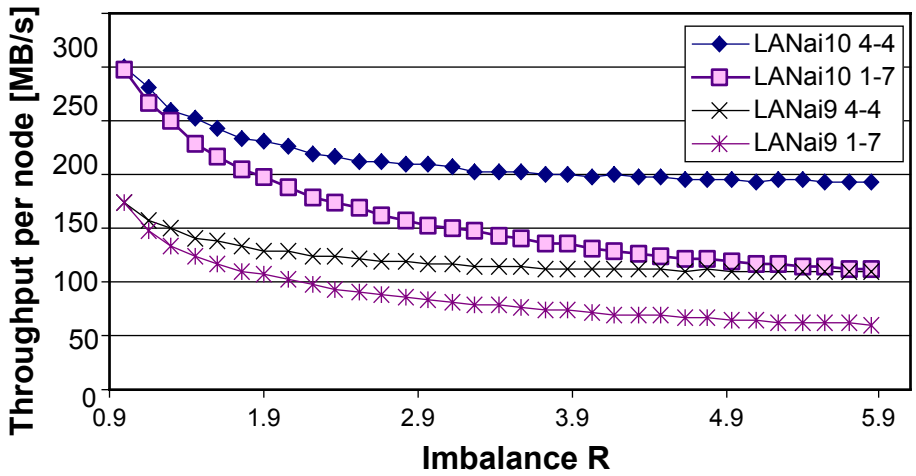


Figure 6-15 The simulated throughput of an 8×8 FED Builder as a function of the imbalance parameter *R*.

In case of the 4-4 division, it is found that for reasonable values of R up to 3 the utilisation drops from 60% for R=1 to 40% for R=3 (two-rail LANai10). An imbalance of R=3, corresponds to $\overline{L_1}$ =3 kB and $\overline{L_2}$ =1 kB. For the 1-7 division, an imbalance of R=1.5 is acceptable, corresponding to $\overline{L_1}$ =2.9 kB and $\overline{L_2}$ =1.9 kB. In conclusion, for super-fragment sizes of 16 kB on average, a throughput of 200 MB/s per node can be achieved, provided no source exceeds 3 kB fragment size on average (or 300 MB/s). The case when one FRL sends a fragment of abnormally large size may occur and the DAQ system must handle such cases. The simulation shows, that the DAQ system can handle these events with negligible degradation of performance because the fragments are transmitted in blocks (see Appendix B.4 for details).

6.5 RU Builder

This section discusses the second stage of the Event Builder; the RU Builder. Two possible implementations are outlined; with Ethernet and Myrinet. Measurements made on test-benches and performance estimates made through simulation show that the networking requirements of the RU Builder can be met with the networking technology available today.

6.5.1 Introduction

The function of the RU Builder (see Section 4.3, "RU Builder") is to assemble the event super-fragments from the 64 FED Builders into full events ready for processing in the filter farm. The main functional elements of the RU Builder are:

- 64 Readout Units (RUs) with large buffers that serve to de-couple the flow of data out of the FED Builders from the flow of data into the RU Builder.
- 64 Builder Units (BUs) where full events are assembled and then sent off to the filter farm
- 1 Event Manager, whose primary function is to adapt the traffic in the RU Builder to the throughput capacity of the BUs and their associated filter farms. It consists of two closely coupled units, the Readout Manager (RM) and the Builder Manager (BM)
- 3 logical networks connecting the storage and processing elements
    a. Builder Control Network (BCN). Connects all units in the system and whose main function is to enable the flow of control information.
    b. Builder Data Network (BDN). The high-throughput network used for the transfer of event data. It connects all the RUs to all the BUs.
    c. Readout Control Network (RCN).Connects the Event Manager to all the RUs. It serves to distribute common Readout information to all the RUs

Each RU Builder must be able to sustain an event rate of up to 12.5 kHz. which corresponds to a data throughput of 12.5 kHz · 16 kB = 200 MB/s for each RU and BU node.

6.5.2 RU Builder Implementation with Ethernet

6.5.2.1 Design

The required 200 MB/s sustained throughput per node can be reached with a two-rail Gigabit Ethernet configuration and an effective throughput above 80% of the wire speed. Hence, each node accommodates two ports (either by using two NICs or by using a single NIC with two ports), each connected to one of the two rails. The network for each rail is provided by a 128 port switch. As an alternative to a single chassis 128 port switch, the switch network can be provided by a multi-stage arrangement of switches with a smaller number of ports in one chassis.

An important design choice concerns the communication protocol. In particular whether to use Ethernet Layer-2 frames or a reliable high level protocol, such as TCP/IP. A comparison of their merits is given in Table 6-6.

Table 6-6 Communication protocol comparison for Ethernet Event Builder.

	Layer-2 frames	**TCP/IP**
host	computer or more basic (e.g. FPGA)	computer
reliability	packet loss if congestion	reliable
zero-copy	yes	no
host CPU usage	low	high [a]
EVB - traffic-shaping	required	maybe
EVB - recovery	at application level	built-in
EVB - latency	medium	high

a. according to an empirical rule, 1 Hz of CPU is required per bps transmitted

As will be shown in this section, it is feasible to operate an EVB close to the wire speed, using Layer-2 frames. However, this necessitates traffic-shaping and a recovery protocol at the application level. Use of TCP/IP is more demanding on host resources and the scaling properties for event builder traffic at high load are still unknown. Preliminary results with TCP/IP are given in Appendix B.3, "Gigabit Ethernet TCP/IP Test-benches" and are summarised below.

6.5.2.2 EVB Protocol for Layer-2 Frames

Experience so far has shown that, typically, large switches do not implement endpoint-to-endpoint flow control through the switch. Hence, packets can get lost due to congestion. With traffic-shaping the congestion can be reduced.

The event building protocol has been presented in general terms in Section 5.3.2.1, "Event Flow Protocols". The specific protocol used here is shown in Figure 6-16. To reduce congestion at the output ports, each BU requests event fragments from each RU sequentially. This is similar to the destination based traffic-shaping, discussed in Section 6.2.3.2. In order to use the links efficiently, several events have to be built concurrently.

The protocol has been augmented to recover from occasional packet loss exploiting the fact that the event building protocol is transaction based, where data are sent on request. The recovery is based on sequence numbers, time-outs and a retry mechanism.

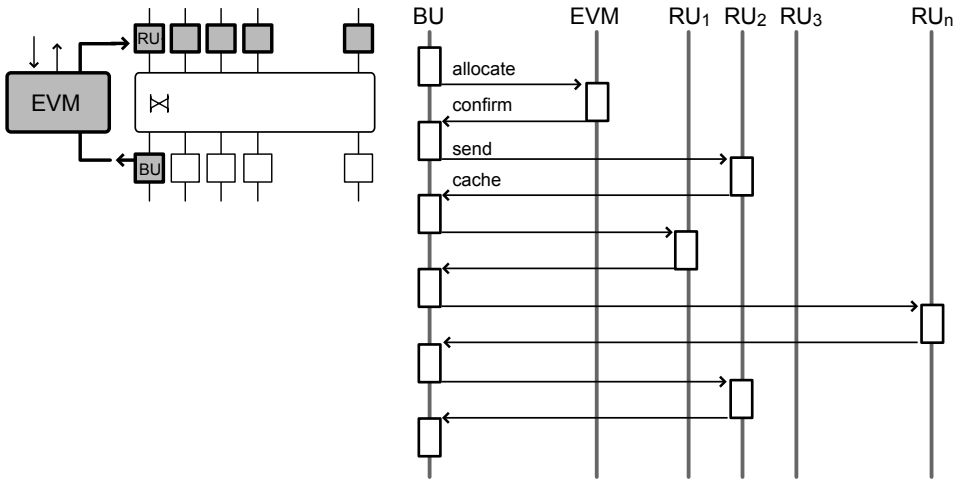


Figure 6-16 Protocol for the Ethernet Event Builder with Layer-2 Frames.

6.5.2.3 Test-bench Results for Layer-2 Frames

In the Event Builder test-bench, PCs are used to emulate BM, RUs and BUs. The BDN and BCN share the same physical Gigabit Ethernet network. The trigger is not emulated and the throughput measured corresponds to the saturation limit.

The application software running on the PCs implements the event building protocol discussed above, using Layer-2 frames to transmit data. A library has been developed to send and receive Layer-2 frames from user space with zero-copy. This uses a modified Linux driver for the NIC.

6.5.2.3.1 One-rail Configuration

A 31×31 Event Builder based on Gigabit Ethernet has been set up. The nodes are 62 PCs emulating the RUs, BUs. In addition, one PC serves as the Event Manager BM. Details on the host configuration can be found in Table 6-7. The Gigabit Ethernet NIC of each host is connected to a FastIron-8000 switch from

Table 6-7 Configuration of the hosts used in the Gigabit Ethernet EVB test-bench.

Configuration element	Equipment type
PC motherboard	SuperMicro 370DLE with ServerWorks LE chipset
CPU	Pentium-III, 750 MHz (RU and BM) or 1 GHz (BU)
PCIbus	64b/66MHz
Gigabit Ethernet NIC	AceNIC from Alteon[6-9]
Operating System	Linux 2.4

Foundry Networks [6-8]. A fully populated FastIron switch comprises 8 modules with 8 Gigabit Ethernet ports each, connected to a crosspoint backplane. Each module contains 2 MB of shared memory to buffer packets. The bandwidth of the memory system and backplane is dimensioned such that the switch is fully non-blocking. For our tests, the switch is configured such that each port can buffer up to 63 packets.

The throughput per node as a function of the fragment size is shown in Figure 6-17. Also shown is the calculated maximum performance taking header sizes into account. The saw-tooth structure reflects the Ethernet MTU size (1500 B) and is due to the fact that no aggregation of event fragments into packets is performed. For fragment sizes up to roughly 8 kB, the throughput is reduced due to the packet handling overhead for control and data messages[1]. For medium fragment sizes in the range 8 kB-20 kB, close to maximum performance is achieved. For fragment sizes larger than 20 kB, significant packet loss leads to inefficiency due to retransmission. Packet loss occurs because the BUs are building a number of events simultaneously (set to 32) and it can happen that several RUs send instantaneously to the same destination port, which will drop the packets if its buffer exceeds 63 packets. For the nominal (super)-fragment size of 16 kB the achieved throughput per node is 115 MB/s.

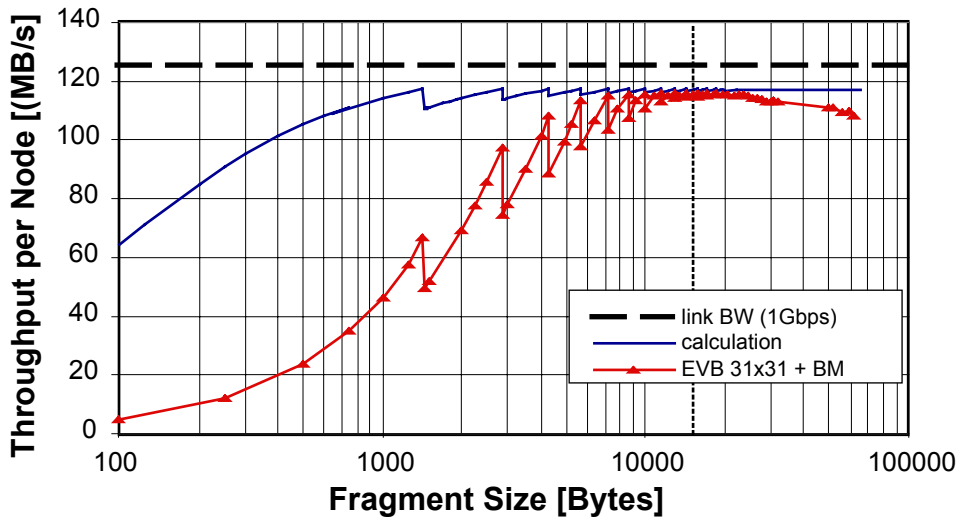


Figure 6-17 Throughput per node versus size for the 31×31 one-rail Gigabit Ethernet EVB.

The performance for variable sizes has been studied by generating fragment sizes according to log-normal distributions. The obtained performance for a number of rms values is presented in Figure 6-18. It can be seen that there is no considerable degradation with respect to the case of fixed size (rms=0).

The size of the Event Builder is varied from 3×3 to 31×31 in Figure 6-19 for the nominal case of variable size fragments with an average of 16 kB and an rms of 8 kB. It can be seen that the throughput per node scales with the size of the configuration.

6.5.2.3.2 Two-rail Configuration

A two-rail 15×15 EVB has been assembled by equipping all nodes with two NICs. Point-to-point measurements over two links show that the transmission rate is almost a factor two higher than with a single

1. A 15×15 setup based on different hardware and software with lower overhead is described in Appendix B.2.1, "Single-chassis Configuration for 15 x 15 Event Builder".

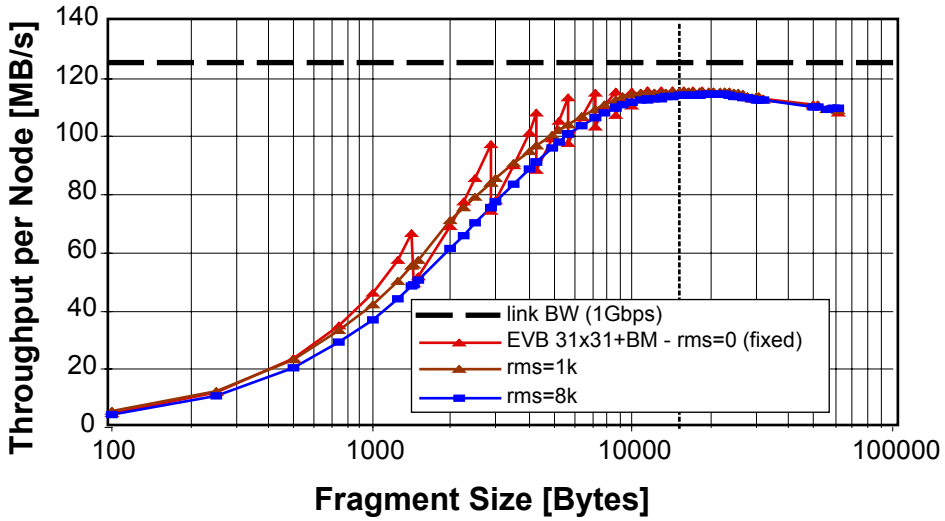


Figure 6-18 Throughput per node versus average size for the 31×31 one-rail Gigabit Ethernet EVB. The fragment sizes are generated according to log-normal distributions.

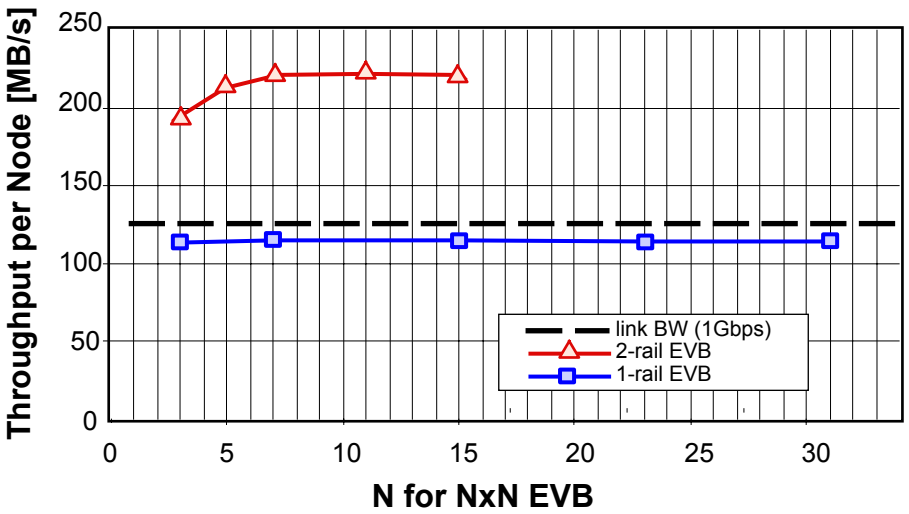


Figure 6-19 Throughput per node versus N×N for one-rail and two-rail Gigabit Ethernet EVB. The fragment sizes are generated according to a log-normal distribution with an average of 16 kB and an rms of 8 kB.

link, indicating that the transfers via the two NICs largely overlap. The resulting EVB performance for nominal fragment sizes is compared to the one-rail configuration in Figure 6-19. The throughput per node scales with the size of the configuration and a value of 220 MB/s is achieved.

6.5.2.3.3 Multi-chassis Configuration

A multi-chassis configuration is an alternative to a single-chassis 128 port Gigabit Ethernet switch. The multi-chassis option is investigated as a possible alternative whose cost-effectiveness will be evaluated at the time of procurement. An Event Builder based on a multi chassis Ethernet switching network has been studied. For details see Appendix B.2.2, "A Multi-chassis Switching Network for a 48¥48 Event Builder". With this configuration it is possible to build large switching networks out of smaller switches by in-

troducing an intermediate layer of switches. However the intermediate layer introduces multiple paths between sources and destinations (RUs and BUs) and this is not allowed with Ethernet layer-2 switching. This problem is solved by using VLANs (IEEE 802.1q), supported by many Ethernet switches. VLANs can effectively divide a switch into multiple smaller independent switches. This can be used in the intermediate switch layer to split the multiple paths over separate networks.

6.5.2.4 Simulation for Layer-2 Frames

A simulation of the Ethernet-based Event Builder has been developed. The model includes the measured overhead of about 10 µs to transmit a packet. The switch model used is based on the FastIron architecture, including the internal buffer memory, output queues and high-performance crosspoint backplane. A comparison of the results with the test-bench measurements is shown in Figure 6-20. Good agreement is obtained for fragment sizes up to 20 kB. The simulation will be extended to include recovery from packet loss.

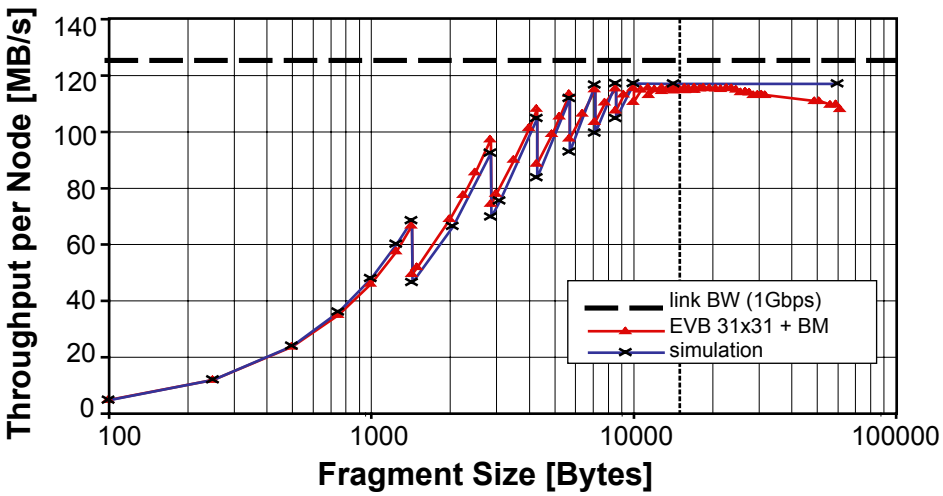


Figure 6-20 Comparison with simulation.

6.5.2.5 Test-bench Results for TCP/IP

The TCP/IP option has been evaluated (see Appendix B.3, "Gigabit Ethernet TCP/IP Test-benches") using the same test-bench. An EVB throughput per node of 75 MB/s has been obtained for variable fragment sizes with an average of 16 kB and an rms of 8 kB. The performance scales linearly with the size of the EVB configuration from 3×3 to 31×31. The throughput corresponds to 85% of the measured TCP/IP point-to-point throughput (88 MB/s). Due to this limitation of the host and/or NIC, the switch load is only about 60%.

With recent hardware[1] a point-to-point TCP/IP performance of 123 MB/s for a single link and 247 MB/s for a dual link can be obtained using “Jumbo frames” (MTU size of 9000 B). Further study is required to investigate if a similar throughput can be achieved with an Event Builder based on such hardware.

1. PC based on a single 2 GHz Pentium-IV Xeon CPU.

6.5.3 RU Builder Implementation with Myrinet.

6.5.3.1 Design

The RU Builder can be implemented with a one-rail configuration based on Myrinet-2000 (which has an effective link speed of 2 Gb/s). A single Clos-128 switch provides the 128 ports of the 64×64 switching network. Shaping the traffic through the switch by operating it as a barrel-shifter is expected to yield close to 100% utilisation. A system without traffic-shaping is easier to implement and operate and has lower latency, but, would yield only about 40% utilisation of the bisection bandwidth due to blocking in the network of multiple stages. Therefore, the barrel shifter system is the preferred solution.

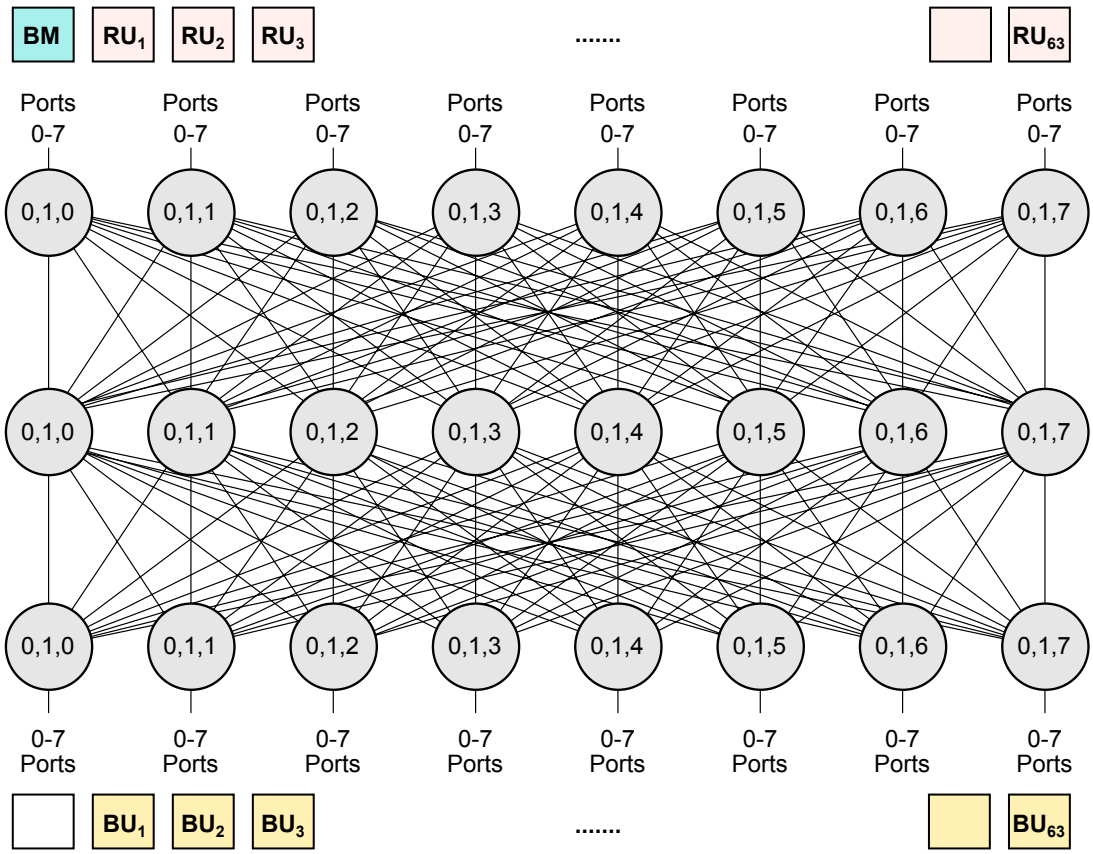


Figure 6-21 Myrinet Clos-128 network used for 63×63 EVB + BM.

A possible configuration of the 63×63 EVB is shown in Figure 6-21, where both BDN and BCN are using the same physical network. The ‘up-side’ of the 64×64 Clos-128 network connects to 63 RUs and the BM, whereas the ‘down-side’ connects to 63 BUs and one dummy node. The dummy node can potentially be used as BU (hence a 63×64 EVB) or as RM in a 63×63 EVB.

6.5.3.2 Barrel-shifter Implementation

As explained in Section 6.2.3.1, one can exploit the characteristics of the event building data traffic pattern to increase the system throughput by arranging the traffic through the switching network as a barrel-shifter.

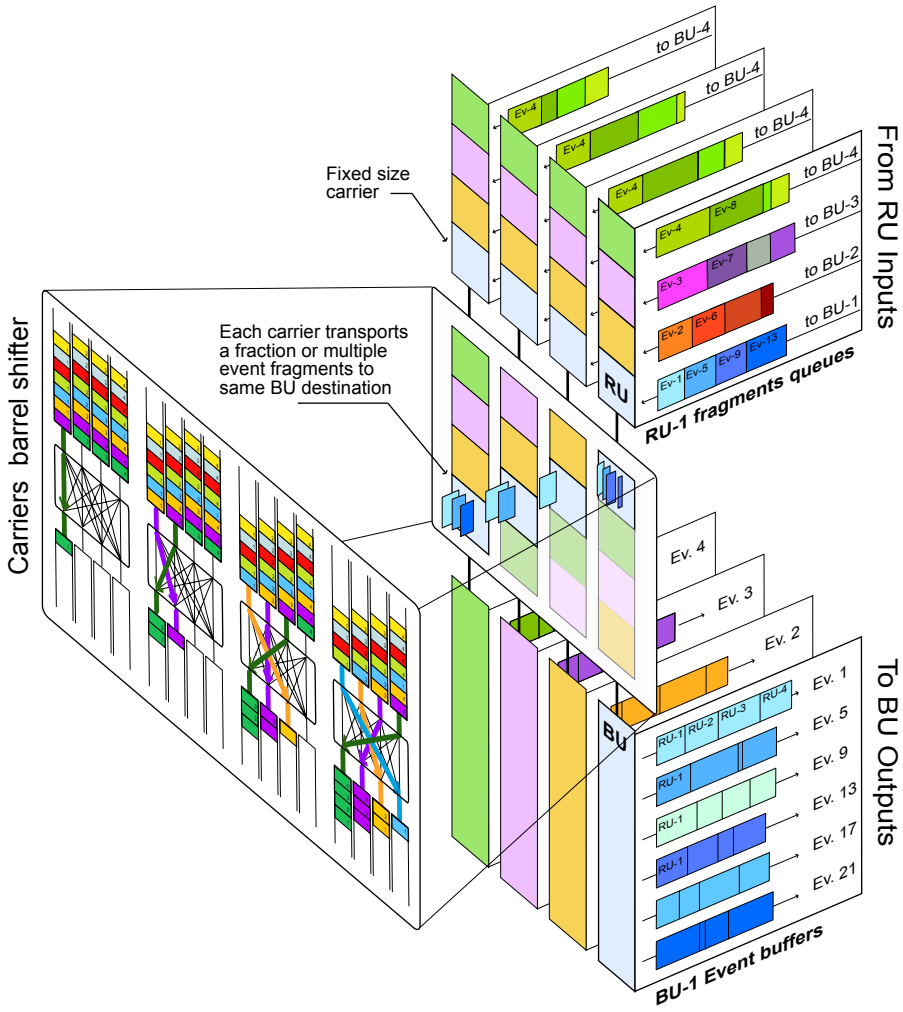


Figure 6-22 Schematic drawing of the Myrinet Barrel Shifter.

The basic idea of the Myrinet barrel-shifter is shown schematically in Figure 6-22. Each host on the source side has a separate queue for each possible destination. The MCP program running on the NIC, cycles through all destinations, copies the data by DMA to the NIC buffer and sends the data in packets to all destinations in turn. The barrel shifter imposes fixed-sized carriers, but event fragments have variable size. In order to deal with this, fragment data are treated as a continuous stream that is cut up into the fixed sized carriers of the barrel shifter. The NIC of the sender cuts the stream into the carriers, and the NIC of the receiver re-assembles the stream of fragment data from the carriers. This transport mechanism is true zero copy, as the host CPU is not used to copy the data. It also supports buffer-loaning.

In a ‘classic’ barrel-shifter, sources are synchronized to emit fixed size carriers in time slots in such a way that no two sources send to the same destination during the same time slot and all sources regularly send to all destinations in a cycle. A central controller is needed to synchronize the time sliced operation of all sources. Note that there is no central controller to apply an external synchronisation in this implementation. Instead, it relies on the back-pressure flow control of Myrinet. After an initial synchronization step utilising the Myrinet network itself, all sources send through mutually exclusive paths to different destinations in a cycle. To keep this synchronization, the sender NICs always emit packets, sending empty carriers when there is no data available for a particular destination.

If the NxN port switch has a composite structure, not all values of n (=[1..N]) are possible for a nxn barrel shifter operation. The possible configurations for a Clos-128 switch built out of 8×8 crossbars, are given in Table 6-8[1].

Table 6-8 Barrel shifter operation in a Clos-128 switch. Possible and impossible values of n with respect to an N×N barrel shifter.

	n
possible	1-10,12,14-16,18,20-21,24-25,28,30,32,35-36,40,42,48-49,56,64
not possible	11,13,17,19,22-23,26-27,29,31,33-34,37-39,41,43-47,50-55,57-63

6.5.3.3 Error Handling

The Myrinet switching fabric does not lose packets. However, packets can be corrupted due to transmission errors. These errors are detected via a CRC. All packets with CRC errors are simply dropped in the receiving NIC, but this leads to an insignificant event loss probability of about $O(10^{-8})$ with a Myrinet bit error rate below 10^{-15} and an event size of 10^7 bits.

There is a potential for buffer overrun in the receiving NIC, due to the limited amount of memory in the NIC (typically 128 packets). The receiving NIC has to drop the incoming packet in this case, as otherwise the barrel shifter synchronization would be lost. However, this buffer overrun is unlikely to occur in practice because the PCI bandwidth is about a factor two higher than the Myrinet link speed and the amount of buffers available in the host can be made large.

6.5.3.4 Test-bench Results

The test-bench configuration is based on a 32×32 switching network with Myrinet-2000 hardware. It comprises a half-populated Clos-128 switch connecting 64 PC nodes, acting as emulators of the RUs, BUs and BM. The BDN and BCN share the same physical network. The trigger is not emulated and the throughput measured corresponds to the saturation limit. Details of the host configuration can be found in Table 6-9.

Table 6-9 Configuration of the hosts used in the Myrinet EVB test-bench.

Configuration element	Equipment type
PC motherboard	SuperMicro 370DLE with ServerWorks LE chipset
CPU	Pentium-III, 750 MHz (RU and BM) or 1 GHz (BU)
PCIbus	64b/66MHz
Myrinet NIC	M3S-PCI64B. LANai9 based with 2 MByte of local SRAM.
Operating System	Linux 2.4

1. A nxn barrel shifter is possible if there exists a product n=n1·n2 where n1=[1..8] is number of crossbars used in up-side layer and n2=[1..8] is number of ports used per crossbar in this layer. The number of crossbars used in the up-side and down-side layers are equal. The routing is such that the crossbar in the intermediate layer is given by the destination port of the crossbar in the down-side layer.

The software running on the PC nodes implements the event building protocol presented in Section 5.3.2.1, "Event Flow Protocols".

The Myrinet NIC in each host is based on LANai9. A custom MCP firmware implements the barrel-shifter for the traffic in the up-down (RU/BM to BU) direction. Note that no traffic-shaping is applied in the down-up (BU to RU/BM) direction, used for small control messages. All measurements were done with the barrel-shifter carrier size set to 4 kB, corresponding to a transfer time of about 17 µs.

In the following, the results obtained with the test-bench are summarized. The configuration tested corresponds to a 31x31 EVB with BM, i.e. half the size of the full system described above (Section 6.5.3.1). The throughput per node as a function of fragment size is shown in Figure 6-23. In the case of fixed size fragments, it reaches a plateau of about 230 MB/s for sizes above 6 kB. This corresponds to the maximum performance expected, taking header sizes into account. If needed, the plateau can be reached for lower sizes by aggregating the event request control messages. The performance for variable size event fragments, is also shown in Figure 6-23, and it can be seen that the throughput per node reduces with increasing rms, as expected.

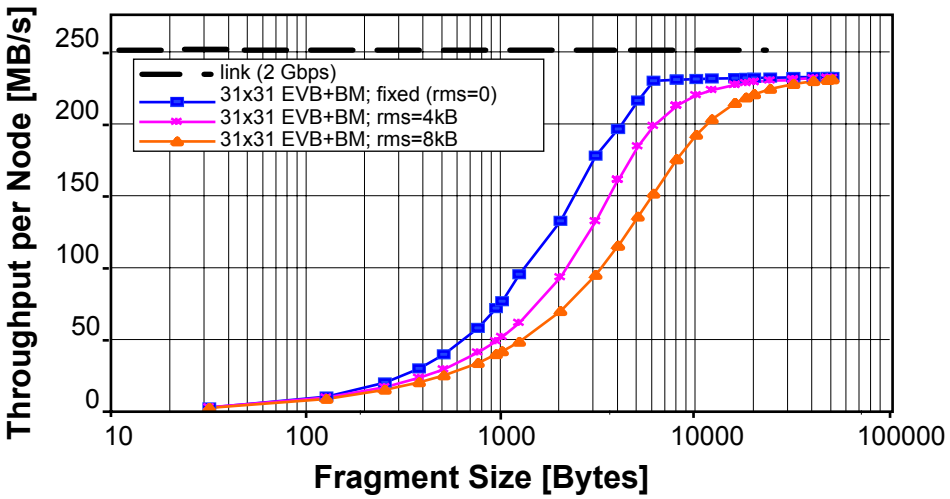


Figure 6-23 Throughput per node versus average fragment size of the Myrinet 31×31 EVB. The series with variable sizes corresponds to log normal distributions.

In order to investigate the scaling of the performance, network configurations of 8×8, 16×16, 24×24 and 32×32 have been tested. As one port of the barrel shifter is connected to the BM, they correspond to a 7×7, 15×15, 23×23 and 31×31 EVB, respectively. The results of these measurements are presented in Figure 6-24. The throughput per node for a NxN EVB varies as N/N+1, because N out of N+1 cycles of the barrel shifter are used to transfer event data. This leads to an efficiency of 97% for the tested 31x31 EVB and to 98% for the full 63×63 configuration. For the nominal case of variable size fragments with an average size of 16 kB and a rms of 8 kB, a throughput per node of about 215 MB/s is achieved.

6.5.3.5 Simulation Results

A detailed simulation of the event builder has been developed. Details of the simulation can be found in Appendix B.4, "Simulation of Myrinet Event Builder". A comparison of results from the simulation with the test-bench measurements presented in the previous subsection is shown in Figure 6-25. The throughput per node is shown as a function of the fragment size. The symbols represent the data points and the

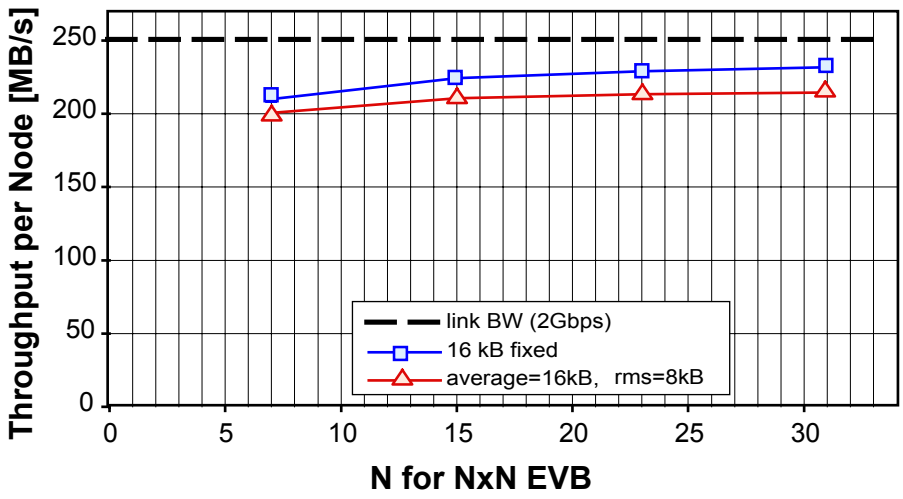


Figure 6-24 Throughput per node versus N×N for Myrinet EVB.

two bands are obtained by simulation. The bands differ in the assumptions made for the performance of the NIC card (see Appendix B.4 for details). The points are enclosed by the two bands showing a good agreement between the test-bench data and the simulation.

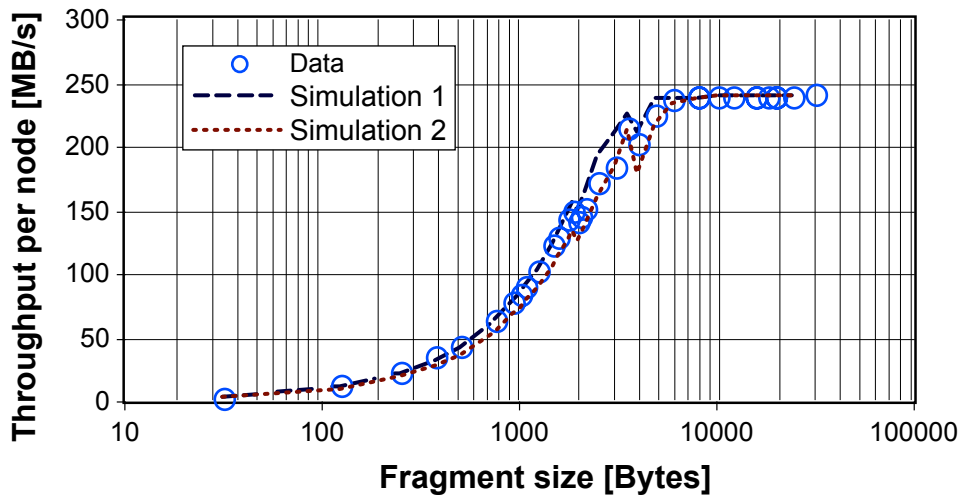


Figure 6-25 Throughput per node as a function of the fragment size for a 31×31 EVB with BM and LANai9 hardware. The bands represent the simulation result for two different NIC performance assumptions. The symbols show the data points measured with the test-bench.

Starting from the accurate description of the current LANai9 hardware, a prediction is made for the Myrinet LANai10 hardware. The result for a 64×64 network configuration based on a one-rail network with LANai10 hardware including a BM is shown in Figure 6-26. At the working point of 16 kB fragment size, a throughput of 236 MB/s is expected for fixed sizes and is only slightly reduced to 231 MB/s for variable sizes with an rms of 6 kB. This provides a comfortable safety margin for the required throughput of 200 MB/s for a trigger rate of 12.5 kHz per RU Builder.

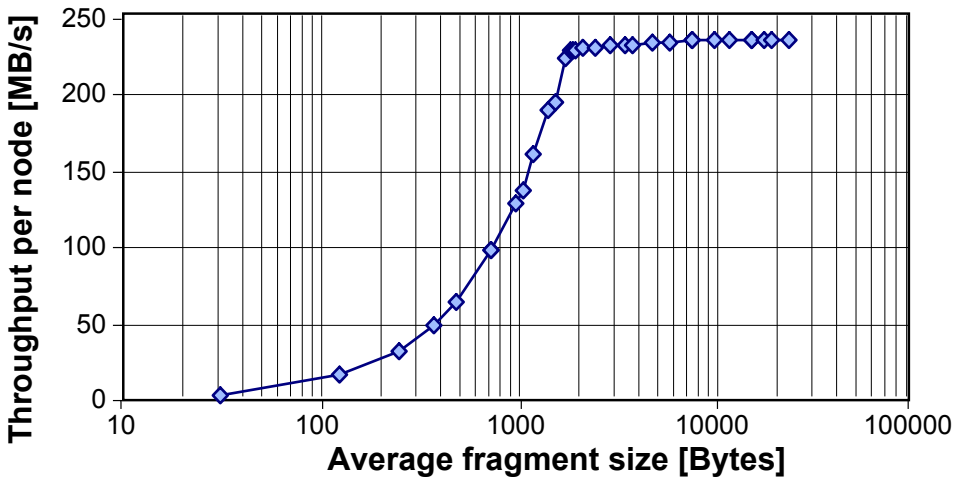


Figure 6-26 Simulated throughput as a function of the fragment size for a 63×63 EVB with BM and LANai10 hardware in a one-rail network.

6.6 Readout Control Network

The Readout Control Network (RCN) is a component of the RU Builder. It is used to distribute readout information that is common to all the RUs in the RU Builder. This information includes the association of Event-ID to the Level-1 event number (Section 4.3, "RU Builder").

6.6.1 Overview of RCN Requirements

The RCN is the logical network used by the Event Manager to distribute control information to all RUs in a single RU Builder (partition). The main requirements are therefore:

- The RCN must handle 64 nodes (1 RM and 63 RUs).
- The distribution of the RCN messages must be reliable.
- Network support for broadcast/multicast may be used to reduce the number of messages sent by the RM because the data content of the RCN message is identical for all RUs.
- The maximum event rate in a RU Builder is 12.5 kHz, which correspond to an average time between events of 80 µs. It is difficult to broadcast a message reliably below 1 ms with current commercial network technologies on a PC. However, the broadcast rate can be reduced by packing several messages in each broadcast. Packing of several tens of messages per broadcast would reduce the rate sufficiently.
- The minimum information to be sent for each event is the “Event_ID/event number” pair, which requires 8 bytes. It may be desirable to send more information such as the Bunch crossing number. The design is not yet finalised at this level.
- Assuming that 12 bytes per event are distributed per broadcast, then the effective bandwidth needed (discarding overheads) would be 12 B × 12.5 kHz = ~1.2 Mbps which corresponds to 0.12% of the bandwidth available on Gigabit Ethernet. Peer-to-peer communication between the EVM and the RUs would increase the RCN traffic by a factor 63, requiring an effective bandwidth of 63 × 1.2 Mbps or 8% of the Gigabit Ethernet bandwidth at the EVM port.

- The latency requirement on delivery of the Event_ID is relaxed because the Event-IDs are assigned to super-fragments only after these have been buffered in the RU memory (see Section 4.3.7, "RU Builder Data Flow"). With a RU buffer depth of O(512 MB), corresponding to 32000 super-fragments of 16 kB, it is possible to accept latencies of several thousand triggers. It is therefore not a problem to pack several tens of RCN messages per broadcast.
- The network must be consistent with partitioning of the RU Builder. This means that it must be possible to send the RCN messages to only a subset of the RUs either in multicast or peer-to-peer mode.

6.6.2 Implementation

RCN can be implemented as a Gigabit Ethernet physical network distinct from the Builder network or it can be integrated with the BCN and BDN in one builder network as described in Section 5.4.2, "Event Manager Networks". This implementation choice has not yet been made.

If RCN is implemented on a distinct Gigabit Ethernet network, it is possible to use multicast over UDP for distribution of the RCN messages to the RUs. A reliable broadcast protocol has been developed to satisfy the requirement of reliable delivery of RCN messages.

6.6.3 Reliable Broadcast Protocol for the RCN

The distribution of the RCN control information must be totally reliable, which means no loss of information and guaranteed latency. There are many broadcast/multicast protocols proposed or implemented. Most of them aim to serve multimedia (audio/visual) data streams and are not totally reliable. Some of them guarantee total reliability, but are not fast enough, because the main applications of these are file transfer and data mirroring.

The proposed RCN Protocol (RCNP), copes with both total reliability and low (O(ms)) latency. It is a combination of popular techniques and utilizes knowledge of the network topology and usage of the RCN to reduce the latency.

Key features of the RCNP are:

- It is NACK-based
- It includes packet loss detection on the receiver side
- It has redundancy to reduce packet repairs
- It utilizes acknowledgments (ACKs) from a few clients as a pacemaker.

ACK-based mechanisms, in which each receiver sends back an ACK packet for every data packet, are not applicable given the message rate and latency in the RU Builders. A flat network would be flooded with ACK packets while a tree-based ACK mechanism introduces a significant latency. This problem is more severe with slower systems like Fast Ethernet. In a NACK(Negative ACK)-based mechanism, a receiver sends back a repair request only when it detects packet losses or unrecoverable corruptions.

Data corruptions inside a packet are detected utilizing CRC (cyclic redundancy check) words included in each packet. Packet losses are detected as jumps between the sequential numbers of adjacent packets. The sequential number may be the L1 trigger number in our case. With these mechanisms the data can already be transmitted reliably. However, the process of repairing packets is expensive in terms of latency. There-

fore a Forward-Error Correction (FEC) technique is introduced to keep the packet retransmissions as rare as possible. In this scheme, the RM sends redundant data and any receiver can reconstruct the original data allowing for packet losses up to the redundancy limit. A simple example is to add one extra packet, that is the XOR of original packets, for every three data packets (3+1 FEC). In that case the receiver misses the data only when there are two or more packets lost from these four packets. Any fraction of redundancy is possible regarding the network reliability.

The last point is congestion control. There is no means to detect buffer overflow on the receiver side, because hardware flow control is not possible (or not practical) in broadcasts. Keeping the transfer rate below a level acceptable for the RUs is necessary to avoid buffer overflows and resulting NACK floods. A possible method is that the EVM waits for a ACK packet from only one or a few RUs and sends out the next packet after receiving it. Using this method the EVM applies an optimized delay between sends.

The RCNP, described above, is illustrated in Figure 6-27. In case of no packet loss or corruption (left side), trigger information is sent as a sequence of packets, which consists of three original data and a redundancy packet for 3+1 FEC. One or several preconfigured RUs send back an ACK packet. Then, the EVM starts sending the next sequence. If packet losses occur (right side), the incomplete sequence is detected when a packet for the next (or later) sequence is received by a RU. The RU sends back a NACK packet to the EVM and the EVM sends the lost packet sequence again as a broadcast.

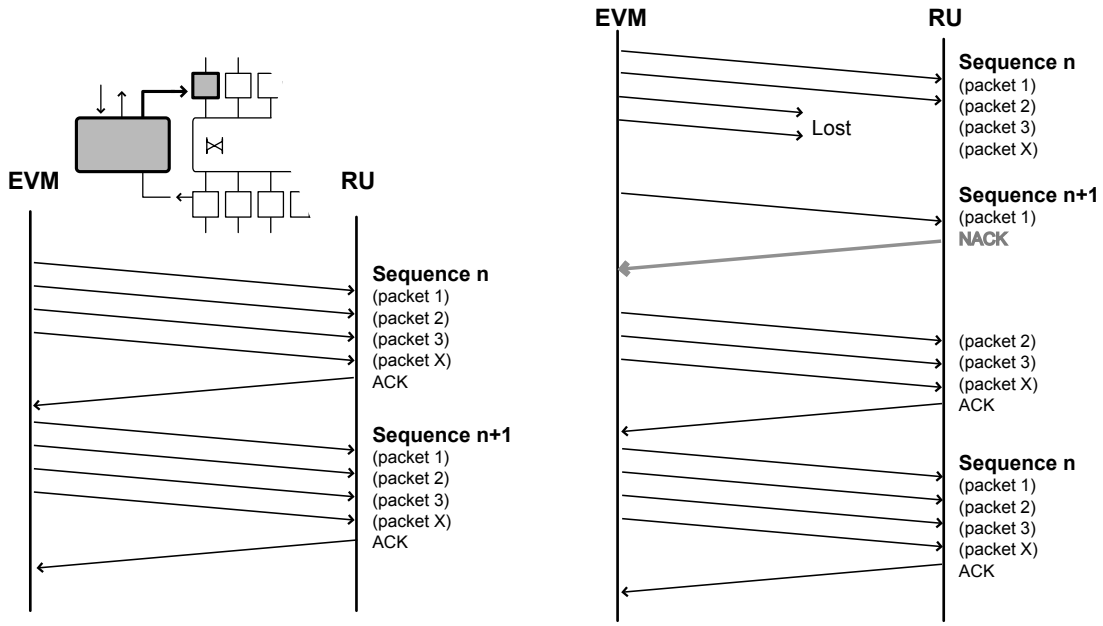


Figure 6-27 RCNP - a protocol for the RCN.

6.6.4 RCN Prototype and Simulation

Prototyping and simulation studies of the RCN have been performed in conjunction with the EVM prototyping and simulation described in Section 5.4.5, "Prototypes for the Event Manager"**.** The studies have not yet reached a final conclusion, but they have reached a level, where there is no doubt that the RCN can be implemented using one or more of the possible implementations under study.

6.7 Full Event Builder

This section discusses issues related to the full EVB system of the combined FED Builder and RU Builder stages. So far, studies of the full system have been limited to simulation.

6.7.1 Simulation

The full system with FED Builder and RU Builder combined has not been simulated. However, both stages have been studied separately considering the missing element as a pure sink or source, respectively. The assumption that both stages are decoupled is justified if the memory in the RU can absorb the fluctuations of the output of the FED Builder. The overall EVB performance is clearly determined by the minimum of the performance of the FED Builder and RU Builder stages.

A prediction is made for the maximum trigger rate as a function of the number of RU Builders, assuming balanced and uncorrelated inputs (see Figure 6-28). The capacity of the set of RU Builders increases linearly with the number of RU Builders (by construction) and will hold irrespective of the technology used for the RU Builder. In Figure 6-28, a RU Builder based on Myrinet with a one-rail and applying a barrel shifter is assumed (described in Section 6.5.3, "RU Builder Implementation with Myrinet."). Both a one-rail (LANai9) and two-rail (LANai10) scenario are simulated for the FED Builder. For a single RU Builder, a one-rail FED Builder will provide enough performance to reach the 12.5 kHz trigger rate. The FED Builder performance does not scale linearly because of the blocking in the Myrinet switch (see Section 6.3.2.2, "Myrinet for Event Building"). For more RU Builders, only the two-rail based FED Builder can match the capacity of the RU Builders. A maximum trigger rate of 110 kHz can then be reached with the full EVB system.

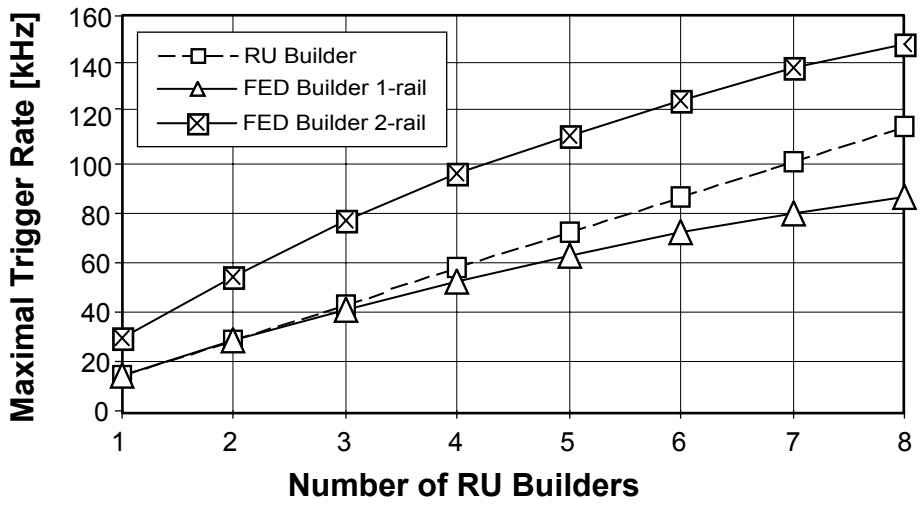


Figure 6-28 The maximum trigger rate as a function of the number of RU Builders in the DAQ system for a one- and two-rail FED Builder and a one-rail Myrinet Barrel Shifter RU Builder. Inputs are assumed to be balanced and uncorrelated. Inputs are generated according to a log-normal distribution with an average of 2 kB (16 kB) and an rms of 2 kB (16 kB / $\sqrt{8}$), for the FED Builder (RU Builder), respectively.

6.8 Summary and Outlook

This chapter described the design and possible implementations of the Event Builder networks. A two-stage EVB design with 64 8×8 switches in the FED Builder stage and 8 64×64 switches in the RU Builder stage has been considered. A major advantage of this architecture compared to a single 512×512 monolithic switching network is that the RU and BU nodes need to operate at a rate of only 12.5 kHz with average fragment sizes of 16 kB, rather than at 100 kHz with average fragment sizes of 2 kB.

A possible configuration of the FED Builder stage connecting the total of ~630 FEDs in the CMS readout has been outlined. The proposed assignment of FEDs to FED Builders provides reasonable balanced inputs to the second stage of the Event Builder.

The two-stage Event Builder design, consisting of FED Builder and RU Builder, imposes very different requirements on the two builder networks. Two networking technologies, Ethernet and Myrinet, have been considered for each of the two stages.

6.8.1 FED Builder

The 8×8 FED Builder is required to handle a 100 kHz rate of fragments with sizes ranging from a few hundred bytes to a few kilobytes and combine them into super-fragments.

The results from the FED Builder prototype combined with simulation studies show that the FED Builder can be implemented with a two-rail Myrinet configuration and LANai10 based NICs. It can be designed to deliver super-fragments in sequential order to the RU at the required 12.5 kHz rate with a low overhead on the RU.

The difficulty of meeting the requirements with Ethernet equipment available in 2002 was judged too high to warrant a prototyping effort.

6.8.2 RU Builder

The requirement of the 64×64 RU Builder with a 12.5 kHz rate of 16 kB fragments can be met with Myrinet and most likely also with Ethernet technology. The equipment available for prototyping allowed testing of RU Builders of half the required size.

6.8.2.1 Myrinet

The measurements on the half-sized RU Builder (32×32) proved that Myrinet can perform event building in barrel shifter mode at near 100% utilization of the network bandwidth. The simulation model describes the equipment accurately and shows good agreement between measured and simulated results. Simulation estimates of the performance of a full RU Builder show that a one-rail Myrinet configuration will meet the requirements of a full RU Builder.

6.8.2.2 Gigabit Ethernet with Layer-2 Frames

Tests have been done with a one-rail 32×32 and a two-rail 16×16 Gigabit Ethernet RU Builder. These configurations corresponded to half and one quarter of the full size, respectively. Both measurements show that with a lightweight protocol using Layer-2 frames it is possible to operate in event building

mode near 100% utilization of the network bandwidth. The performance is critically dependant on the amount of internal buffer memory of the switch to ensure a low packet loss probability. A simple simulation model, which does not yet include the recovery from packet loss, shows good agreement between measurements and predicted throughput. It is plausible, but not proven, that a two-rail Gigabit Ethernet Event Builder with Layer-2 frames will meet the requirement of a full RU Builder. This solution would rely on standard hardware, but requires non-standard software and application level traffic-shaping and error recovery. Only the TCP/IP based solution described below would bring the full advantage of standard Ethernet hardware and software to the RU Builder.

6.8.2.3 Gigabit Ethernet with TCP/IP

The equipment available for prototyping could operate TCP/IP only at 70% of the available bandwidth in point-to-point communication and at 60% of the available bandwidth in event building with a (32×32) one-rail and a half-sized RU Builder. The result is encouraging, but inconclusive, because the PCs and their NICs are only loading the switch at 60%. New models of PCs and NICs have been measured to operate at near 100% of available bandwidth over a two-rail point-to-point TCP/IP communication. However, these systems were not available in quantities that allowed RU Builder testing.

The measurement made so far shows that TCP/IP over Gigabit Ethernet might meet the RU Builder throughput requirements when operated on PCs with more than 2 GHz Pentium-IV Xeon processors and appropriate NICs. It is still an open question whether TCP/IP based event building will work when the switch is operated closer to full capacity. It is difficult to simulate the TCP/IP traffic accurately, because of the complexity and because of missing information on the inner operation of the switches, the NIC and system level TCP/IP software. A complete proof of the TCP/IP option can probably only be obtained through prototyping with a full-scale one-rail RU Builder using equipment that can drive TCP/IP at full rate in point-to-point mode.

6.8.3 Readout Control Network

The RCN can be implemented as a Gigabit Ethernet physical network distinct from the Builder network or it can be integrated with the Builder network. Although the final implementation choice has not yet been made, the problem is reduced to finding the optimal out of several possible solutions.

6.9 Conclusion

It has been demonstrated that the EVB can be implemented with technology available in 2003. In the case of the FED Builder the requirements can be met with Myrinet and in the case of the RU Builder they can be met with both Myrinet and Gigabit Ethernet.

The EVB architecture is flexible enough to be implemented with other network technologies as well. Potential candidates for future feasibility studies include 10-Gigabit Ethernet and Infiniband [6-10]. The final choice of technology will be made based on numerous considerations including cost, ease of operation, maintenance and ease of upgrades.

6.10 References

6-1 M. Bellato et al., “The CMS Event Builder Demonstrators based on GigaEthernet Switched Network”, in Proceeding of the CHEP2000 Conference, Padova, Italy, 7-11 Feb. 1999, pp. 303-306.

6-2 G. Antchev et al., “The CMS Event Builder Demonstrator and results with Myrinet”, Comput. Phys. Commun., 140 (2001) 130.

6-3 E. Barsotti, A. Booth and M. Bowden, “Effects of Various Event Building Techniques on Data Acquisition System Architectures”, Santa Fe Computing 1990:82-101, http://fnalpubs.fnal.gov/archive/1990/conf/Conf-90-061.pdf

6-4 N.J. Boden et al., “Myrinet - A Gigabit per Second Local Area Network”, IEEE-Micro, February 1995, Vol. 15, No. 1, pp. 29-36.

6-5 VITA Standards Organization, “Myrinet-on-VME protocol specification”, ANSI/VITA 26-1998.

6-6 Myrinet products from Myricom, Inc.,Arcadia, CA, USA, see http://www.myri.com

6-7 O. Kodolova, I. Tomalin and P. Yepes, “Expected Data Rates from the Silicon Strip Tracker”, CMS Note-2002/047.

6-8 FastIron switch from Foundry Networks, see http://www.foundry.com

6-9 AceNIC from Alteon Corp., see http://www.alteon.com

6-10 Infiniband, see http://www.inifinibandta.org

7 Readout Column

The previous chapters of Part 2 have dealt with data flow from the standpoint of event building and have discussed the related flow control. This chapter discusses the components which provide the functionality to read out the front-end electronics, transfer the data from the underground experimental area to the surface, and buffer them before the complete event can be assembled. These components are collectively referred to as a Readout Column. The next section gives an overview of the structure of the Readout Column. Section 7.2 summarizes the readout requirements of the various sub-detectors. A functional description of the column components can be found in Sections 7.3, 7.4 and 7.5. Section 7.6 explains how the data flow in the column is controlled via the Trigger-Throttling System (TTS). In Section 7.7 a description of the prototypes and test-benches built so far, and a summary of the measurements obtained with them, are given. A summary section concludes the chapter.

7.1 Introduction

The Readout Column is a vertical slice of the DAQ system which consists of a Front-End Driver (FED), reading out the Front-End Systems, and feeding these data into a Front-end Readout Link (FRL). The FRL output is carried to the surface part of the FED Builder, where it is assembled into a larger data structure and stored in a Readout Unit (RU) (see Figure 7-1).

The FES and the FED, up to the interface with the DAQ, are subdetector-specific components.There are approximately 10,000 FES modules in the CMS readout, all located in the underground area. For every event accepted by the Level-1 Trigger, several Front-End Systems send data to a FED. The Front-End System (FES), and at least part of the FED, operate synchronously with the LHC clock. They are driven by fast synchronous signals distributed by the Trigger Timing and Control system (TTC) [7-1]. Examples for these signals are the Level-1 Trigger, the 40 MHz LHC clock (BX) and reset signals to synchronize the various Front-End Systems amongst each other. The transfer of data from the FED to the FRL is not synchronized to the LHC clock, and, therefore, the rest of the Readout Column operates asynchronously.

In order to provide common handling of event fragments from the different subdetectors, the interface between the FED and the DAQ system has been standardized in the S-LINK64 specification [7-2]. The S-LINK64 transfers data from the FED to the Front-end Readout Link (FRL). To reduce the number of FED Builder input ports, and to balance input fragment sizes for efficient operation of the FED Builder switch, event fragments from up to two FEDs can be merged into a single data block in the FRL. The FED Builder switch itself is located on the surface, at a distance of about 200 m from the FRL. A long optical link is used to transfer data from the underground experimental area to the surface. The FED Builder Input (FBI) drives this link using the FED Builder communication protocol.

Event data blocks from the FRLs are organized in the RU memory to form an event super-fragment containing data from several FEDs. The RU is the largest data buffer in the Readout Column. This large buffer is needed to decouple the final event building from the readout operations performed in the Readout Column. From the RU, data are transferred, via the RU Builder, to the Builder Unit (BU), which assembles event super-fragments from many RUs to form entire events. Subsequently, these events are transferred to a Filter Column (FC) where they are processed in order to obtain the High-Level Trigger decision (HLT).

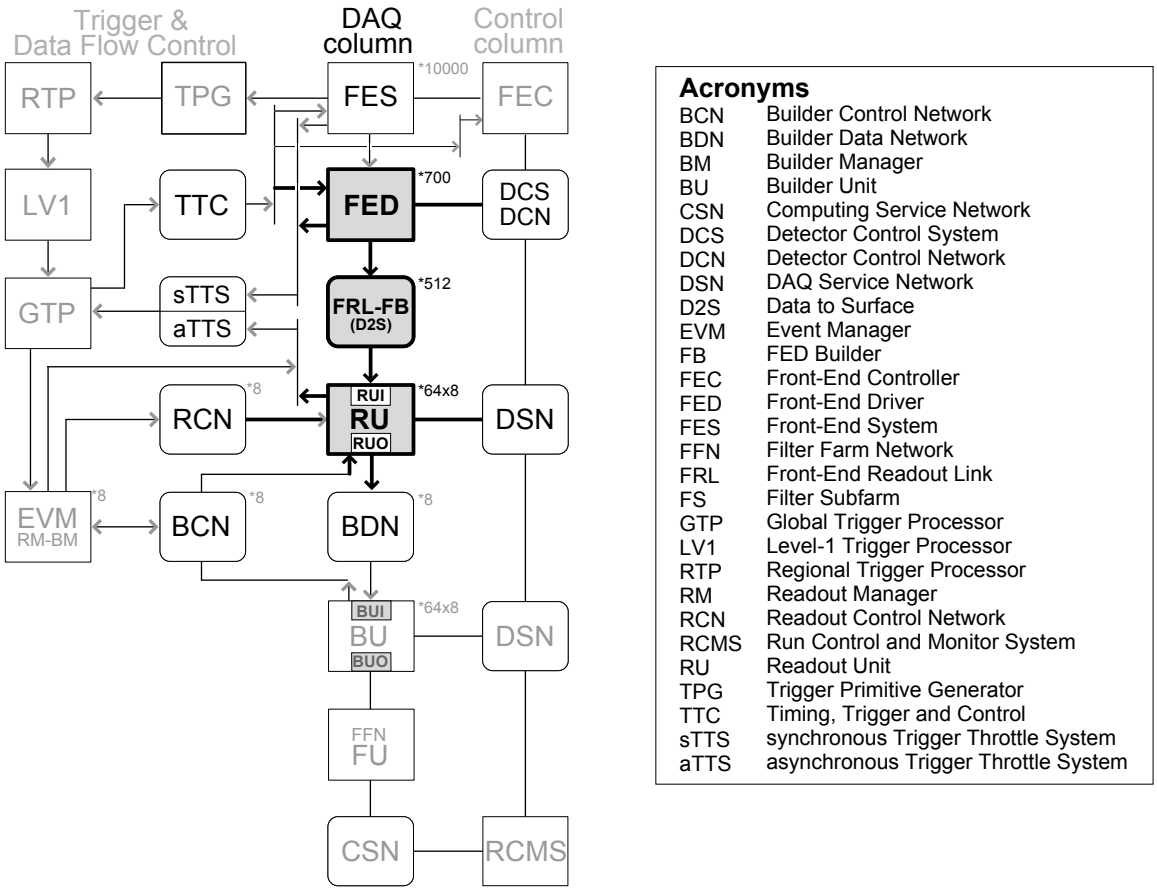


Figure 7-1 Location of the readout column in the CMS DAQ. Also shown are the main elements in the Readout Column, namely the FED, the FED Builder and the Readout Unit. The FED Builder contains the data link to the surface (D2S).

7.2 Detector Readout Requirements

This section summarizes the requirements of the sub-detectors relevant for the DAQ system. For an exhaustive description of each CMS sub-detector, the reader is referred to the corresponding Technical Design Reports [7-3][7-4].

7.2.1 Requirements Common to all Sub-detectors

This part describes the generic functions performed by each front-end electronics. Although their implementation is specific, they all follow a common logical model described in [7-5] and illustrated in Figure 7-2.

All sub-detectors generate analog signals which are processed in several stages by the front-end electronics. The signal path then splits into one for the DAQ system and another for the trigger system[1]. Trigger

1. Not all sub-detectors send information to the Level-1 Trigger system.

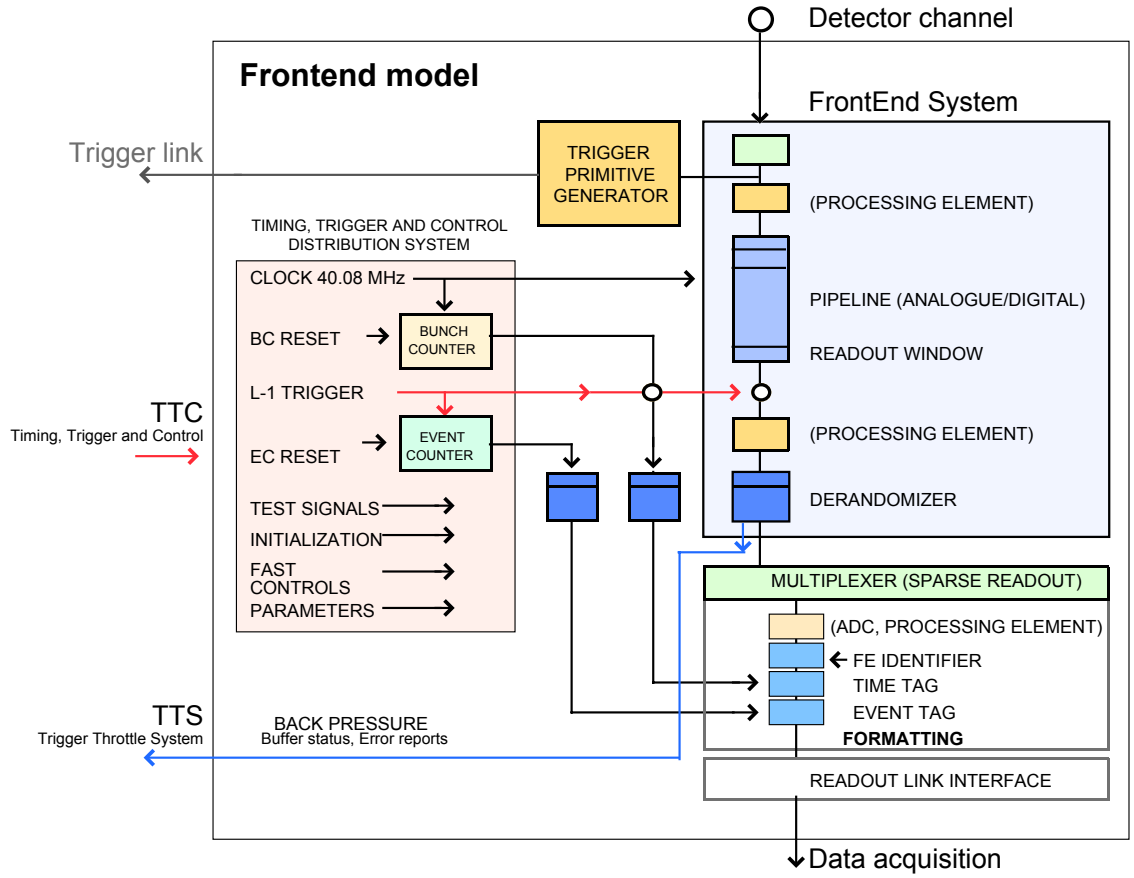


Figure 7-2 CMS front-end common logical model.

information is transmitted synchronously with the LHC machine clock via a dedicated link to the trigger logic. There it is processed to obtain the Level-1 Trigger decision.

On the DAQ path, data must be buffered during the processing time of the Level-1 Trigger. Since the trigger system operates synchronously to the LHC machine clock, the arrival time of the trigger decision at the Front-End System is fixed. Therefore the data buffer is implemented with a pipeline (analog or digital) driven by the LHC clock. The length of the pipeline is calibrated to the latency of arrival of the Level-1 decision signal. On a positive trigger decision data is written into a “derandomizer” buffer which has the functionality of a FIFO.

For every Level-1 Trigger the contents of several derandomizer buffers are merged to form event fragments. In order to identify the event fragment and to allow for consistency checks in the DAQ system, the header contains the event number and the bunch-crossing identifier. Both numbers are generated locally in the Trigger Timing and Control receiver chip (TTCrx). The formatted event fragments are transmitted into the Readout Column.

7.2.2 Front-end Buffer Overflow Management

Space and power constraints in the front-end electronics only allow to implement small derandomizer buffers. Since the Level-1 Trigger is distributed statistically in time, and for each sub-detector there is a minimal derandomizer specific readout time (called service time), the buffer can in principle overflow.

Table 7-1 lists the service time and buffer sizes for each of the various sub-detectors. In case of an overflow, most sub-detectors drop the event data and continue to transmit empty events until the buffers are ready to receive new event fragments. This allows to retain synchronization with the DAQ system and hence avoids lengthy recovery procedures that would deteriorate the DAQ efficiency.

In order to minimize the probability of a buffer overflow, rules which limit the number of triggers in a fixed time interval are applied in the Global Trigger Processor. Table 7-1 summarizes the result of simulations (if available) for the various sub-detectors, assuming different trigger rules and 100 kHz trigger rate. As can be seen, buffer overflows can not be avoided completely by introducing trigger rules. Therefore, where possible, the Global Trigger Logic (GTL) emulates the state of the front-end buffers in order to suppress triggers which would cause a buffer overflow.

Finally, the Front-End Systems assert fast signals to throttle the trigger via the synchronous Trigger-Throttling System (sTTS) described in Section 7.6.

Table 7-1 Front-end and overflow management.

Detector	Buffer size (events)	Service time (us)	Overflow probability w/o Trigger rule	Front-end Emulation	Derandomizer Warning signals	Trigger rules
Pixel	16	N/A	< 2.5%[a]	not possible (all the Front-ends are independent)	status bit in the data stream (3 levels of warning)	empty orbit every second and at least 2 empty BX between consecutive triggers
Tracker	10	7	3.10^{-4} [b]	Hardware emulator based on APV core [7-7]	no	at least 2 empty BX between consecutive triggers
Preshower	10	5.4	2.10^{-5} [b]	simple state machine	yes	no more than N+7 triggers during N x service time
ECAL	25	7.1	very low	easy emulation if needed	no (considered to be too slow)	none
HCAL	44	4	very low	not needed	not needed	no more than 22 triggers per orbit
Muons CSC	~85	0.3/3/26	0.3 events/ 24 hours	no	yes	at least 2 empty BX between consecutive triggers
Muons RPC	64	8	very low	not possible (occupancy dependent)	yes	none
Muons DT	16	2	very low	not possible (occupancy dependent)	yes	none

a. on-going simulation work.

b. extracted from [7-6].

7.2.3 Sub-detector Readout Parameters

In this paragraph the readout parameters of the various sub-detectors are summarized. A comprehensive description of the detector geometry, the readout chain and the data format are available from [7-8] and from Chapter 14, "Detector Simulation and Reconstruction". Table 7-2 lists the number of sub-detector

Table 7-2 Detector readout chain comparative overview.

Detector	Number of channels	Number of FE chips	Number of detector data links	Number of data sources	Number of DAQ links
Pixel	~44M	15840	1504	38	38
Tracker	~9.3M	~72k	~36k	440	272 (merged)
Preshower	144384	4512	1128	47	47
ECAL	82728	~17k	~9k	54 (DCC)	54
HCAL	9072	9072	3072	32 (DCC)	32
Muons CSC	~500k	~76k	540	8 (DCC)	8
Muons RPC	192k	~8.6k	636(732)[a]	6 (DCC)	6
Muons DT	195k	48820	60	5 (DCC)	5
Global trigger	N/A	N/A	N/A	4	4
DT track finder	N/A	N/A	N/A	1	1
CSC track finder	N/A	N/A	N/A	1	1
Total	**~54.5M**	--	--	**636**	**468**

a. The figure in parentheses is the number of links with station 5 extension.

specific data sources. The fifth column shows the number of data sources for the DAQ system. The expected event sizes of some of these are so small that two data sources can be merged to single event fragments in the Front-end Readout Link (see Section 7.4). This reduces the total number of data sources to the Front-End Driver Builder to 468.

Expected event fragment sizes, total event size and its rms-estimate of the various sub-detectors for pp (high luminosity run) are sumamrized in Table 7-3. The average expected event size is approximately 680 kB. To account for remaining uncertainties on these average estimates, the average event size is rounded up, for contingency, to 1 MB throughout the rest of this TDR. The DAQ system is therefore designed to handle average event sizes up to 1 MB.

The event sizes for Pb-Pb operation of LHC are dominated by the event sizes of the tracker. The design of the DAQ system does not impose any limit on the size of the events as long as the average data throughput per data source does not exceed 200 MB/s. In Section 14.3.2.1.3, "Tracker Data Format and Data Rates" it is shown, that at the estimated interaction rate of 8 kHz, the increased event size of the Tracker is compatible with this data throughput.

Table 7-3 Event sizes and variations for pp (high luminosity).

Detector	Event size per DAQ link pp (kB)	Total event size pp (kB)	rms of total event size pp
Pixel	1.8 - 2.0	72	~30%
Tracker	0.4 - 1.5 [a]	300	~30%
Preshower	2.3	110	tbd[b]
ECAL	2	~100	~20%
HCAL	2	64	~80%
Muons CSC	1.5	12	~80%
Muons RPC	0.3	1.5	~30%
Muons DT	1.6	8	tbd
Global trigger	2	8	none
DT track finder	2	2	none
CSC track finder	1	1	tbd
Total	--	**678.5**	

a. see [7-9] for event size distribution.

b. to be defined - no information available at the time of writing.

7.3 The Common Interface between the FEDs and the DAQ System

A basic design principle for the CMS DAQ system is the handling of event fragments from all CMS sub-detectors in the same way [7-8], so that the event-building process can be implemented with a uniform set of hardware and software components, independent of the data source. For this reason, a standard interface from the FED to the DAQ system has been designed. The interface specifies a common hardware platform and a data format which is a header and a trailer with information relevant to the event building process. The payload of the event fragments is only inspected in the filter farm, i.e. it is not considered during event building. As a result of this common interface, there is no sub-detector specific hardware after the FED in the readout chain.

7.3.1 Front-End Driver Overview

The main function of the Front-End Driver (FED) is to read out a number of FES modules, store the data in an intermediate buffer, and provide them to the DAQ via a common interface. The FED consists of two functional blocks as shown in Figure 7-3. The first block contains all functionality specific to each individual sub-detector while the second block contains the interface to the DAQ.

The Readout Column is designed for an average event fragment size of 2 kB which at the maximum event rate of 100 kHz corresponds to a sustained data throughput of 200 MB/s. For sub-detectors which deliver significantly smaller event fragments, a Data Concentrator Card (DCC) is developed in order to merge several small fragments into one fragment of approximately 2 kB. In these cases it is the DCC that interfaces to the DAQ system.

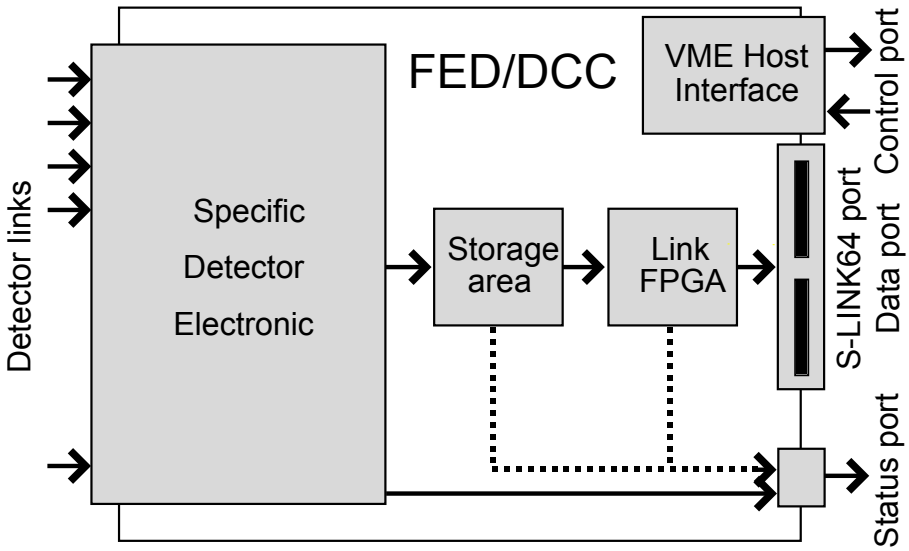


Figure 7-3 Functional block diagram of the Front-End Driver (FED).

During operation, the FED must provide the Trigger Control System (TCS) with status information (see Section 7.6). In case of an imminent buffer overflow, the FED uses the TCS to throttle or inhibit further Level-1 Triggers. Once data in the buffers have been successfully transmitted to the DAQ, the TCS system is notified and triggers resume at full rate. The system is also used to indicate error conditions in the FED. In these cases the Control and Monitor system (see Chapter 12, "Run Control and Monitor System") must take appropriate actions.

For various purposes like monitoring, local DAQ operation, calibration or debugging, each FED implements a local DAQ interface, allowing the readout of event fragments via a local bus interface (VME or PCI). The hardware implementation of this is FED specific. The software environment, however, is provided by the XDAQ system (see Chapter 10, "Online Software").

7.3.2 Hardware of the FED Interface to the DAQ

The interface to the DAQ is an S-LINK64 interface [7-2]. The S-LINK64 protocol is based on S-LINK which has been extended to match the CMS needs for higher bandwidth (64 bits data width and 100 MHz maximum input rate, instead of 32 bits data width and 40 MHz maximum input rate for S-LINK). Both S-LINK and S-LINK64 only specify a set of connectors for sending and receiving data, along with mechanical constraints for the sender and the receiver cards. The physical link between them is not specified beyond these constraints. In addition, both interfaces specify the protocols for writing into the sender card and reading from the receiver card. Both protocols resemble standard protocols that read from and write to a synchronous FIFO. A built-in test mode allows to verify the correct functioning of the link.

7.3.3 FED Data Format

The FED encapsulates the data received from the FES in a common data structure by adding a header and a trailer that mark the beginning and the end of an event fragment. Their content consists of event information partly used in the event building process.

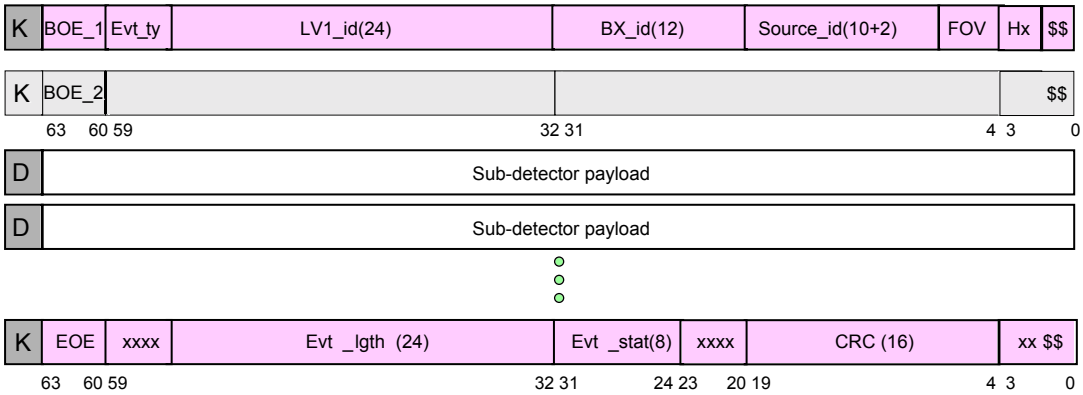


Figure 7-4 The format of the header and trailer 64-bit words in the common data encapsulation. The meaning of the various fields is described in Table 7-4. The K or D field denote a 65th bit transmitted over the SLINK64, which allows to unambiguously differentiate between the payload on one side and the header or trailer words on the other side. This bit is necessary in the S-LINK64 receiver to identify the end of an event fragment.

In Table 7-4 the various data fields in the header and trailer, also shown in Figure 7-4, are described. Since the number of reserved bits in the format is small, the number of header words can be increased dynamically (see “H”- field). This ensures flexibility for future changes.

7.4 Front-end Readout Link

The event fragments of the FED are transferred via the S-LINK64 interface to the FED Builder. The interface between the S-LINK64 receiver card and the FED Builder is called the Front-end Readout Link (FRL) and provides the FED Builder with event fragments. The protocol used by this interface has been developed for a FED Builder built on Myrinet technology and is discussed in detail in Section 7.7.4.

Although some sub-detectors develop Data Concentrator Cards in order to provide the DAQ system with event fragments of 2 kB on average, a significant number of FEDs are expected to provide event fragments with less than 1 kB average size (see Section 7.2). In order to reduce the number of inputs to the FED Builders and to better exploit the available bandwidth in the event builder, the FRL is able to merge data from two FEDs into a single event fragment. The merging is done by concatenating the two incoming fragments for each event in the FRL memory to form a single data block.

The current design of the FRL is a Compact-PCI card with three interfaces:

- Input interface which can host up to 2 S-LINK64 receiver cards.
- Output interfaces to the FED Builder Input (FBI) - currently implemented as a PCI bus since the FBI is a PCI-Myrinet Network Interface Card (NIC). This NIC, along with the optical fibre, implements the data link to the counting room on the surface.
- Configuration and control interface, implemented as a Compact-PCI interface.

Table 7-4 Description of the various data fields in the FED event fragment header and trailer.

Data field	Header (H) Trailer (T)	width [bits]	Description
BOE	H	4	Identifier for the beginning of an event fragment.
Evt_ty	H	4	Event type identifier: distinguishes different kinds of events (e.g. physics, calibration).
LV1_id	H	24	The level-1 event number.
BX_id	H	12	The bunch crossing number. Reset on every LHC orbit.
Source_id	H	12	10 bits unambiguously identify the FED of the fragment. Two bits are reserved for internal FED use.
FOV	H	4	Version identifier of the FED data format.
H	H	1	Indicates if the following data word is an additional header word or if it is the first word of the payload.
EOE	T	4	Identifier for the trailer of the event fragment.
Evt_lgth	T	24	The length of the event fragment counted in 64-bit words including header and trailer.
Evt_stat	T	8	Event fragment status information.
CRC	T	16	CRC code of the event fragment including header and trailer
x	H/T	-	Indicates a reserved bit.
$	H/T	-	Indicates a bit used by the S-LINK64 hardware.

The layout of the FRL card is shown in Figure 7-5. The FRL cards are housed in Compact PCI crates sharing one PC per crate for configuration and control. By implementing test facilities into the FRL, it is possible to test all Readout Column components “downstream” of the FRL without having to rely on functional FEDs as data sources. This will be helpful during the commissioning phase of the system.

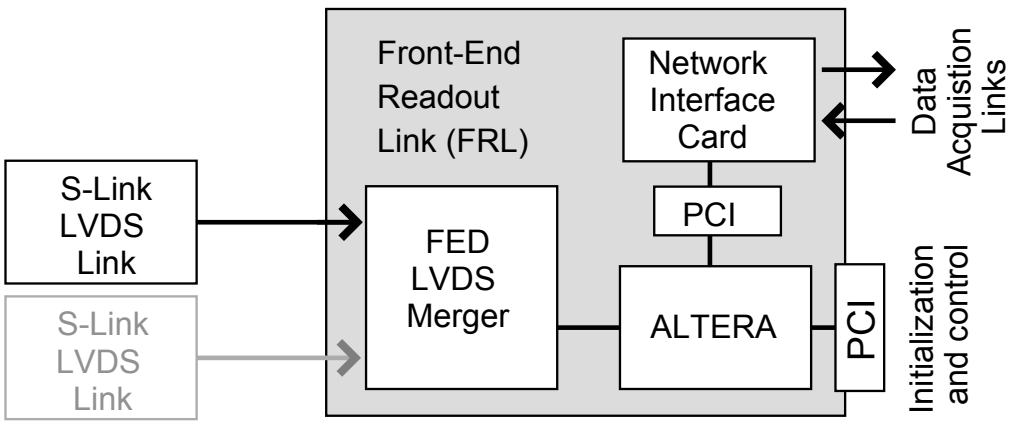


Figure 7-5 Layout of the FRL card. The input can be connected to one or two S-LINKs. The output interfaces with the FBI, which is a commercial Myrinet NIC with PCI form factor plugged directly into the FRL cards internal PCI bus. The PCI bus for initialization and control is a Compact PCI bus. The FRLs are placed into crates with a Compact PCI backplane.

In Figure 7-6 the FRL is shown in the context of the S-LINK64 and the FED Builder.

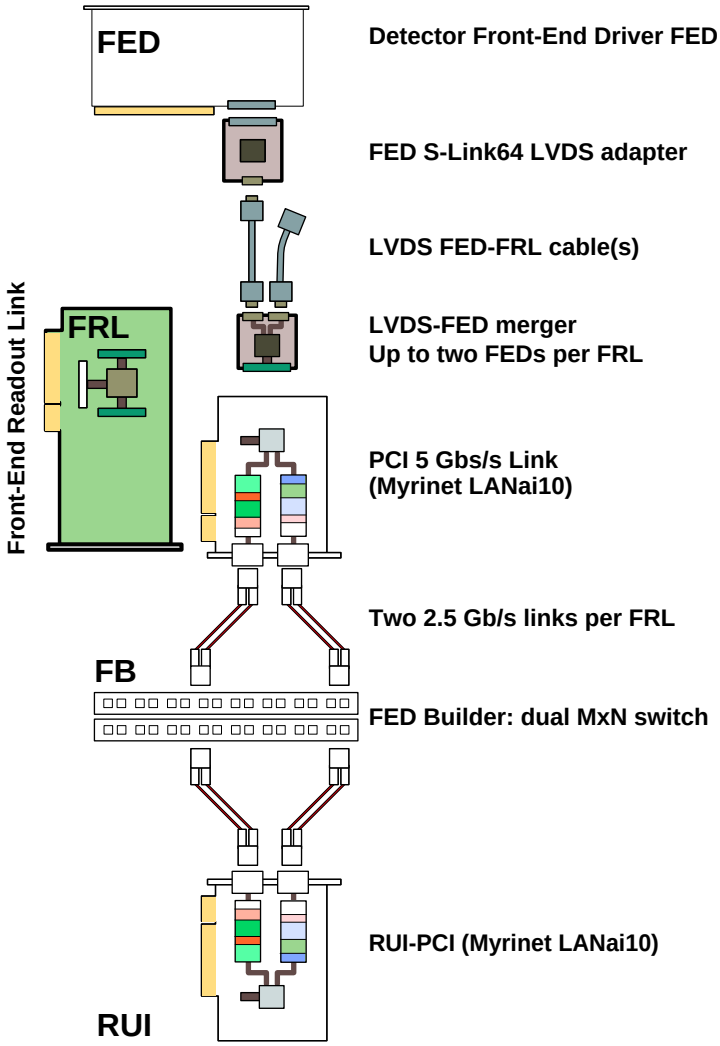


Figure 7-6 Elements in a FED Builder. The FED uses SLINK64 to communicate with an FRL. An FRL can accept data from up to two FEDs. It hosts a Network Interface Card which connects it to the FED Builder switch. On the other side of the switch is the Readout Unit Input (RUI) which can accept data from all the FRLs connected to the FED Builder switch.

7.5 Readout Unit (RU)

Data fragments are assembled in the FED Builder into super-fragments with an average size of 16 kB per event (see Section 4.2, "FED Builder"). For simplicity, super-fragments will be referred to as “fragments” in the rest of this chapter. The Readout Unit (RU) buffers these fragments and sends them, via the Readout Unit Builder switch, to the Builder Unit (BU) which assembles the fragments into full events.

The Readout Unit is the main buffer in the data stream and is used to decouple the event building process in the RU Builder from the corresponding process in the FED Builder. This memory is necessary to absorb the latency of event building and fluctuations of available resources in the Builder Units. The memo-

ry will be large enough to buffer the data stream for seconds (this corresponds to several hundreds of MB).

A block diagram of the RU is shown in Figure 7-7. The RU logically consists of an input unit, the Readout Unit Input (RUI), an output unit, the RU Output (RUO) and a supervising unit, the RU Supervisor (RUS), which controls the RU functions. In addition the RU interfaces to the Event Manager via the Readout Control Network (RCN), and to the Builder Unit via the Builder Unit Control Network (BCN). Both RCN and BCN are logical networks which are used by the event building protocol. If the RU detects a data inconsistency or an internal error which prevents it to continue to function normally, it reports this error via the aTTS system.

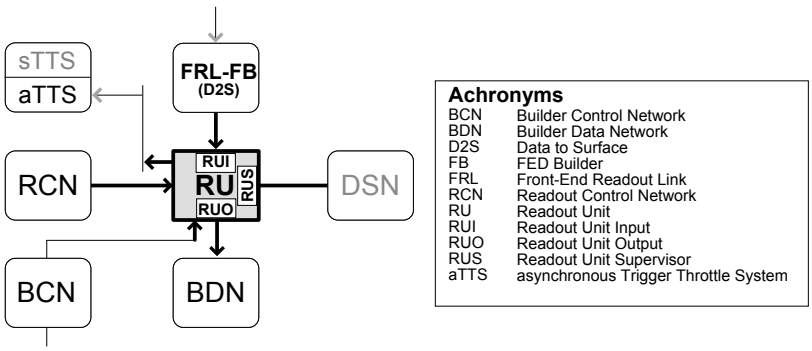


Figure 7-7 The Readout Unit and all its interfaces to other DAQ components.

The event fragments stored in the Readout Unit are requested by the BUs on a random-access basis since the Builder Units concurrently process many different events.

7.5.1 RU Implementation

The functionality of the RU can be implemented in two different ways: a custom design based on FPGAs or a fully programmable commercial component like a high-end Personal Computer (PC). Both options have been evaluated.

A custom board based on FPGAs with interfaces for data input and output has been built. The prototype is described in detail in Appendix D, "Hardware Implementation of the Readout Unit"**.**

With the availability of cost-effective personal computers that can meet the data throughput requirement of the RU, the ongoing development efforts focus on a software design for the RU. This approach ensures flexibility for changes in the event building protocols, the hardware setup and the networking equipment. These changes would be more difficult to implement in a design that is based on FPGAs. The architecture of a PC allows the transfer of data from external interfaces to main memory and from there to other devices via system buses. Depending on the capabilities of these devices, the I/O operation can be driven by direct memory access (DMA) engines to avoid tying up the CPU. It can be initiated either by the CPU or by an intelligent device designed specifically to handle I/O. For this particular application the abilities of the RUI or the RUO to initiate the DMA of fragments into and out of the RU memory are exploited.

The main task of the RU is to intermediately store large amounts of data. The CPU provides the input device with the available addresses in the computer's main memory that serves as intermediate buffer for fragments. This dialogue is performed asynchronously through the use of two message-queues between the CPU and the device. Event data are organized in main memory as chains of fixed sized blocks. Each

stored chain is tagged with an identifier provided by the EVM (see Section 4.3.2). A catalogue of events is created which can be accessed randomly for transmission. The described mechanism avoids CPU-intensive data-copy operations (only two DMA operations per data block are involved). All means to perform these tasks efficiently are provided by the buffer-loaning scheme of the online-software infrastructure described in Section 11.2.3, "Memory Management".

7.6 Flow Control and Front-end Synchronization

The DAQ system has been designed to match the needs resulting from the LHC parameters, the properties of the expected physics events and the architecture of the detector components. As a result the system can operate at an average Level-1 Trigger rate of 100 kHz at an event size of 1 MB. The trigger rate as well as the event size will fluctuate due to changing beam conditions as well as statistical fluctuations. For this reason, buffers have been implemented in various places in the system to absorb these variations. Nevertheless, the system cannot be designed to exclude an overflow of these buffers under all circumstances. An overflow of a buffer in the Front-End Systems (FES) might lead to an inconsistent state and requires a recovery procedure which causes dead-time for the DAQ. In order to avoid this situation, a flow control system called the “Trigger-Throttling System” (TTS) has been designed. By monitoring the state of the DAQ-system components, it adjusts or suspends temporary the Level-1 Trigger. The system also detects a limited set of error conditions for which it tries to execute automatic recovery procedures.

The TTS is divided into a synchronous part (sTTS), and an asynchronous part (aTTS). sTTS communicates with the FES, which run synchronously to the LHC clock, and the FEDs. The buffer space in these systems is small, which requires the sTTS to react quickly to incoming signals. The sTTS is therefore entirely based on hardware. It sends commands synchronously to different LHC signals. The aTTS monitors the DAQ system over extended time periods. According to programmable rules, software processes take actions or inform the Run Control and Monitor system when exceptional running conditions are detected.

The FES and FEDs operate synchronously to the LHC clock and must be synchronized among each other before starting a run. The precisely timed signals required for this are provided by the TTC system. A block diagram of the flow control and synchronization system in the context of the readout column is shown in Figure 7-8.

The following section discusses the functional components for the front-end flow control and synchronization system (sTTS and TTC) and how they are integrated with the Trigger Control System (TCS). It is followed by a description of the sTTS implementation. Finally the concept of the aTTS system is explained.

7.6.1 Front-end Flow Control and Synchronization

The fast control of the front-end readout is described in detail in The Level-1 Trigger TDR [7-3]. In this section the points relevant to the DAQ system are summarized. The Trigger Timing and Control (TTC) system which is used to broadcast commands and signals synchronously to the LHC clock to the FES is discussed first, followed by a description of the front-end flow control system and the way in which it is integrated with the Trigger Control System (TCS).

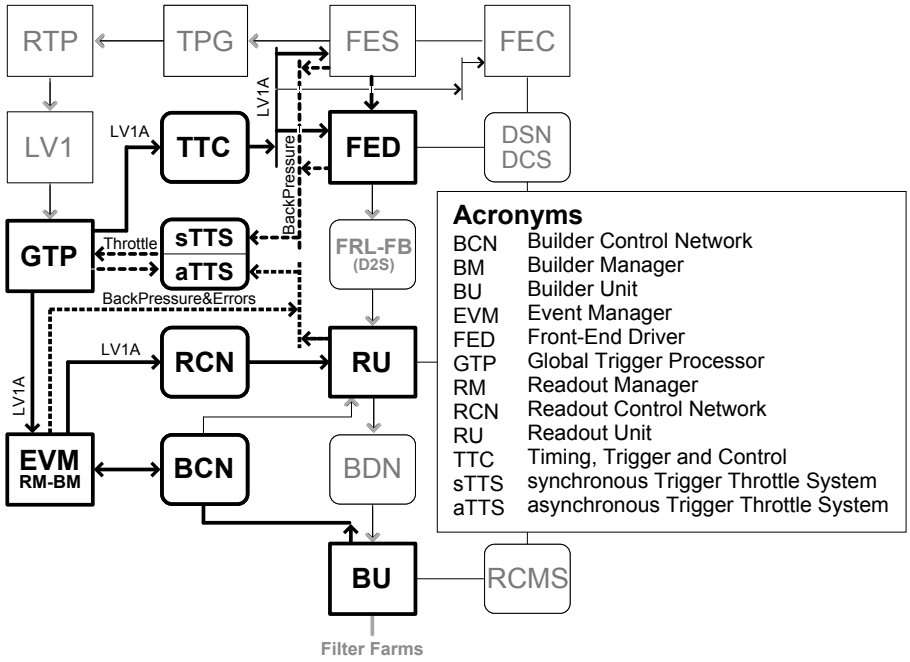


Figure 7-8 Block diagram of the flow control and the front-end synchronization system implemented by the TTS and the TTC system. The systems are shown in the context of the readout system.

7.6.1.1 TTC Network and LHC Clock Interface

The Trigger Timing and Control system (TTC) [7-1] is an optical network that allows the sending of messages from one sender to many receivers. The system has been designed for LHC experiments to broadcast the LHC clock, trigger and control messages to the Front-End Systems with a jitter of less than 100 ps rms. In CMS the TTC is used to broadcast the Level-1 Trigger decision to the FES. The precisely known latency of the TTC messages for each Front-End System is used to extract data from the pipelines for each Level-1 Trigger. Other TTC commands are used to synchronize the FES amongst each other, and to reset and re-synchronize a part of the FES, in case it has lost synchronization with the rest of the system.

The LHC Clock and Orbit signals are distributed from the Prevessin control room to the LHC experiments through single-mode optical fibres. At the experiment counting room, the clock and orbit signals are recovered by circuitry (LHCrx module) in the TTC Machine Interface (TTCmi) crate. At this level the clock jitter is expected to be of the order of 10 ps rms. Fan-out modules (TTCcf) in the TTCmi crate are used to distribute the clock and orbit signals via the TTC system.

7.6.1.2 Front-end Emulators

To prevent overflow of the front-end derandomizers, the TCS houses emulators for the most critical sub-detector derandomizers, i.e. those in the Tracker and the Preshower[1]. The derandomizers store a fixed number of words per Level-1 Trigger. For Tracker and Preshower they behave identically. Their buffer occupancy depends only on the Level-1 rate and on the readout throughput. A state machine receiving the

1. These are the most “critical” in the sense that they have the smallest derandomizer buffers.

Level-1 Trigger emulates the derandomizer behaviour and thus determines its occupancy. Level-1 Triggers which are estimated to cause a derandomizer overflow are inhibited.

7.6.1.3 Trigger Rules and Deadtime Monitor

The assertion of Level-1 Triggers follows a set of Trigger Rules of the general form “No more than *N* Level-1 Triggers in a given time interval”. Suitable rules, inducing less than 1% dead time, minimize the buffer overflow probability. The TCS contains a dead-time monitor that tracks for each crossing whether it is accepted, rejected, or lost due to downtime of trigger and/or DAQ.

7.6.1.4 Synchronization Control

In addition to the Level-1 Trigger, the TCS sends a number of other fast signals to the Front-End Systems also using the TTC system. The Bunch Crossing Zero command, sent after each LHC orbit, is used to reset the front-end bunch counters. Other commands reset the event and orbit counters and issue synchronous start or stop requests.

In certain situations, during data taking, a general reset may be required. Examples are synchronization errors due to single event upsets in the front-end pipelines, generated by buffer overflows or errors in signal transmission. A recovery procedure involving a new run start would lead to an unacceptable DAQ inefficiency. For this reason the TCS is able to distribute ReSync commands to the Front-End Systems. It defines a time interval without triggers that can be used to reset event counters and readout buffers or to partially reset the front-end electronics. These commands can be issued on a periodic basis, independently of the status of the readout electronics, or in response to a loss of synchronization identified in some Front-End System.

7.6.1.5 Calibration and Test Triggers

Calibration and test triggers can be delivered in several different contexts:

- Sub-detectors in stand-alone mode: sub-detectors generate test and calibration sequences locally and capture the data with the sub-detector local DAQ.
- Sub-detectors in DAQ partition mode: The TCS generates test and calibration triggers and the data are collected by a central DAQ partition.
- Periodic test and calibration triggers issued by the TCS during a physics run: the triggers are issued centrally and all Front-End Systems deliver an event fragment in order to stay in synchronization with the rest of the system (the fragment can be empty if the sub-detector doesn’t require test data).
- Local test and calibration triggers issued by the Front-End Systems during a physics run: the Front-End Systems perform test, calibration or monitoring activities during private gaps in the LHC orbit structure or private orbits defined by the TCS.

Test and calibration triggers issued centrally by the TCS follow a well defined protocol. With a fixed delay before the trigger, the TCS issues a TTC command that is used to prepare the test (e.g. laser pulse generation). In order to identify test and calibration triggers, the Trigger system includes a trigger type in the data sent to the Event Managers.

7.6.1.6 Front-end Partitioning

The FES are organized in TTC/sTTS partitions, each corresponding to a major component of a sub-detector. The TCS distributes TTC commands and collects sTTS signals independently for each partition. The TCS software groups the partitions so that they match the DAQ partitions (see Section 4.4). Table 7-5 lists the number of TTC/sTTS partitions for various sub-detectors.

Partitioning requires that the TTC and sTTS trees are organized in branches (TTC/sTTS partitions). The TCS allows to independently operate these partition-groups. It is able to deliver the Level-1 Trigger and other fast commands independently to the partition-groups, and to handle the sTTS fast feedback received from independent partition-groups.

Partitions from different sub-detectors may be combined in one so called partition-group. All partitions are integrated in a single group during a global run. The configuration and operation of partition-groups is done by the TCS. Each partition-group provides its own trigger conditions and only receives the corresponding Level-1 Triggers. A Level-1 Trigger is distributed to only one partition-group at a time.

For test and debugging, it is possible that a sub-detector controls the timing and the trigger via a local trigger controller.

Table 7-5 List of sub-detector TTC/sTTS partitions.

Sub-detector	Number of Partitions	Partitions
Pixels	2	Barrel, Forward
Si-Tracker	4	Disk+, Disk-, Inner barrel, Outer barrel
ECAL	4	EB+, EB-, EE+, EE-
Preshower	2	SE+, SE-
HCAL	6	HB+, HB-, HE+, HE-, HF, HO
RPC	4	Endcap+, Endcap-, Barrel+, Barrel-
DT	2	Barrel+, Barrel-
CSC	2	Endcap+, Endcap-
Calorimeter, Global Muon and Global Trigger	3	Calorimeter trigger, Global Muon trigger and Global trigger
Muon Trigger	2	CSC and DT trigger

7.6.2 Synchronous TTS (sTTS) Implementation

The sTTS network is organized in a tree-structure. It is based on a set of 4 signals which encode up to 16 different system-states as shown in Table 7-6. A tree is made of Fast Merging Modules (FMM) which combine up to 32 incoming signal sets to two different output sets using programmable logic. For each TTS partition a tree is formed of which the combined sTTS signal set is sent to the TCS. The logic of the TCS executes the appropriate actions on each partition.

Table 7-6 Encoding of the four signals in the TTS tree.

signal names					
T3 (Ready)	**T2 (Busy)**	**T1 (Synch. failure)**	**T0 (Overflow warning)**	**name**	**description**
0	0	0	0	DISCONNECTED	hardware failure or broken cable
0	0	0	1	WARNING OVERFLOW	imminent buffer overflow
0	0	1	0	OUT_OF_SYNC	
0	0	1	1	reserved	
0	1	0	0	BUSY	cannot accept more Level-1 Triggers
0	1	0	1	reserved	
0	1	1	0	reserved	
0	1	1	1	reserved	
1	0	0	0	READY	ready to run
1	0	0	1	reserved	
1	0	1	0	reserved	
1	0	1	1	reserved	
1	1	0	0	ERROR	any other error state that prevents correct functioning
1	1	0	1	reserved	
1	1	1	0	reserved	
1	1	1	1	DISCONNECTED	hardware failure or broken cable

7.6.3 Asynchronous TTS (aTTS)

Some error conditions can only be detected by consistency checks of the event data in the Readout Unit or Builder Unit. Those errors are reported to the asynchronous TTS Controller. In the aTTS Controller software processes decide which action to take if a state change is detected. The aTTS Controller also receives the combined sTTS signal sets of the configured DAQ-partitions so that it can use this information when evaluating appropriate actions. In addition to an interface to the Run Control system, the aTTS Controller provides the TCS with a sTTS signal set as described in the previous section for each DAQ partition. This enables the aTTS to request the sending of TTC commands to the Front-end.

7.7 Readout Column Prototypes and Test-benches

The logical components of the Readout Column and their functionality have been described in detail in the previous sections. It has been shown that for some of them there exist several possible implementations. Test-benches are used to select the most suitable one and to investigate the following system aspects:

- Interfaces between the different components of the Readout Column. In particular, the protocols developed to transfer data between components of the column have to be tested and possibly optimized in a test-bench. Similarly, the interfaces between hardware and software components must be tested since also here the communication is performed using custom designed protocols.
- Different implementations of the same logical unit of the Readout Column.
- Performance of the various implementation options for the components.

Three test-benches have been built and used extensively so far. The first test-bench has been used to develop the S-LINK64 interface and a general development platform called FED-kit, which provides hardware and software in order to read data from any FED. The second integrates the Front-end Readout Link (FRL), and the third allows testing the implementation of the Readout Unit (RU). In future, these benches will be connected so that data will flow through all components of the CMS DAQ system from the FED-readout to the Filter farm.

This section contains the status of an ongoing programme of activities and therefore does not present a complete set of measurements.

7.7.1 The Generic Prototyping Platform GIII

Many components of the DAQ column and the test-benches contain hardware devices with a similar structure and functionality. A data stream has to be received, processed with dedicated logic and transferred to another component. An interface to allow configuration and control by a host computer is needed as well. These requirements have led to the development of a generic prototyping platform called “Generic-III” or “GIII” for short.

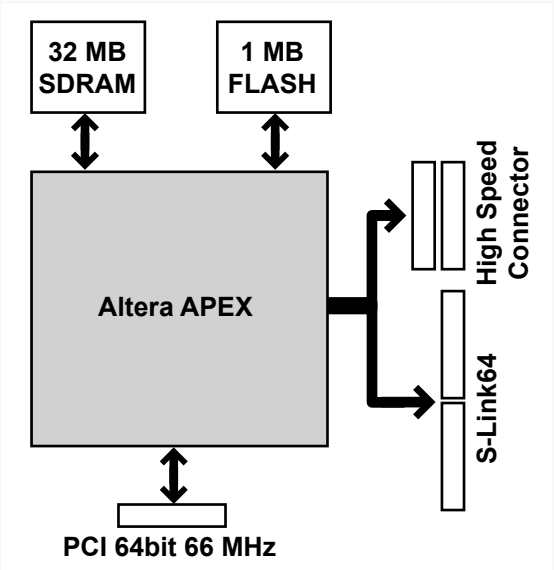


Figure 7-9 Block diagram of the GIII card.

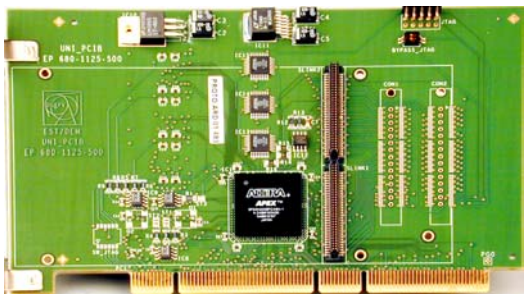


Figure 7-10 Photograph of the GIII prototype card. The High Speed Connectors are not soldered onto the board.

The GIII features a single FPGA (200k or 400k logic gates equivalent) along with a 32 MB SDRAM memory, 1 MB flash RAM and three sets of connectors. A block diagram and a photograph are shown in

Figure 7-9 and Figure 7-10 respectively. One of these connector sets is compliant with S-LINK64. The second allows a custom “add-on” card to be plugged onto the GIII. The third connector is a 64 bit / 66 MHz PCI connector. The logic needed to interface the FPGA with the host PCI bus, the memory devices and the connector sets leaves ~80% of the FPGA free for user applications (for a 200k gate FPGA). During operation, the card is housed in a PC which can control its functionality via the PCI Bus. Since the PCI interface of the card is master-capable the PCI bus can also be used for data transfer into the host PC or into other PCI cards sharing the same PCI address space.

7.7.2 S-LINK64 Prototypes

For the S-LINK64 two prototypes have been built. The first has been used to study the data transmission from the sender to the receiver card via an LVDS cable. It implements a subset of the S-LINK64 functionality. (The S-LINK64 test facility has not been implemented in this prototype.) The design is based on two GIII cards and two driver cards which plug onto the S-LINK64 connectors of the GIII. The driver card functioning as an S-LINK64 source contains TTL to LVDS converters whereas the receiving driver card contains LVDS to TTL converters. The GIII card in the sender functions as a data source whereas the receiving card transfers the incoming data into the host PC, where they are checked by the control software. The result of the measurements are summarized in Table 7-7:

Table 7-7 Summary of the measurements made with an S-LINK64 prototype. Various LVDS cables from different vendors have been tested.

vendor	length [m]	test duration	rate [MB/s] / ν [MHz]	remark
AMP	2	1 month	800 / 100	no error
	7.5	8 hours	800 / 100	no error
	2+7.5+7.5	8 hours	528 / 66	no error (3 cables connected to each other)
Amphenol	15	4.5 hours	528 / 66	no error
	20	minutes	264 / 33	errors
3M	10	1 hour	528 / 66	no error
	10	1 hour	528 / 66	no error

The data rates achieved with this setup depend on the length of the LVDS cable: 528 MB/s with cables up to 17 m are achieved. A 20 m long cable of one vendor did not function correctly. With a cable of 2 m length a long term test has been performed. For two months data have been transferred at a rate of 800 MB/s and no transfer error has been detected.

Although the layout of the racks in the control room is not yet finalized, it is assumed that no FED needs to be connected to the DAQ system by a cable longer than 15 m. In the final system the link will be operated at 66 MHz (equivalent to a maximal data throughput of 528 MB/s).

The second prototype implements the full functionality of the S-LINK64. It consists of a sender card in PMC form factor with an FPGA which buffers the incoming data and implements the protocol to the FED. Data to be transferred is converted to LVDS levels. The receiver card has been designed to receive data from either one or two sender cards thus allowing for merging of data from two FEDs, as described in Section 7.4, "Front-end Readout Link". The card contains LVDS to TTL converters, FIFOs to buffer the incoming data and an FPGA to implement the protocol to the mother board.

7.7.3 The FED-kit

The S-LINK64 is needed by FED developers to test their hardware. For this reason the FED-kit[7-10] has been developed. It consists of an S-LINK64 sender, a GIII card housing an S-LINK64 receiver, and a software package integrated into XDAQ. It provides an easy-to-use API that enables users to read out FEDs via the S-LINK64-port. Figure 7-11 shows the hardware components of the FED-kit.

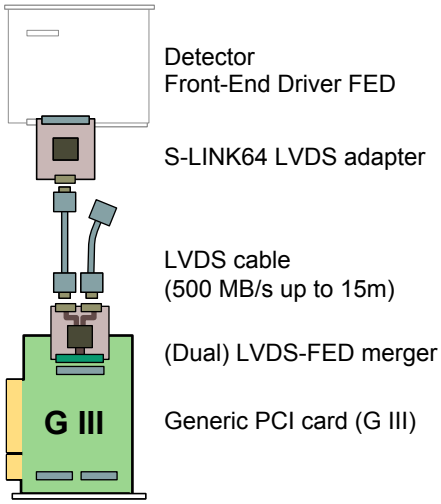


Figure 7-11 Hardware components of the FED-kit.

7.7.4 The FED-Readout Link Test-bench

The aim of the FRL test-bench is to develop the protocol between the FRL and the FED Builder Input (FBI) and to show that this system can transfer data from the FEDs to the RUs with a sustained throughput of 200 MB/s. For this purpose the setup shown in Figure 7-12 has been implemented. A GIII-card is

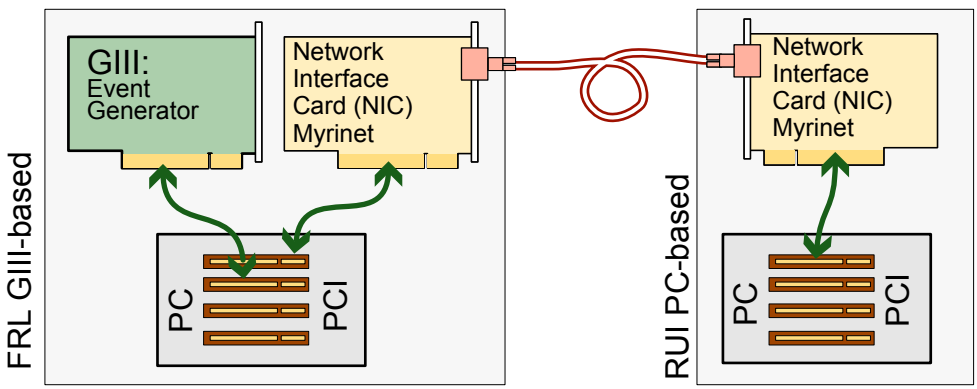


Figure 7-12 Block diagram of the FRL test-bench.

programmed as an event-generator and FRL. The event-generator provides the FRL prototype with dummy event fragments of programmable length formatted as described in Section 7.3.3. LV1_id and Source_id are programmable for each event. Data are transferred over a 64 bit PCI Bus (66 MHz) to a

Myrinet network interface card which implements the FBI. The GIII card and NIC are housed in one PC and share the same PCI bus. On the receiver side the Myrinet interface card transfers the data over the PCI bus into the memory of a PC. The protocol between the FRL and the NIC card is discussed in the following section.

7.7.4.1 Protocol Between the FRL and the Network Interface Card

The Myrinet NIC card contains a programmable RISC CPU. The on-board memory contains the program for the RISC and serves as a buffer for the data to be transmitted. The PCI interface can be operated as master or slave.

As can be seen in Figure 7-4 the length of an event fragment is only stored at the end of the fragment which makes it impossible for the FRL to determine if enough memory space for the entire fragment is available in the NIC. To solve this problem the protocol has been implemented using a “buffer-loaning”-scheme based on the technique described in Section 11.2.3, "Memory Management": the buffer memory of the NIC is divided into blocks of fixed size. Each block contains reserved space for a Myrinet header, the FRL header, and the data payload. The following steps are executed during operation:

1. As soon as there is a free memory block available, the NIC writes its address into a FIFO in the FRL. At start-up all blocks are available. During operation a block becomes available after the data it contained has been sent successfully over the Myrinet link.
2. As soon as there are pointers to free blocks in the FIFO, an event is generated. In this test-bench setup, the parameters of the event fragments to be generated (length, LV1_id, Source_id) must have been written previously into a second FIFO by the control software. The generated event is transferred by the FRL via a DMA directly into the payload area of the memory block. The FRL header for this block is written afterwards. If the fragment is too large to fit into one memory block, a special flag in the FRL header indicates that the fragment is continued in the next block. Step 2 is repeated until the whole fragment has been written into the NIC buffer memory.
3. The control process in the NIC monitors the headers which are filled by the FRL. As soon as a flag therein indicates the presence of a new block, it starts to fill the reserved space for the Myrinet header and subsequently sends out the block over the optical link. After successful transmission the block is freed by setting the fragment length to 0. Its address is written into the FRL FIFO.

7.7.4.2 Measurements and Results

Two different measurements have been performed with this test-bench. In order to measure the overhead due to the protocol introduced by the FRL, the first measurement was taken with the FRL prototype plugged into a PC without any further hardware. A control program in the PC implemented the protocol described above. In this configuration the overhead due to the control program of the PC is negligible. The data throughput on the PCI bus has been measured as a function of the fragment size (see Figure 7-13). The asymptotic limit for a large fragment size is measured to be 500 MB/s which is close to the theoretical limit of 528 MB/s for the PCI bus. The block size was set to 4 kB. A sawtooth structure is observed since every time a new block is needed to transfer a complete fragment, additional overhead is introduced in order to set up the DMA transfer for the block. This overhead, which has been measured to be 0.6 μs, limits the total data throughput to 500 MB/s.

In a second measurement, the complete chain described above was set up. Details of this measurement are described in Section 6.4.4.3.2, "Building of Super-fragments".

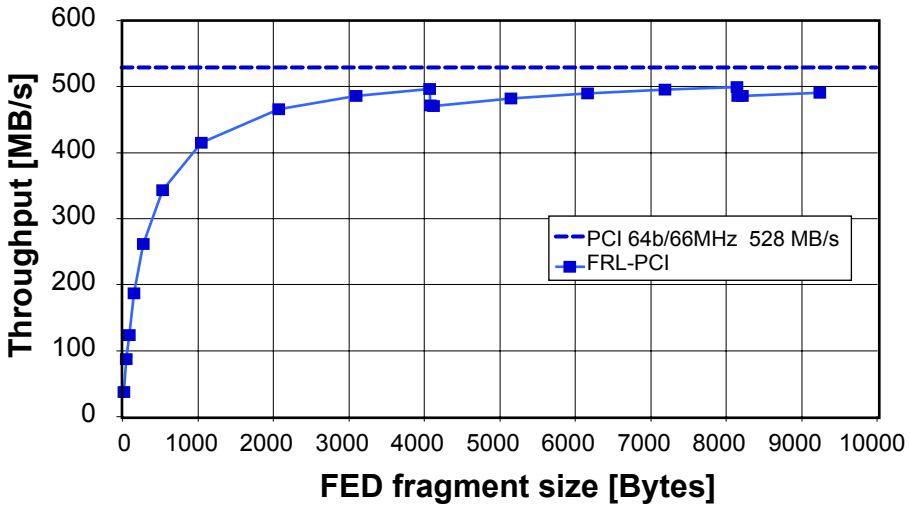


Figure 7-13 Performance of the FRL PCI interface: throughput as a function of the fragment size. A GIII-based FRL is plugged into a PC which runs a control program implementing the protocol discussed in the text.

7.7.5 The Readout Unit Test-bench

A Readout Unit test-bench has been set up, in order to measure the data throughput of the RU under realistic conditions. The functional diagram is shown in Figure 7-14. The Readout Unit is implemented by a PC with at least two 64 bit / 66 MHz PCI slots. The RU-input (RUI), implemented by a GIII card, pushes data into the RU. At the output, data are transferred first to a NIC, which functions as a RUO, and from there to the Builder Unit (BU). The BU checks the incoming data for correctness.

The setup contains a data generator and a trigger emulator. The trigger emulator accepts veto signals to simulate the Trigger-Throttling System (TTS). Both the Event Manager (EVM) and the data generator can send a veto to the trigger emulator. The event generator is contained in the RUI and generates an event on receipt of each Level-1 Trigger. The distribution of the event-size is programmable. The generated event is fed into the RU via the PCI bus of the host PC.

The data and the control messages of the event builder protocol are sent over a Myrinet network. The protocol used is GM, developed by Myricom.

The software of the test-bench is written in the XDAQ framework which is described in Chapter 10. The buffer-loaning scheme described above (see Section 7.7.4) prevents the loss of data in case of buffer shortage in the RU. Different event building protocols between RU, BU and EVM can be tested in the set-up. All measurements presented in the next section were performed using a variant of the event building protocol described in Section 5.3.2.1, "Event Flow Protocols", currently implemented in XDAQ. This is not expected to affect the results on throughput described hereafter.

7.7.5.1 Measurements and Results

As can be seen in Figure 7-14, the control messages are sent over the same network as the data. In order to reduce the network load due to control messages, their number is reduced by bundling the “send to BU” messages from the EVM to the RU: a single message contains the request for 50 event fragments. The same method has been applied for the “resource available” messages of the BU to the EVM.

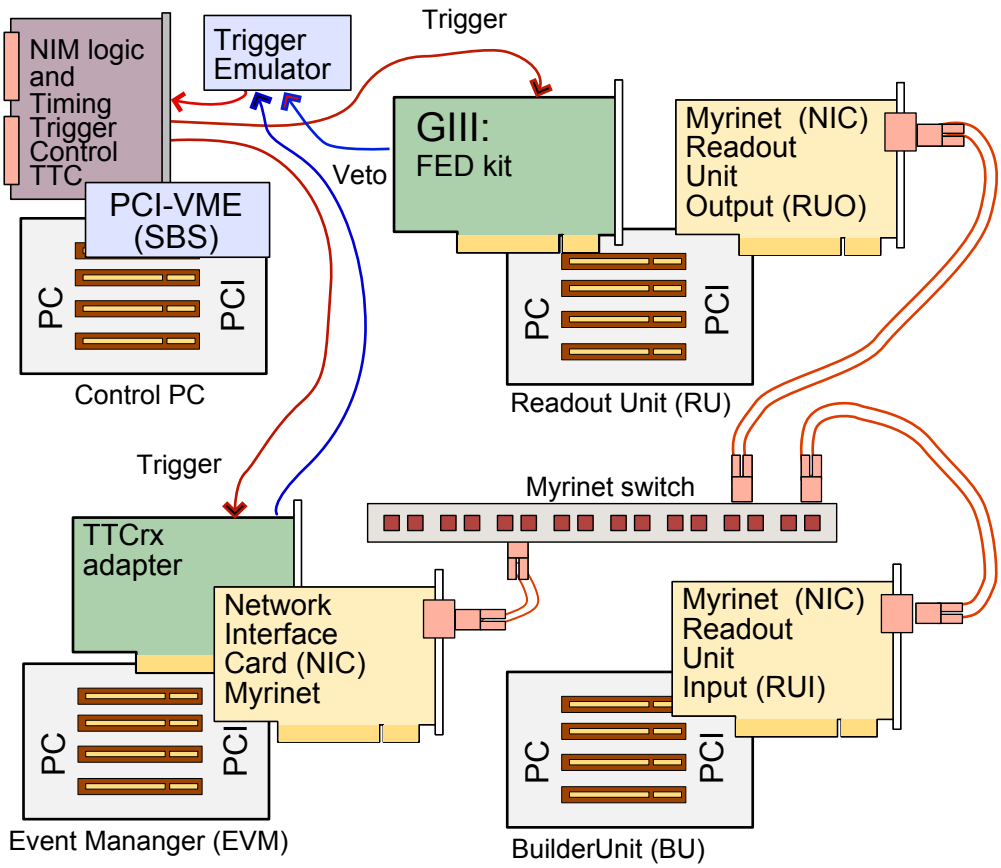


Figure 7-14 Functional diagram of the Readout Unit test-bench.

Measurements have been performed with four different PCs. Their characteristics are summarized in Table 7-8.

The measured throughput for the RU as a function of the fragment size is shown in Figure 7-15. The size of memory blocks used in the buffer-loaning scheme has been chosen to be 16 kB. Measurements with buffer sizes of 4 kB or 8 kB showed no significant changes in the result. The measurements show a characteristic sawtooth structure with a 4 kB structure, caused by the use of 4 kB buffers in the GM protocol. If the fragment size passes a 4 kB boundary the overhead of the GM protocol to prepare a new 4 kB buffer is added.

The measurements for the Poweredge 1550 and the P4DL6 motherboards result in the highest throughput. On these motherboards the two PCI slots used for the RUI and the RUO are independent. Although the P4DL6 based PC is a significantly more powerful machine than the Poweredge based PC[1], the difference in the measured throughput is insignificant since it is to a large extent limited by the performance of the Myrinet link (250 MB/s bandwidth) in the RUO. On the DLE370 the RUI and the RUO share the same PCI bus. Therefore it is not possible to transfer data into and out of the RU at the same time. The measured data throughput for this configuration is 25% lower than the above mentioned measurements. The throughput drops further to 115 MB/s if the DLE370 motherboard is equipped with only one CPU.

1. A hyperthreaded XEON processor contains two logical processors which for many applications results in a performance almost equivalent to a dual processor system.

Table 7-8 Important characteristics of the PCs used to implement the RU in the Readout Unit test-bench. The PCs are based on motherboards of two vendors: SuperMicro and DELL.

	SuperMicro	SuperMicro	DELL	SuperMicro
Motherboard	DLE 370	DLE370	Powerdege 1550	P4DL6
Chipset vendor	ServerWorks	ServerWorks	ServerWorks	ServerWorks
Chipset	ServerSet III-LE	ServerSet III-LE	ServerSet III HE-SE-SL	Grand Champion LE
CPU	1 Pentium III	2 Pentium III	2 Pentium III	1 Pentium IV Xeon (hyperthreaded)
CPU clock	1 GHz	1 GHz	930 MHz	2.0 GHz
Memory	SDRAM (1.06 GB/s)	SDRAM (1.06 GB/s)	SDRAM (interleaved) (2.12 GB/s)	DDR SDRAM (interleaved) (3.2 GB/s)
System Bus	1.06 GB/s	1.06 GB/s	1.06 GB/s	3.2 GB/s
PCI 64 bit / 66 MHZ slots	2 slots on 1 bus	2 slots on 1 bus	2 slots on independent buses	4 slots on 2 independent buses

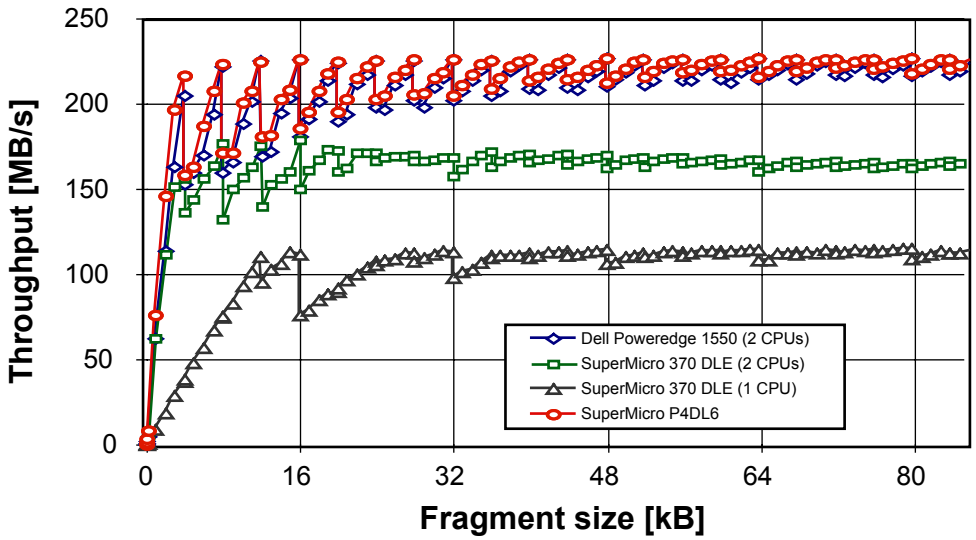


Figure 7-15 Data throughput measured in the RU as a function of the fragment size. Each measurement has been performed with fixed fragment size. The measurements have been performed with 4 different PCs (see text).

A similar measurement is shown in Figure 7-16. Here the fragment sizes are varied according to a log-normal distribution. The rms of the distribution has been set to its mean divided by $\sqrt{8}$[1].The variation of the event size smears out the characteristic sawtooth structure of the previous measurements. For a mean event size of 16 kB, a data throughput of 205 MB/s is measured for the P4DL6-based PC and

1. This is motivated by naively assuming that each 16 kB fragment has been assembled in the FED Builder out of 8 2-kB fragments. For each of these fragments a size distribution according to a log normal distribution with an rms equivalent to its mean is assumed.

198 MB/s for the PC based on the Poweredge 1150. Since the required sustained data throughput for the readout column is 200 MB/s, this test-bench shows that it is possible to implement the RU with today's standard PC-technology.

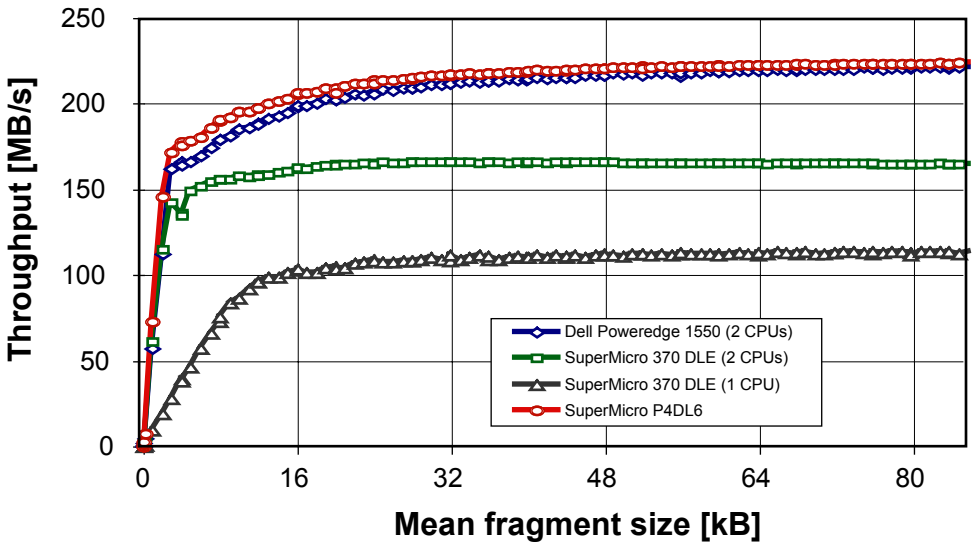


Figure 7-16 Data throughput measured in the RU as a function of the fragment size. Each measurement has been performed with fragment sizes distributed according to a log-normal distribution. The mean of the distribution has been varied. The rms of the distribution has been set to the mean divided by $\sqrt{8}$.

7.8 Summary and Outlook

This chapter has described the structure and components of the Readout Column. Design choices for this components were motivated, in light of both general requirements of the DAQ, and requirements specific to the readout of sub-detectors.

In particular, a common interface for the connection of all FEDs to the Readout Column has been defined, along with the generic FED data format. The physical implementation of this link, the S-LINK64 driver and receiver cards, have been prototyped and tested successfully using a generic prototype PCI card developed to handle the input, processing and output of an arbitrary data stream (GIII). Cables from different vendors were evaluated for the FED-FRL connection, and most were found to be up to the specifications for the distances to be covered.

The FRL board conceptual design has been presented. This board serves the dual purpose of providing a first merging of FED data, to balance fragment sizes for improved efficiency, and a transition to a more suitable protocol to transfer data to the rest of the DAQ system on the surface. It also adds flexibility in case of modifications to details of the S-LINK64 protocol as well as testing capabilities which will be invaluable during the commissioning phase.

Using the same generic platform discussed above, the protocol to be used to connect the FRL to the FED Builder Input was studied to show that a sustained throughput of 200 MB/s is achieved. These results, along with the ones discussed in Chapter 6.4.4, indicate that a Myrinet-based FED Builder, fed by a two-rail Myrinet NIC (FBI) connected to the FRL, can provide the necessary throughput. The final design of the FRL board is well under way.

The main requirement placed on the Readout Unit is the concurrent read/write access to memory, with a throughput of 200 MB/s. A custom hardware solution has been made obsolete by the availability of commercial personal computers which can meet this I/O requirements. As a consequence the prototype activity has concentrated on maximizing the exploitation of the memory bandwidth by means of DMA data transfers for input and output, using an efficient memory management based on buffer-loaning. The prototype software has been tested in a dedicated setup, on several different personal computers. The two most recent machines used in the test can both provide the necessary throughput.

In conclusion, for all the components of the Readout Column, an implementation that can meet the requirements has been identified and tested. The design of the custom components that have to be installed in the underground area is being finalized and the following production phase is expected to meet the installation deadlines. Further tests of the production design, as well as integration tests of all the components in a complete Readout Column, are planned.

7.9 References

7-1 B.G. Taylor, “TTC Distribution for LHC Detectors”, IEEE Trans. Nucl. Sci., 45 (1998) 82;
B.G. Taylor,”Timing Distribution at the LHC”, in Proceedings of the 8th Workshop on Electronics for LHC Experiments, Colmar, France, 9-13 September 2002, pp. 63-74.
Timing Trigger and Control system, see http://www.cern.ch/TTC/intro.htm.

7-2 A. Racz, R. McLaren, E. van der Bij, “The S-LINK64 bit extension specification: S-LINK64”, available at http://hsi.web.cern.ch/HSI/s-link;
O. Boyle, R McLaren, E. van der Bij, “The S-LINK Interface Specification”, available at http://hsi.web.cern.ch/HSI/s-link

7-3 CMS Coll.,“The Trigger and Data Acquisition project, Volume I, The Level-1 Trigger, Technical Design Report”, CERN/LHCC 2000-038, CMS TDR 6.1, 15 December 2000.

7-4 Technical Design Reports of the CMS sub-detectors:
CMS Coll.,“The Tracker Project, Technical Design Report”, CERN/LHCC 98-6, CMS TDR 5, 15 April 1998
Tracker Addendum - CERN/LHCC 2000-016, CMS TDR 5 Addendum 1, 21 February 2000
CMS Coll.,“The Electromagnetic Calorimeter Project, Technical Design Report”, CERN/LHCC 97-33, CMS TDR 4, 15 December 1997
CMS Coll.,“The Hadronic Calorimeter Project, Technical Design Report”, CERN/LHCC 97-31, CMS TDR 2, 20 June 1997
CMS Coll.,“The Muon Project, Technical Design Report”, CERN/LHCC 97-32, CMS TDR 3, 15 December 1997

7-5 S. Cittolin, J.F. Gillot, A. Racz, “Front-end logical model in CMS”, CMS Technical Note 1996/015.

7-6 A. Racz ,”Trigger throttling system for CMS DAQ”, in Proceedings of the 6th Workshop on Electronics for LHC Experiments, Cracow, Poland, 11-15 September 2000, pp. 405-409.

7-7 N. Marinelli, “APV logic simulations”, CMS Note 1999/028.

7-8 CMS DAQ Readout Unit Working Group, see http://cmsdoc.cern.ch/cms/TRIDAS/horizontal

7-9 O. Kodolova, I. Tomalin and P. Yepes, “Expected Data Rates from the Silicon Strip Tracker”, CMS Note-2002/047.

7-10 E. Cano et al. “FED-kit design for CMS DAQ system”, in Proceedings of the 8th Workshop on Electronics for LHC Experiments, Colmar, France, 9-13 September 2002, pp. 274-277.

8 Filter Column

The last stage of the event building consists of assembling data fragments belonging to the same Level-1 Trigger into full events in the “Builder Unit”. The Builder Unit forwards events to processors (Filter Units) running the High-Level Trigger algorithms which select the small fraction of events that are forwarded to permanent storage. A Builder Unit and the Filter Units it is connected to constitute a Filter Column which is the subject of this chapter.

The components of the Filter Column are introduced in the following section, while the requirements and general architecture of the Filter Farm are reviewed in Section 8.2. Section 8.3 describes the Builder Unit while Sections 8.4 and 8.5 describe the Filter Farm and its elements. FC prototypes and results are contained in Section 8.6. Finally Section 8.7 presents an outlook on the system.

8.1 Introduction

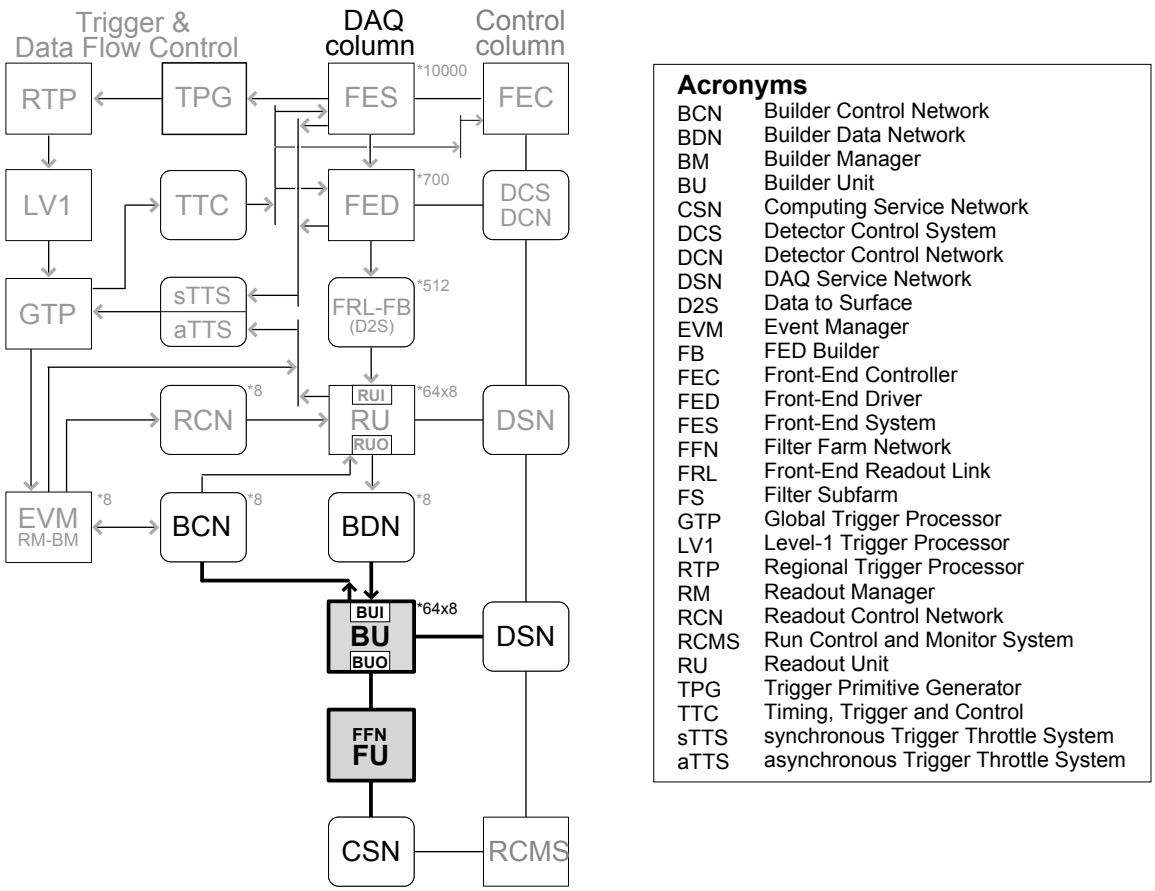


Figure 8-1 Location of the Filter Column in the CMS DAQ.

The Filter Column (Figure 8-1) provides two functions in the DAQ system. In the first, event data from the readout subsystems are assembled (or “built”) into a single data structure that contains all the data necessary for the reconstruction of the information in the CMS detector from a single bunch crossing. In the second, the raw data from these assembled events undergo an analysis process which, through the ap-

plication of reconstruction and filtering algorithms, referred to as the High-Level Trigger (HLT), leads to a decision on whether to reject the event or to keep it for offline analysis. The events rejected by the HLT are discarded, whereas the events accepted are transferred to the Computing Services (see Chapter 16, "Computing Services") for permanent storage.

The Filter Column (FC) represents a “vertical slice” of the system, which is functionally split into two units, namely the Builder Unit (BU) and the Filter Unit (FU). The Builder Unit is part of the Event Builder, and its functionality is to provide the event assembly. It is thus tightly coupled to the architecture of the data network and the readout. The Filter Unit is the basic element on which the HLT algorithms are executed. It is a unit which is presented, by the BU, with fully assembled events and thus its functionality is independent of the details of the Event Builder. The entire collection of Filter Units is referred to as the “Filter Farm”.

8.2 Filter Farm Requirements and Architecture

The CMS Filter Farm consists of a large number, of O(10^3), of processors. The farm is organised in a number of smaller subfarms each with ~100 processors (see Figure 8-2). This organization has numerous advantages, since:

- it allows for a staged deployment of the total computational power required
- it introduces a natural hierarchical structure to the farm, simplifying its management by facilitating the dispatching of control, monitor, and error messages.
- it improves the robustness and fault tolerance of the system by making it possible to keep the output streams separated at the level of each subfarm, thus decreasing the risk of data loss (and experiment deadtime) due to unavailability or fault in the central data storage services.

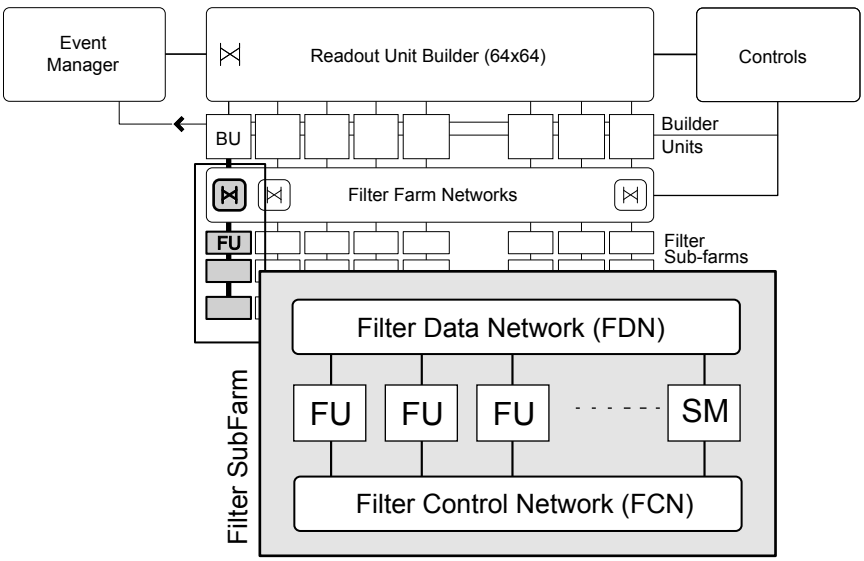


Figure 8-2 Filter Farm and the Event Builder (top) and the subfarm structure (bottom).

Each subfarm corresponds to one or more BUs connected to one or more of the Builder Data Network (BDN) output ports, and comprises a number of FUs. The BUs of a single subfarm belong to the same RU

Builder. A FU may correspond to a physical unit of computation (e.g. a PC) or to a logical unit in a more complex system consisting of several computational units (e.g. a multi-cpu server). In a subfarm, the FUs are connected to the BUs via a Filter Data Network (FDN). A Special node called Subfarm Manager (SM) handles the distribution of code, configuration and control commands to all the FUs as well as the collection of monitor data. It is connected to the FUs via the Filter Control Network (FCN). The FDN and FCN are collectively referred to as Filter Farm Networks.

The requirements on the performance of the filter farm are particularly tight. The farm must cope with the full event rate accepted by the Level-1 Trigger, i.e. an aggregate event rate of up to 100 kHz. Moreover, the rate of events output to storage is $O(10^2)$ Hz and thus the HLT algorithms must achieve a rejection factor of $O(10^3)$. A subfarm spanning n BUs of the N available must sustain an event rate up to $n/N \times 10^5$ Hz, which corresponds to a mean time of $N/n \times 10^{-5}$ s per event. Using, as indicative figures, the values N=512 and n=8, we conclude that each subfarm must handle ~1.5 kHz of events, or a mean of 0.64 ms per event. Irrespective of the size of the subfarm, a single BU must handle a rate up to 200 Hz, at 5 ms/event.

Each of the FU nodes is connected to the BUs via a network link. The time available for a FU to process an event is proportional to the number of FUs per BU. A variable number of FUs per BU will therefore be deployed depending on the performance of a unit. The minimal amount of connectivity to fit a unique correspondence between a BU and its FUs is driven by the maximum number of FUs per BU to be deployed. This connectivity must be guaranteed by a switched network. To allow dynamic, partial or full reconfigurability of a subfarm, a larger fabric is necessary.

The above throughput requirements imply either the usage of multi-gigabit/s links for the BU-FU connection, or the employment of additional techniques to reduce the traffic of data between the two units. An example of such a technique is the possibility to transfer events between the BU and FU only partially, on a “need” basis. Since the actual filtering process is performed in steps, and each step uses typically only a subset of the event data and results in the rejection of numerous events, one can envisage the transfer of the event data from the BU to the FU in multiple steps, each step containing only the data necessary for the next filtering algorithm. We illustrate this procedure using a simple two-stage filter which consists of two “trigger algorithms”, called symbolically “Level-2” and “Level-3”, in complete analogy with the trigger levels of traditional DAQ systems. Suppose that the Level 2 decision can be based on only 1/4 of the total event data (e.g. the data from the calorimeter and muon detectors), and that results in the rejection of O(90%) of the input events. At an average event size of 1 MB, the average amount of data to be transferred by a single BU to the nodes of a subfarm is only ~70 MB/s, compared to the 200 MB/s that would correspond to the transfer of the full data for each event. The figures of the example used are taken from measurements using simulations of physics events in the CMS detector. In general, the amount of processing to be performed to reach a decision can be minimised by only reconstructing regions of the detector corresponding to interesting features of the event. These regional reconstruction can help to further reduce the fraction of data to be transferred for a certain detector, if the readout segmentation conveniently maps typical regions of interest. A more detailed discussion of the data needed for each filtering step, regional reconstruction, and the associated performance of each algorithm can be found in Chapter 15, "Physics Object Selection and HLT Performance".

Although the Builder Units may absorb processing time fluctuations, the Filter subfarm must keep up with the BU input rate at “equilibrium”. To ensure the scalability and seamless evolution of the system, the FU software implementation must clearly be made as platform independent as possible, allowing a mixed operating system / hardware architecture scenario. However, the level of portability is expected to be determined by the reconstruction software and related products.

The aggregate output of an individual subfarm is the aggregate accept rate ($n \times 0.2$ Hz, where n is the number of BUs per subfarm) times the average size of an output event, which for contingency we can assume to be ~ 1.5 MB[1]. The amount of output storage needed by a single subfarm with n = 8 and over 24

hours of data taking is therefore ~ 200 GB. The corresponding average number of transactions per second to such a storage is small compared to the Farm total. To allow the Filter Farm to continue working while e.g. the connection to the Computing Services is unavailable (see Section 16.2), it is convenient to provide local mass storage on an individual node of the subfarm, (which can be physically the same as the Subfarm Manager), to store accepted events. Subsequent transfers to the Computing Services can be conveniently sized and scheduled, e.g. on a run-by-run or fill-by-fill basis. In case of failure of a subfarm, causing local data corruption, the amount of lost data is also minimized.

When data taking is first started, or after power off of the whole farm, a “cold start” must be initiated. The cold start includes booting and configuration of all the machines in the farm, download of new software and bootstrap of the relevant DAQ daemons on the FU nodes, and of server software on the Subfarm Managers. The cold start is managed by the Farm Control under the supervision of the main Run Control system. In order not to cause experiment dead time, the cold start time must be negligible on the time scale of the setup time of the LHC beam. At the end of a cold start the farm is ready to be configured and started, which is the normal condition between machine fills. The configuration and starting of all the processes in the farm, which leads to the farm being ready to take data, is referred to as a “warm start”. The warm start time must be negligible with respect to the interval between two machine fills. The hierarchical structure of the farm is meant to minimize the cold start time by parallelizing the software distribution and the execution of configuration commands on all the nodes. The distribution of control commands needed to start data taking on the farm once it is configured, exploit the same hierarchical structure, whereby individual nodes are controlled and report their status to the relevant Subfarm Manager, which in turn responds to commands from Run Control. This hierarchical structure is outlined in more detail in Section 8.5.

The farm monitoring must guarantee reliable delivery of alarm messages which have system wide scope or require operator intervention, with a delay which is small compared to the reaction time of the system or operator. Provisions must be made to allow for alarm masking and resetting. Monitoring of working parameters must guarantee timely delivery of monitor information. For sampled monitor information (e.g. event display), sampling of all the nodes according to a “fair share” policy must be guaranteed. The data throughput required for update of monitor information to the monitor clients is expected to be consistent with what can be provided by a commercial “office” network.

The farm is operated as a collection of independent subfarms. Different subfarms can be assigned to different partitions, and failure of a subfarm in a partition will not affect the performance of the other subfarms. Within a subfarm, FU nodes are only coupled insofar as they depend on the same BU for their input. Therefore, in principle, a subfarm can also be partitioned. Failure of one or more FUs hanging from a given BU will cause that BU to work at a lower event rate. If “more graceful” performance deterioration is required, dynamic reconfiguration of FU assignments must be guaranteed by appropriate switching capabilities in the FDN.

8.3 Builder Unit (BU)

The task of the Builder Unit is to retrieve data from the RUs in the form of event “super-fragments” (see Section 4.2), to assemble these super-fragments into full events, and to send data from assembled events to FUs upon request. The BU is connected, as shown in Figure 8-3, to the Readout Units (RU) and the Fil-

1. Throughout this Technical Design Report, an average size of 1 MB for the raw event is assumed. Here, provision is made for the possibility to attach to the raw event additional information which is the result of the reconstruction and analysis carried out during the selection.

ter Units (FU), and works under the control of a Builder Manager (BM) which is part of the Event Manager (EVM).

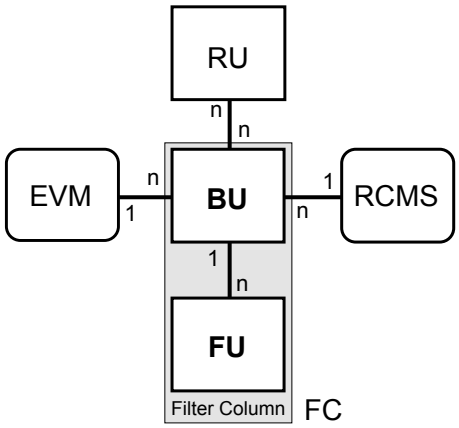


Figure 8-3 Builder Unit and its connections to external subsystems.

8.3.1 BU Connectivity and Elements

The Builder Control Network (BCN) carries requests for events from BUs to the BM. The latter provides the mechanism to control the flow of transactions in the Readout Unit Builder, as explained in Chapter 5, "Event Flow Control". The BM responds to a BU request with an event identifier, that is used by the BU to request data directly from the RUs.

The Builder Data Network (BDN) serves as a medium to transfer event fragments from the RUs to the BUs. The DAQ Services Network (DSN) provides the link to the Run Control and Monitor System (RCMS). The Filter Data Network (FDN) serves event data transactions to the FUs.

The Builder Unit is logically split into three functional components (Figure 8-4). These components can operate concurrently for improved efficiency:

- The Builder Unit Supervisor (BUS) handles all configuration, subsystem control and monitoring tasks.
- The Builder Unit Input (BUI) handles all communication with the BCN and the BDN. It also keeps track of the event data in the Builder Unit.
- The Builder Unit Output (BUO) implements the interface to the FDN.

Depending on the chosen implementation platform, the internal communication between these three components relies on a single or multiple physical links. One possible approach is presented in Section 8.6.1.

The requirements on the BU interfaces are summarized in Table 8-1. These figures correspond to an average super-fragment size of 16 kB per Readout Unit, and a total Level-1 accept rate of 100 kHz, or 12.5 kHz per RU Builder with 8 RU Builders each containing 64 Readout Units. A BU must therefore sustain a throughput of 200 MB/s on the BDN. The average message rate per link, listed in Table 8-1, depends on the Level-1 Trigger rate and the number of packets needed to fit a message. This number can be derived from the Maximum Transfer Unit size (MTUSize) and the average size of an event fragment (FragSize). As an example, for an MTU size of 4 kB, the average message rate is 50 kHz.

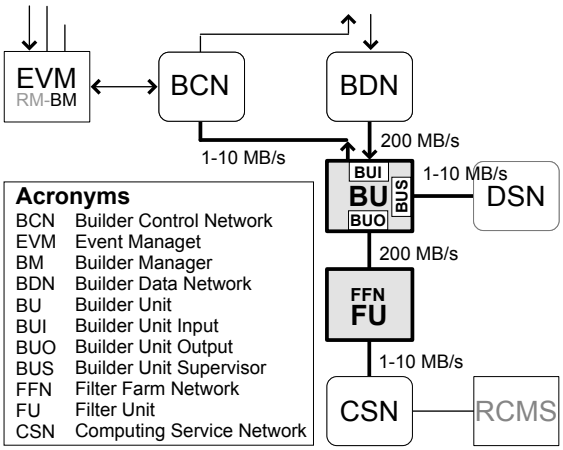


Figure 8-4 Builder Unit internals and network connectivity.

Table 8-1 Requirements on the BU network interfaces. The message rates are parametrized by the Maximum Transfer Unit size *(MTUSize)* for the switched network.

	BDN	**BCN**	**DSN**	**FDN**
Average throughput	200 MB/s	<< 1 MB/s (~ 1 MB/s)	< 1 MB/s	200 MB/s (70 /s)
Average message rate	$12.5\,\text{kHz}\frac{\text{FragSize}}{\text{MTUSize}}$	$12.5\,\text{kHz}\frac{\text{MsgSize}}{\text{MTUSize}}$	< 1 kHz	~ 600 Hz
Traffic type	event building	event building control	configuration, control, monitoring	event filtering

For the BCN port, a large number of messages can be packed in a single physical transfer since the messages are small. Each BU sends requests for new events at a rate of 200 Hz. It also receives confirmations and sends out clear messages at the same rate, resulting in a total message rate to and from the Event Manager of 600 Hz. Assuming a typical message size (indicated with *MsgSize* in the table) of 100 B, an average throughput of 60 kB/s is required. This traffic is negligible compared to that required for the communication with the Readout Units (in parentheses in the table). Event requests to RUs must be sent to 64 destinations, for a total rate of 12.5 kHz. If needed, this rate can be reduced by appropriate message gathering. The required throughput is of the order of 1 MB/s.

The load of the FDN at the BU output depends on the FU operational mode. As already mentioned, the transfer of full events requires an aggregate output bandwidth of 200 MB/s. If multi-step filtering is used, as outlined in Section 8.2 above, this bandwidth can decrease significantly. The message rate at this port, however, is increased of a fraction equivalent to the fraction of events accepted by the first filtering step.

The BU internal communication capabilities have to meet the BDN, the BCN and the FDN port requirements. A single shared interconnect must sustain an aggregate throughput of BUI and BUO of 400 MB/s. The bandwidth to the event data memory must exceed 400 MB/s.

8.3.2 Data Flow

The protocol used by the BUI to communicate with the RU and the EVM is discussed in Section 5.3.2.1, "Event Flow Protocols" and in more detail in Appendix C, "Event Flow Control Protocols in RU Builder". In summary, the BU requests a new event of a given type from the Event Manager. The Event Manager then allocates an event to the BU and returns an Event Identifier. Subsequently, the BU requests data from all the RUs. Once the event is assembled, it can be allocated to a Filter Unit in response to a request for a new event, at which point the BU informs the Event Manager that all resources associated with this event can be liberated.

The BUI accepts messages from the Event Manager containing newly allocated Event Identifiers, and messages with super-fragment data from the Readout Units. The Builder Unit Output accepts two types of requests from the Filter Unit: requests to allocate new events by event type, and requests for sets of event super-fragments. A set comprises a configurable number of Readout Units. A Filter Unit can retrieve more data for a certain event until it decides to discard the event[1]. If an event is discarded, the Builder Unit releases all internal resources associated with it.

The BUO interface is used by the FU to allocate new events, using the allocate message, and to retrieve more data for an allocated event, using the collect message. In both cases, the BUO delivers the requested data to the FU using the take message. The FU sends a discard message to the BUO when the event has been rejected or sent to storage.

The internal interface between Builder Unit Input (BUI) and Output (BUO) comprises five functions, three belonging to the BUI and two to BUO. The interaction diagram for the builder unit internal communication is shown in Figure 8-5. The BUO asks the BUI to build an event of a given type. The BUI re-

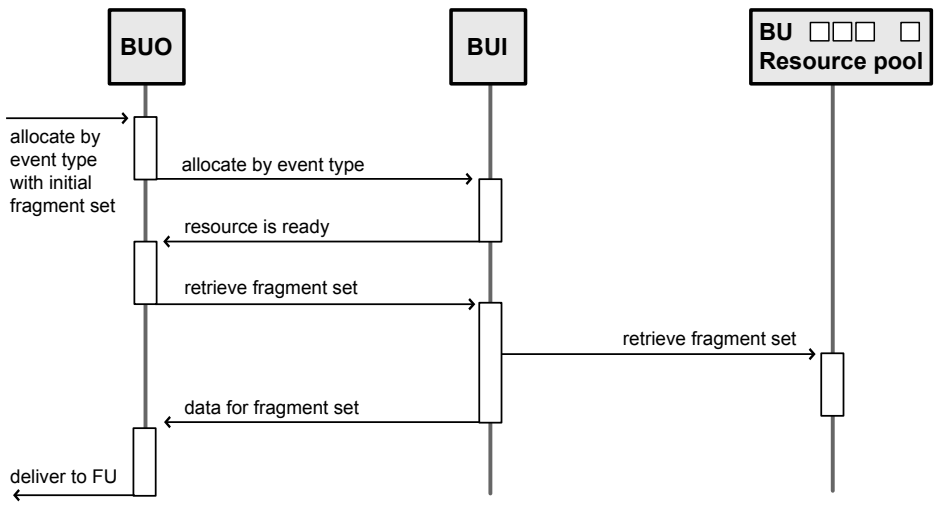


Figure 8-5 Communications among the Builder Unit elements.

sponds to this request by informing the BUO as soon as the event is ready for delivery, also providing the resource identifier that a Filter Unit uses subsequently as an identifier for this event. The BUO forwards subsequent requests for event fragment sets from Filter Units to the BUI using its retrieve interface

1. This protocol is designed to allow multi-step transfer of events to the Filter Unit. However, should the FDN connectivity allow it, the same protocol can be used to transfer entire events with a single request message.

(Figure 8-6). The BUI replies to this requests with the data message that contains all necessary information to send the data to a filter unit. To release all resources that are associated with an event in the BUI, the BUO uses the clear command. A FU cannot request more fragment sets for an event after it has issued a discard.

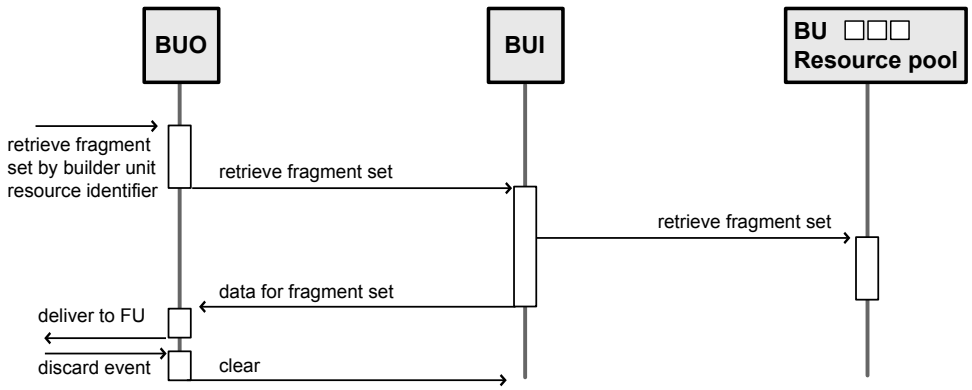


Figure 8-6 Subsequent requests for fragment sets.

8.3.3 Control and Monitor

A Builder Unit is controlled, configured and monitored by the Run Control and Monitor System (RCMS). In the case of the BU there are two types of configuration parameters, those relevant to the networking environment and those relevant to the operation of the system. The operational parameters are listed in Table 8-2.

Table 8-2 Builder Unit configuration: operational parameters.

Name	Purpose
Maximum number of events	Determines the BU memory resources
Fragment set definitions	Mapping of fragment set identifiers to collections of RU identifiers
Event type definitions	Mapping of identifiers to event types
Request gathering	Number of requests gathered in a packet for a given network

The minimal monitoring capabilities of the builder unit comprise:

- Occupancy of the resource pool.
- Event building rate
- Binary dump of all event fragments for a given resource.
- Error counts (by event, by resource, etc.)

Operational anomalies are reported to the run control system. Internally, they are classified according to their severity. Warnings are occurrences that may subsequently lead to a failure. A failure requires external intervention to recover. In this case, it will be possible to store all the information relative to the current state of the application with the monitoring service.

Errors in the communication with the rest of the Event Builder are described in detail in Section 9.2.2.5. A fault tolerance analysis of the BU-FU protocol is presented in Section 9.2.3. For the communication with RCMS, reliable delivery is assumed. The BU must be capable of continuing its operation in case the connection to the RCMS is lost.

8.4 Filter Unit (FU)

A Filter Unit node is an elementary computational unit of the Filter Farm, i.e. it is either one PC or one CPU in a CPU server. Its local resources (e.g. memory and disk space) are limited, but in general it also has access to external shared resources. It is part of a distributed computing environment consisting of the Filter Farm, the Run Control and Monitoring System, and the Computing Services. Shared resources include boot servers, configuration and file servers, and databases containing software, configurations, conditions and other data.

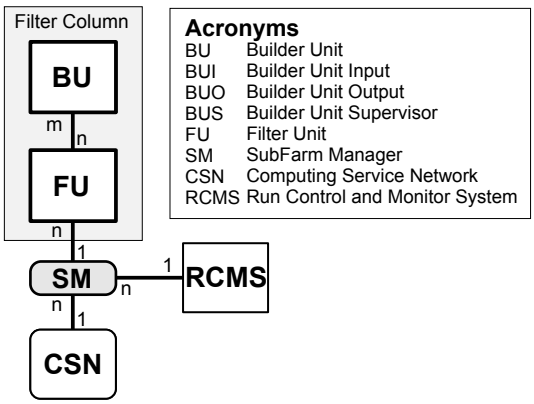


Figure 8-7 Filter Unit and its connections to other DAQ and external subsystems.

The Filter Unit connects to two logically distinct networks (see Figure 8-7): the FDN, which connects it to the BU, and the FCN, which connects it to the Subfarm Manager. The FU makes indirect transactions to the Computing Services and Run Control System through the Subfarm Manager (SM), which, via the CSN, connects to the central computing and storage facilities (see Chapter 16, "Computing Services").

8.4.1 FU Functionality and Elements

The Filter Unit software consists of application components which fit in the layered scheme of the Online software outlined in Section 3.3, "Software Architecture" and discussed in more depth in Chapter 10.

There are four main software components in the FU, illustrated, along with their relationships, in Figure 8-8:

- The Filter Framework which embeds all the FU data access and control layers.
- The Filter Tasks which run HLT algorithms and perform the event selection in the context of the framework. An HLT task allocates an event, reads sets of fragments from that event, processes these data and finally closes the event.

- The Filter Unit Monitor, which, in conjunction with the framework, provides a consistent interface for monitoring of system-level and application-level parameters of the Filter Unit. In particular, it handles monitorable parameters of the HLT Tasks and status information of the FU, collects and elaborates statistical data which are delivered to the relevant consumers via the FCN.
- The Storage Manager which handles the transfer of event data out of the FU. An event accepted by the HLT task remains inside the FU framework until its data have been transferred to a persistent storage.

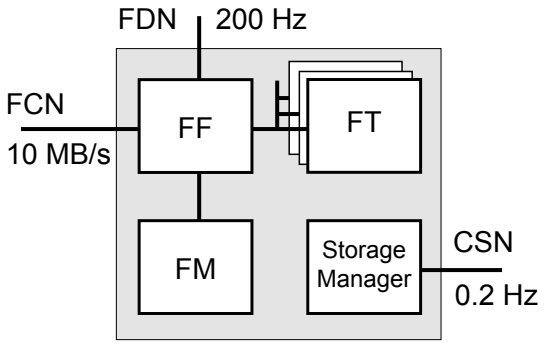


Figure 8-8 Filter Unit Components: FF=Filter Framework, FM=Filter Monitor, FT=Filter Task.

The following sections contain more details on the four components of the Filter Unit.

8.4.1.1 Filter Unit Framework

The Filter Unit framework provides two major functionalities: it receives and dispatches messages from and to the BU, and coordinates the operation of the Filter Tasks by handling configuration and control messages from the Subfarm Manager. At startup, under the supervision of the SM, it sets up the various tasks in the Filter Unit, and configures them according to configuration messages and parameters. These operations are performed by the FU Framework in the context of the DAQ execution framework, and communication is handled by generic core plugins and interfaces as described in Section 11.2.1.

The framework allocates new events and collects data for allocated events using the BUO interface (see Section 8.3.2). It handles response messages from a BU, performing consistency checks[1] and reformatting the contents of the message data. It then maps the buffer sequence containing the event data to a structure that can be used by the HLT application in the Filter Task.

To avoid processor idle time when a task wants to open a new event, the framework maintains a queue of allocated events (Figure 8-9). For each allocated event, an Event object is created and pushed in the queue. When a Filter Task becomes available, an Event object is popped from the queue and its ownership transferred to the task. A request to allocate a new event is immediately issued. The Event object then serves as the “data bus” between the framework and the Filter Tasks. When new data are accessed for the first time by the task, the framework forwards the request to the BU. Completed requests are handed back to the task by the framework via the Event object, which stores pointers to the raw data present in

1. The actual nature and extent of these consistency checks will be defined after finalization of the raw data formats and definition of consistency check performed by other DAQ components. However, the total computation time required, per event, by such tests, must not represent a significant fraction of the total computation time available to reach a filter decision.

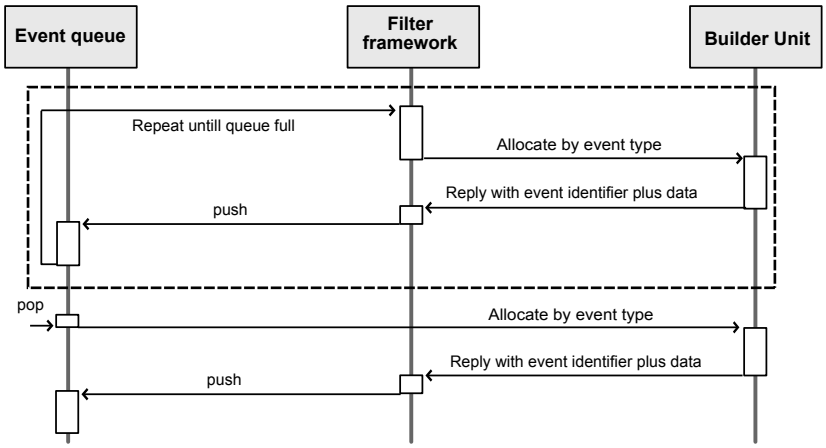


Figure 8-9 Filter Framework managing the event queue.

memory. Since multiple Filter tasks in a task pool can work concurrently on data from different events to reach the HLT decision (see below), multiple events are concurrently under analysis in the framework. For each active event in the FU, a Filter Task has ownership of the corresponding Event object. The deletion of an Event object by a Filter Task means the event is rejected. The framework uses the discard interface of the BUO to indicate that the corresponding resources can be freed and the event removed from the DAQ. For accepted events, ownership of the Event object is handed over to the Storage Manager. Figure 8-10 illustrates how a Filter Task operates on data from a certain Event object.

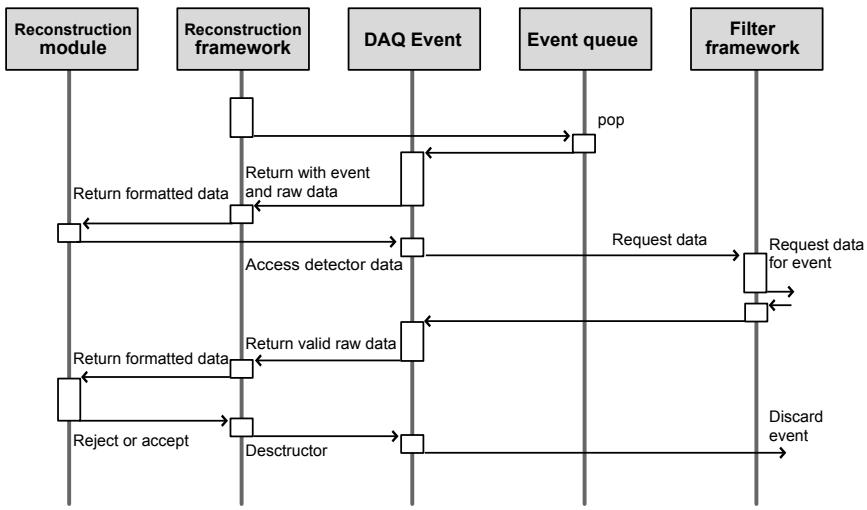


Figure 8-10 Filter Task using the Event object to access raw event data.

The time needed by the BU to service a request for data may be non-negligible on the scale of the available processing time for an event. As an example, a 1 Gb/s link needs approximately 8 µs/kB, which translates into ~2 ms transfer time for “Level-2” information and three times more for “Level-3” information (assuming the HLT example cited above). Furthermore, the BU may be busy servicing prior requests and there may be local fluctuations on the number of “Level-2” accepts causing delay in servicing subsequent “Level-3” requests. As a result, the FU can remain idle waiting for data to be delivered, even though new events would be available for processing. To increase the efficiency of FU usage, several Filter Tasks may be running on the same FU in different threads or processes, so that several events are handled concur-

rently and the CPU idle time is minimized. The actual number of concurrent tasks will depend upon proper optimization of the service and transfer time, and on the specific scheduling policy of the operating system.

8.4.1.2 The Filter Task

The High-Level Trigger selection algorithms, described in detail in Chapter 15, "Physics Object Selection and HLT Performance", run as plugins in the context of the Filter Task. The HLT application consists of an event loop, a series of reconstruction units performing, on demand, parts of the reconstruction necessary to reach the trigger decision, and a number of filter modules that apply the selection criteria. The various modules are integrated through the use of base software which is common to the offline reconstruction [8-1]. Control messages and configurable parameters are dispatched to the application by the local segment of the configuration and control system. For example, the filter application may be started, suspended or stopped by a message from the Subfarm Manager. Whenever a task is available, the framework assigns to it a new Event object for processing. The execution of the HLT application is directed by a steering module. This steering code minimizes the amount of processing time by only running the filter modules relevant for a given event type, or according to its Level-1 information. Following a policy set out in the configuration of the HLT, certain modules can break the processing and unconditionally reject the event (HLT veto). In all other cases, an event is accepted if it has been accepted by at least one of the relevant filter modules. An alternative option which is always available is to process an event until the full chain of filter modules has reached a decision. This can be useful later in the offline processing step which can utilize the result of the filter modules to classify the event as a candidate for different data streams.

After an accept or reject, the filter application may perform additional calculations before dispatching the response to the framework. The HLT application is then ready for a new event, which means all intermediate objects produced in the reconstruction have been deleted.

The HLT application accesses raw data in the form of buffer sequences containing data from several RUs. For each buffer, the framework handles the creation of a map of pointers in the Event object. When the filter application attempts to access data from a specific part of the CMS detector, the map is searched and, in case the data is missing, the framework issues the appropriate request for that data, and blocks while waiting for the data to arrive. If the search is successful, or upon reception of data from the BU, a pointer is returned to the beginning of the relevant portion of data, together with the size of that portion. The set of event fragments requested from the BU may include portions that have not yet been explicitly accessed, according to static or dynamic policies set out in the Filter Unit configuration.

The HLT application formats data fragments into objects. This may involve the invocation of methods from detector geometry, condition and calibration classes etc. The data structures thus created are transient and have the same life span as the event.

All intermediate information produced by the filter application during the processing of an event, can be attached along with raw event data to the Event object. Before an accepted event is handed to the storage manager, it might be necessary to finalize its data content by collecting data that have not been requested so far or have not been delivered. The request and formatting of these data is the task’s last operation on the event.

The HLT application is dynamically loaded into the framework using a standard service method. Once plugged in, the application waits for configuration and initialization and then goes into a paused state. Upon reception of a “start” message from the Subfarm Manager, it enters the event loop. Eventually, a suspend or stop message may be received. A suspend does not stop execution on the current event, but

will block after the latter has been processed. A stop stops execution independently of the status of the current event. Neither command clears the state of the reconstruction, for which an explicit reset message must be issued.

All configurable parameters, which must be controlled from outside the application, must implement a specific *ConfigurationElement* interface to be published to the configuration and control services. Modifications to configurable parameters will only take effect at event boundaries or “run” boundaries, depending on the policy set out in the configuration for that specific parameter.

All the reconstruction and filtering parameters that need to be monitored by the online monitor consumers via the SM, will have to implement a specific *MonitorElement* interface to be published to the Filter Monitor. The HLT application may include monitor components which perform specific reconstruction or analysis used to monitor the application performance, producing complex results such as histograms. The *MonitorElement* interface supports a publish and subscribe mechanism to give access to such complex data types.

In order to allow for immediate notification of error conditions or other situations that need system-wide response or operator intervention, a special *MonitorElement*, called *AlarmElement* is always published and made available to every component of the Filter Unit, including the HLT application. Alarm messages posted to the *AlarmElement* are queued, so that no alarm message is cleared before having been notified to all subscribed clients (and having been acknowledged by all subscribed clients, see below).

8.4.1.3 The Filter Monitor

The Filter Monitor serves several different purposes: it caches monitor information that has been previously published to it by one of the Filter Unit components, keeps track of clients which subscribed for updates, and performs updates by delivering monitor information to the subscribers. It is also responsible for notification of *AlarmElement* messages and for feeding the event display queue on the relevant clients.

Updates to monitor clients are performed by periodically waking up an update thread or process at intervals set out in the configuration. The update thread subsequently dispatches the data content of all *MonitorElement*s which varied since the last update to the relevant clients. Updates of MonitorElements do not need to be acknowledged by the consumers.

Notification of alarms is preemptive: all the *AlarmElement* clients are notified as soon as the alarm is known to the Filter Monitor, and reception has to be acknowledged before processing can continue. It is possible, by issuing an appropriate command from Run Control, to mask certain types of alarms.

The Filter Monitor periodically notifies the subscribers of the special *EventDisplay* element with the full data content of the last event marked for display (according to the relevant configuration parameters).

The Filter Monitor interface allows a client to retrieve the full list of available *MonitorElement*s, and to register the client for update notifications (subscription). These interface extensions are implemented according to the principles and using the facilities described in Section 11.2.2. The Filter Monitor is capable of notifying any subscribed client which complies to the specification for control and monitoring communication using the standard web protocol, as outlined in Section 11.2.5.

8.4.1.4 The Storage Manager

The Storage Manager takes ownership of events which have been accepted by the HLT, and are therefore marked for storage. An accepted event consists of the entire set of detector raw data as delivered to the

FU[1], and of all the object collections that have been attached to it as a by-product of the HLT reconstruction and selection. The event is then stored remotely on the Subfarm Manager using a specific interface and protocol. This protocol will match the one chosen for persistent storage of event data in the context of the offline framework. The actual implementation of the Storage Manager will therefore depend on this choice [8-2]. After successful delivery of an event to the persistent services of the Subfarm Manager, the Storage Manager deletes the event, thereby triggering the issue of a discard message to the BU.

8.4.2 Internal Error Handling

In the event of an error condition in the HLT application, an exception is raised. The exception is caught by the first element of the filter application which is able to handle it. In case the filter application is not able to recover from the error condition, the unhandled exception is passed on to the framework which then takes action by restarting the faulty task and reassigning or discarding the event(s) on which the faulty task was working[2]. A recovered error condition (e.g. a task crash and subsequent restart) is logged to the SM via the monitoring interface. A failure of one of the components, or the failure to recover from an error condition after a certain number of attempts, will cause the FU to transition to the failed state (see Section 11.2.2 and Section 12.3.1, "State Definitions and System Synchronization"), and the failure will be logged, if possible, to the SM. No error condition of the HLT application shall leave the FU in an undefined state.

Failure to transfer an event marked for storage to the persistent services of the SM (timeout) results in an exception being raised by the storage manager. Since the fault may be transient, an alarm is first issued to the SM and the failed transaction is reiterated. After a certain number of attempts, a connection failure is assumed. Events in the queue are flushed and the FU transitions to a paused state, waiting for operator intervention.

8.5 Farm Control and Monitor

The configuration, setup and management of a large farm of computers, both from the hardware and operating system point of view, is a common problem in large High Energy Physics experiments. In this respect, the Filter Farm, consisting of “Commercial Off the Shelf” (COTS) components, will, as far as possible, adopt commercial hardware and software tools. These products, and possible home grown software if needed, must satisfy requirements which are common to online and offline farms, and to GRID production centers [8-3]. Therefore, common solutions are expected to be adopted wherever possible.

This section is devoted to the requirements and constraints on Control and Monitor operations which are specific to the Filter Farm. For each specific scenario, a sequence of operations is described, with emphasis on operations that are critical for efficient data taking.

The Farm will be arranged, as previously stated, in a hierarchical structure consisting of a number of smaller, independent subfarms. Any configuration, control or monitor operation will, in general, follow a tree structure, as exemplified in Figure 8-11.

1. Stripped of all DAQ-specific header and trailer information.
2. Depending on the type of failure, the event can be tagged as corrupt and not reassigned for processing. It will be possible to store any event causing a failure condition in the filter application for further analysis.

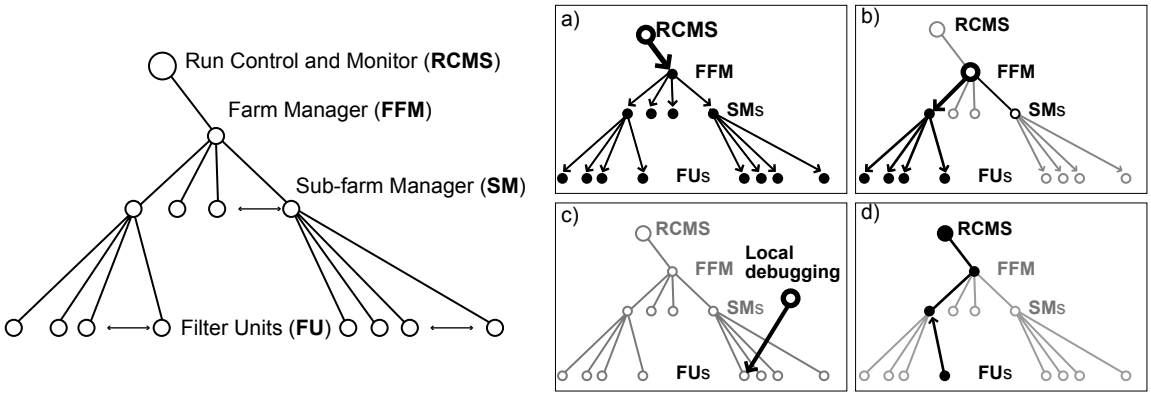


Figure 8-11 Filter Farm hierarchy structure and message flow. a) A Run Control command is distributed to all FU nodes. b) The Filter Farm Manager sends a command to a specific subfarm. c) A command is issued directly to one of the FUs for debugging purposes. d) One of the FUs reports an alarm or updates a monitored parameter in the RCMS.

8.5.1 Subfarm Manager (SM)

The Subfarm Manager (SM) is the supervising element of a subfarm, representing it, to all external entities, as a single element of the DAQ, which can be controlled and monitored in unison. It also represents the subfarm's access point to computing services.

The core of the SM is a generic application server, into which various services can be plugged to maintain the configuration of the nodes in the subfarm, relay control messages coming from external entities such as the RCMS, keep track of the state of each node, and collect, process and present monitor information on behalf of external consumers.

Besides the software services mentioned above, the SM hardware hosts the mass storage systems where the reference copy of all the necessary software and external data structures (such as calibrations and constants) are resident. This includes the operating system, general purpose software libraries as well as HLT-specific libraries. This storage system must have a high level of redundancy. The SM machine also serves as the central boot node for all the subfarm machines. Finally, the SM hardware hosts the mass storage system for local data storage[1]. In this last function, it represents the point of access for data transfer to the central storage system and the offline production farms.

8.5.1.1 SM Functionality and Elements

A block scheme of the Subfarm Manager is shown in Figure 8-12. The various services are coordinated as plugins by the application server to provide a unique point of access for RCMS and monitor clients, and to give a uniform interface to the various functions of the subfarm. A Subfarm monitor handles the subscription and collection of monitor and alarm information from all the FU nodes. Monitor information is stored in a transient cache for local elaboration and distribution. A Monitor and Alarm engine can elaborate and analyse this information, as well as state information of FUs, SM and the local storage. Thresholds on various observables related to these components can be set here to produce alarms and warnings to be delivered to Run Control or other clients. Alarms originated by the FU or raised locally, can be

1. The controller machine for the local mass storage system may, for the sake of robustness, be physically separated from the SM, although being logically part of it.

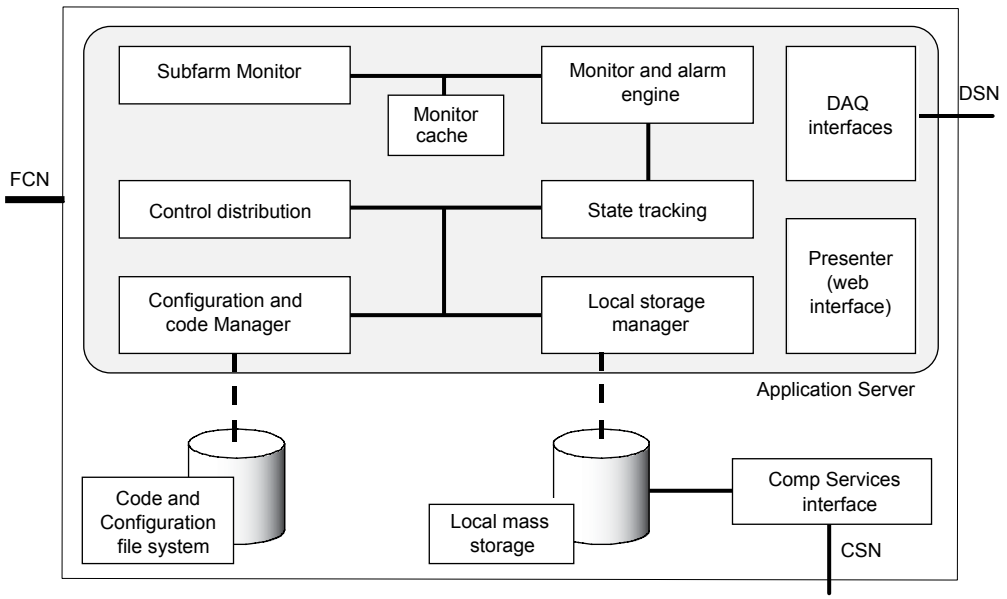


Figure 8-12 Subfarm Manager block scheme.

masked in the Monitor engine. Control and Configuration messages are distributed upon request from RCMS by the corresponding service, and the State tracking service collects responses to monitor the real-time state of the system and acknowledge state transitions to the RCMS. The SM supervises the file-system containing the reference code, configuration data, and the replica of the Run Condition and Calibration Databases, through the Configuration manager, and monitors the state of the local mass storage on behalf of Run Control. To give uniform access to the Farm resources, the SM control and monitor interfaces to the RCMS follow the DAQ “standard” and are identical to the corresponding interfaces of the FU.

The SM Presenter elaborates dynamic representations of the monitor and state information for browsing by external clients via the standard Web interface.

8.5.1.2 Error Handling

The Subfarm Manager collects information about the state of the FUs in the subfarm. The FUs update this information with a frequency which is defined in the configuration. If a FU fails to communicate its state within a certain time period, it can be polled by the SM. In lack of response from a certain node, the node’s state is set to failed in the SM. Depending on a policy set out in the configuration, the SM may or may not try to restart the failed FU. The change in state of the subfarm is immediately communicated to the Farm Manager. When the number of active nodes in a subfarm falls below a quorum set in the configuration, the SM raises an alarm flag. In this event, some of the Builder Units may not be able to cope with the full input rate, thus causing a performance deterioration of the whole DAQ. The Farm Manager can reconfigure the system, reassigning the relevant BUs to different Filter Units.

Thanks to the usage of asynchronous protocols for communication with the FUs, the coupling to the SM services is sufficiently loose to guarantee that the subfarm may continue working in the event of a failure of one of the services, or of the whole server. On the other hand, such a failure will make it impossible to control and track the state of subfarm nodes, or update run conditions or parameters. A backup system with Subfarm Manager functionality can be made available to take control in the event of an SM software failure. SM hardware failures, on the other hand, affect the subfarm functionality because of the tight cou-

pling introduced by the local output storage. This case has been analysed in Section 8.4.2. See also footnote 1 of Section 8.5.

8.5.2 Farm Setup for Data-taking

When the RCMS initiates the setup sequence of the DAQ system to prepare for data taking, each of the FU nodes will create an instance of the FU framework. From then on, the state of the FU is controlled and known at all times to the SM. Each SM, in turn, reports the status of all its nodes to the RCMS. When the farm is configured, the HLT application is downloaded to the SMs. The SMs in turn serve the binary to their nodes, which are subsequently requested to upload it to memory, including all relevant libraries.[1] When the system is initialized, bootstrap information (e.g. geometry, run conditions, calibrations etc.) are downloaded by the SMs from the appropriate servers and then distributed to the FU applications on request. Each node keeps a private copy of this information. At the end of the initialization phase, when all FU nodes report being ready, the subfarm goes into ready state and the transition is reported to Run Control, which may then start data taking.

The setup and control of the HLT tasks inside the individual FU nodes is the task of the Subfarm Manager under the supervision of the RCMS. This includes the following:

- bootstrap of the reconstruction application
- control of the state of each of the HLT tasks
- HLT parameter setting
- upload of a trigger table
- Resetting of a task after an exception
- Notification of changes in the run condition and/or calibration db's
- Error detection (and generation of alarms from limits)
- Monitoring of a task's vital parameters (counters, accept rates, trigger and detector maps)

8.5.2.1 Upload of a New Trigger Table in the Filter Farm

The HLT tasks accept and discard events on the basis of a trigger table of relevant trigger channels, which lists the physics objects and quantities, and the cuts to be applied to them. During a data taking period, the trigger table is adjusted to account for variable luminosity, novel physics channels, detector malfunctions (e.g. dead channels, resolution deterioration) and other parameters. Furthermore, during a single machine fill, a broad variation of luminosity is possible and it may be necessary to switch from a high- to a low-luminosity trigger table while data are being taken. The loading of a new trigger table can be disruptive, since a certain amount of time is needed to update all the FU nodes. During this time, the system is in a mixed state where a fraction of the FUs uses the old and a fraction the new trigger table.The change of the trigger table must be unambiguously logged, so that during offline analysis it will be possible to associate each event with the relevant trigger table. If the trigger table change requires processing to be temporarily

1. This may in practice mean: 1) the Farm Controller gets a copy of the new software from a file server. It then instructs the Subfarm Managers to make a local copy (e.g. the relevant disk is nfs mounted by all SMs). 2) The SMs, when done, instruct all their nodes to get the files (e.g. the SM relevant disk is nfs mounted by all FUs in the subfarm). If there has been no code update, the message only triggers a file read from the local disk on each Filter Unit.

stopped on the subfarm, this interval has to be kept as short as possible in order to keep the DAQ efficiency high.

8.5.2.2 Handling of Run Conditions and Calibration Databases

The HLT tasks need information about run conditions (detector geometry, detector status, dead channels, alignment constants, beam characteristics etc.) and detector calibration constants. Some of these informations may vary only seldom or not at all (such as the geometry) and can be accessed only once at startup, while others may vary on a relatively short time scale (e.g. fine beam alignment). Many of these informations are cached locally on the FU at loading time. To reduce the load on the central information servers, only Subfarm Managers are allowed to download information centrally, and then redistribute it to their local nodes. Care must be taken to avoid mismatches between different subfarms, such that they use different calibration sets or different dead channel lists. However, it must be possible, after an event has left the farm, to trace back all constants and conditions that were used to select it. The “standard” unique run and event numbers will be a shorthand for a structure pointing to a certain combination of conditions and calibrations constants.

8.5.3 Online Farm Monitoring

During data taking, it must be possible to monitor vital parameters of the filter farm from the individual node level to the overall farm level. These parameters range from simple accept and reject rates to complex histograms of raw data or reconstructed quantities. For some of them, an aggregate view is sufficient, whereas for others, it is necessary to be able to collect information from a single subfarm or node. In some cases, a certain delay in data collection is acceptable (such as event displays, etc.), whereas other quantities must be monitored in quasi-real-time. The collection of information by the monitor clients must not interfere with normal running of the HLT tasks. Therefore, wherever possible, monitor data is collected and elaborated by the SM on behalf of other consumers. When this is not possible, data are accumulated by the Filter Units and delivered to consumers which must then elaborate them autonomously.

A possible, non exhaustive list of monitorable quantities is:

- Current run condition and calibration constants by node
- Current reconstruction parameters and trigger table by node
- Event latency distribution by node, where the event latency comprises the time the event is in the event queue and the processing time
- Recoverable exception conditions detected by the reconstruction code
- Individual L1 accept rates by node, by subfarm, and aggregate
- Processing time distribution for events broken by event and trigger type
- Individual accept and reject rates of each selection algorithm and/or path in the HLT
- Detector maps for raw data, basic reconstructed objects, and HLT objects
- Instantaneous luminosity
- Beam position and beam parameters as seen by the tracker
- Characteristics (position, energy, momentum) of physics objects used by the HLT (e.g. mean Pt of electrons, photons and muons, mean total and missing transverse energy in jet events, reconstructed mass of Z boson etc.)

- Full representation of raw events and trigger objects sampling the whole farm.

8.5.4 Installation and Testing of HLT Code

8.5.4.1 HLT Coding Guidelines

The code that runs in the filter farm to perform HLT selection has to follow strict guidelines concerning the usage of resources (network, memory, CPU time) and the handling of errors. It also has to undergo strict functionality tests prior to its actual deployment.

Any access to external resources by an HLT task results in a large number of operations in the overall farm. No direct access to external resources, besides those defined by standard run condition and calibration access interfaces, is thus allowed. The use of external software, which internally makes use of operating system resources (e.g. network or shared resources) is restricted. As an example, the adoption of libraries which covertly make use of shared memory or temporary disk files must be avoided.

The memory footprint must be under control at all times. A candidate filter module must be profiled for memory usage before being included in a test application. The CPU time usage under “normal” conditions, including the CPU usage distribution, must be specified and measured on “real data”.

All exceptions a module can raise must be declared in the module specifications, so that they can be handled properly.

Code to be included in a filter application must follow the standard CMS coding guidelines and quality checks [8-4] and is subject to a standard process control [8-5]. Additional requirements will be the subject of a specific guideline document.

8.5.4.2 HLT Code Burn-in Sequence

Code for HLT undergoes the same versioning and release sequence as the offline code. A major version of the filter application will only include modules released with that major version. No new modules can be added. Bug fixes to existing modules will increment the minor version number. When a new filter application is ready for release, it is first validated in an offline farm against a known dataset. If the offline validation is successful, the new filter application is tested in a special section of the filter farm.

Builder Units in the test partition are assigned a special event type called duplicated event. The Event Manager tags random events to be duplicated, with a frequency defined in the configuration. A duplicated event is assigned once to a BU in the farm section running the production application, and then once more, to a BU in the test partition. Alternately, a single subfarm can be split into a production and a test section, and the same events allocated by the BU to one FU in the production and one in the test section. Duplicated events then go into a special output stream. Validation is performed in the offline farms using data from the test and production streams.

8.6 Prototypes and Results

8.6.1 Builder Unit Prototype

The BU prototype results were obtained running the software components included in the XDAQ data acquisition framework (see Chapter 11, "Cross-Platform DAQ Framework (XDAQ)"). The BU implements the interfaces described in Section 8.3.2.

A first setup consists of four PCs with a single 64bit/66MHz PCI bus and a Pentium III processor for the RU, a fifth PC running a BM, and an additional PC for the FU. A dual 64bit/66MHz PCI bus/dual CPU PC (Dell Poweredge 1550) was used for the BU. All machines run the Linux operating system and are connected through a Myrinet 2000 network, accessed using the Myricom GM communication library[1]. In the setup, the RU acts as a data source for event fragments and the FU as data sink that continuously requests complete events. The FU request messages are gathered in groups of 256 and the RU delivers fixed-size fragments of 8 or 16 kB, so the total event size is 32 or 64 kB respectively. The resource pool for event fragments in the BU is configured at 256. The throughput is sampled at the FU by measuring the time needed to assemble 10^5 events. It is 194 MB/s and 204 MB/s respectively for 8 and 16 kB fragments which satisfies the required throughput of 200 MB/s.

8.6.2 Filter Unit Prototypes

8.6.2.1 Tests of BU-FU Communication Protocol

The BU-FU communication protocol has been implemented for this test within the XDAQ data acquisition software framework. The BUO feeds dummy event data to several FUs over a Gigabit-Ethernet switching network using TCP/IP. Event data are segmented in chunks corresponding to a given HLT filter procedure. Chunk average sizes correspond to current estimates of raw data sizes from the various detectors, and the actual size is extracted from a log-normal distribution to account for fluctuations. A parametric Filter Task in the FU implements a step-wise filtering procedure where the average service time of each step, and the corresponding rejection factor, can be set to match a specific HLT configuration. The service time fluctuations are accounted for by extracting the actual times from a log-normal distribution.

The test shows that the BU-FU protocol is adequate. The BU can operate at an event rate of 200 Hz for an average event size of 1 MB, if the average HLT service times are set appropriately to match the input rate. The system reaches a steady state which depends only weakly on the level of fluctuations in the event sizes and fragment times. As an example, for the two-step filtering described previously, a steady throughput of 66 MB/s is observed at the BU output, when the 10 FUs connected provide an average total service time of 50 ms/event. This is achieved by setting the average “Level-2” service time at 40 ms with a rejection factor of 100. The subsequent “Level-3” step can then be performed in an average of up to 250 ms.

8.6.2.2 Prototype HLT Application

To demonstrate the feasibility of operating an offline reconstruction procedure in the context of the Filter Unit, the integration of the CMS reconstruction software framework (COBRA [8-1]), and of the CMS ob-

1. For an overview of Myrinet network products and related protocols see Appendix A.2, "Myrinet Technology and Products"

ject oriented reconstruction (ORCA [8-6]) in the Filter Unit prototype has been carried out. The raw-data access interface (Event object data bus) and the DAQ-specific event loop structure are implemented as extensions to the offline framework. Raw event data in an arbitrary format is read in by a standalone BUO, transferred over the network using the protocol described above, and accessed by a generic ORCA reconstruction module. To achieve this, a skeleton implementation of the raw detector data formatting was also prototyped as a replacement library for ORCA. The prototype HLT application was successfully operated and controlled in a Filter Task through the Filter Unit framework. Final design and implementation of configuration and monitor layers, as extensions to the offline framework, is ongoing.

8.6.2.3 Prototype Reconstruction Code for HLT

The details of the current reconstruction software framework used by the HLT algorithms can be found in Chapter 15, "Physics Object Selection and HLT Performance". For the purpose of this chapter, two key measurements are worth mentioning, namely that the current software requires approximately 150 MB of memory and roughly 400 ms/event passing the Level-1 Trigger requirements described in Chapter 14, "Detector Simulation and Reconstruction".

8.6.3 Subfarm Manager as Control and Monitor Server

A Subfarm Manager prototype software, based on the Apache Tomcat servlet container [8-7] has been demonstrated in a small scale setup. It integrates the online software prototype services described in Section 11.2.7, "Configuration, Control and Monitoring Components", and is able to distribute configuration and control messages using the SOAP protocol. State tracking and monitor information collection are implemented as servlets, and the server is able to present dynamic information about the system to a standard browser. Among the future activities, larger scale tests of configuration and command distribution will be performed. A complete implementation, which will also take into account surveys of available products and work on analogous components for the Run Control System, will be completed for use in the subfarm test-bench (see Section 8.7.3).

8.7 Technology Outlook and Schedule

As mentioned numerous times throughout this Technical Design Report, the design of the CMS DAQ is based on the use of Commercial Off-The-Shelf solutions (COTS) to the greatest possible extent. Whatever CMS-specific hardware has to be developed, it is usually placed close to the detector, where the LHC-specific or the CMS-specific requirements are such that COTS systems can either not be procured at all, or can be procured but at relatively high cost. The elements of the Filter Column, however, are the most clear case where commercially available hardware can satisfy the requirements. An implementation of the subfarm via a number of standard Personal Computers (PCs), possibly with dual CPUs, and a 12-24 port Gigabit Ethernet switch acting as the FDN is going to be both inexpensive and easily maintainable, given the wide availability of these elements. This is especially true when one considers the year 2006 as the actual time scale when much of the DAQ hardware will be procured.

The actual design of the Filter Column can accommodate essentially any technology for the intelligences involved (i.e. the BU and the FU) along with any technology for the switched interconnect. The actual choice of BU will be influenced by the technology chosen for the Builder Data Network that connects the BUs to the Readout Units: BU hardware will have to be compatible with the associated Network Interface Cards of this network. Nevertheless, given the observed explosion of industrial offerings in this general

area, the Filter Column components can easily profit, with minimal disturbance, of most of the announced technological improvements.

8.7.1 Hardware

8.7.1.1 Builder Unit

The basic implementation of the BU is a PC with two independent 64-bit/66 MHz PCI busses and 1GB memory. The CPU is not an issue given that the BU does not process the event data, but only assembles the fragments into full events.

It should be pointed out that the BUs in the CMS DAQ are not expected to be all identical machines. As explained in Part 1, the CMS DAQ will be installed in a number of steps that will be timed to facilitate the operation of the experiment, given the accelerator conditions and the financial resources available. The most likely scenario is therefore one in which the RU Builders are procured at different times. Given the planned BU implementation, it is clear that procurements separated in time by periods of ~6 months to a year will result in different hardware being installed as BUs. With the current design and planned implementation there is, a priori, no anticipated effect from this.

8.7.1.2 Filter Data Network Technology

The strategy for deciding on a FDN technology is analogous to that for deciding a BU implementation. A standard commercial switched network can satisfy all requirements. As such, the most widely available and possibly widely deployed network technology at the time of the procurements of the Filter subfarms will be chosen. Already, as this TDR is being written, Gigabit Ethernet technology is within reach, in terms of financial resources, for such a wide deployment. It is anticipated that by 2006-2007, the associated costs will have dropped even further. Should Ethernet, in the next few years, become an obsolete technology, the FDN will clearly utilize the new technology of choice. In brief, the system design is very flexible, and it allows the use of any FDN technology.

8.7.1.3 Filter Unit Processors

The Filter Units, and in general the filter farm, will require a large number of inexpensive PCs. A potential exception to this is the hardware for the Subfarm Manager, where a larger CPU server may be more appropriate, or even necessary, given that the SM has to also serve as the basic source of code and configuration data for the FUs in the farm.

8.7.2 Software

8.7.2.1 Operating System

With very large numbers of processors installed, the costs of a commercial OS can be significant. The advent of Linux, with its high reliability and wide deployment in High Energy Physics, as well as its standard UNIX interfaces to networking, has provided a “natural” solution for the Operating System, at least for the Builder Units and the Filter Units. At this point in time all indications are that the development and further deployment of Linux will continue in the relevant time scales.

8.7.2.2 Other Software Components

All basic external software packages used in the current prototypes are either commercial or constitute part of the Open Source software initiative. All technologies utilized, e.g. XML, SOAP, TCP/IP are standard and widely available. Thanks to the layered architecture of the Online software, with middleware screening applications from lower-level core functions, an easy migration to new standards is anticipated.

8.7.3 Subfarm Test-bench

To continue verifying on a larger scale the architecture and design of the farm and related software, and taking into account the general principles stated above, a test-bench composed of current COTS computers and networks, with the level of complexity expected for a subfarm, will be assembled. This system will first of all provide the testing ground to verify the scaling properties of some of the configuration, control and monitor features described throughout this chapter. It will also be used to evaluate candidate products and technologies for hardware and system management, concentrating on specific aspects of an online environment. Finally, it will be exploited for stress tests of reconstruction software, using simulated event data with realistic formats, as they are being finalized by the detector groups. Ultimately, it will be part of a complete, scaled down acquisition system which will implement the full chain of the DAQ. The test-bench will represent an intermediate step for validation of all the software components to be subsequently deployed in the CMS Filter Farm.

8.8 Summary

In this chapter the functionality and the design of the Builder Unit and the Filter Farm are described. The events which are assembled in the BU are processed in the Filter Farm in order to reach the HLT decision. This can be done in multiple steps to reduce the data traffic between the BUs and the Filter Units.

The Filter Farm is divided into independent subfarms in order to ease configuration, control, and monitoring of the large number of Filter Units. This modular concept results in a scalable system and improves the reliability against failure of single nodes.

In a BU protoype the required data throughput of 200 MB/s has been achieved. The interface between the BU and the Filter subfarm has been implemented in another prototype and the developed protocols have been successfully tested. A test-bench will be built to further analyse different software implementations of the Filter Column components, to evaluate system management tools, and for performance measurements of reconstruction and selection algorithms.

8.9 References

8-1 V. Innocente, L. Silvestris and D. Stickland, on behalf of the CMS collaboration, “CMS Software Architecture. Software framework, services and persistency in High-Level trigger, reconstruction and analysis”, Comput. Phys. Commun., 140 (2001) 31;
Coherent Object Base for Reconstruction, Analysis and simulation (COBRA), see http://cobra.web.cern.ch/cobra/

8-2 “Report of the LHC Computing Grid Project: Persistency Management RTAG”, CERN-LCG-2002-006, 5 April 2002.

8-3 “DataGrid Work Package 4: Fabric Management, Report on Current Technology”, DataGrid-04-D4.1, 18 May 2001;
The DataGrid Project , see http://eu-datagrid.web.cern.ch/eu-datagrid

8-4 J. P. Wellisch for the CMS software group, “The CMS Coding and Design Guidelines”, CMS Note 1998/070.

8-5 The ISO/IEC Joint Technical Committee 1 (JTC1), Subcommittee 7 (SC7) Working Group 10 (WG10), Terry Rout (ed.), ISO/IEC DTR 15504 Software Process Assessment;
“Report of the LHC Computing Grid Project: Software Management Process RTAG”, CERN-LCG-2002-009, 6 May 2002.

8-6 ORCA, Object-oriented Reconstruction for CMS Analysis, see http://cmsdoc.cern.ch/orca/

8-7 The Jakarta Project: Apache Tomcat v. 4.0; http://jakarta.apache.org/tomcat/index.html

9 Event Builder Fault Tolerance

Important elements of the DAQ system are the error-handling procedures and other mechanisms that allow it to operate with a sufficiently high efficiency. This chapter discusses various aspects of redundancy and fault tolerance of the DAQ architecture and design. It is aiming to demonstrate that the two are expected to lead to a system that can deliver high-quality data with high efficiency.

Dependability is defined as the probability to accept and assemble the next event. A high operation efficiency and dependability is the result of three conditions:

- rarity of cases at which the system has to stop its operation,
- quick identification and equivalently fast recovery procedures for those error conditions that affect the quality of the data in a significant way,
- tolerating and possibly ignoring error conditions that have only a minor effect on the data quality.

The key element to achieving the above three elements is the correct categorization of failures in the system. The purpose of this analysis is thus to determine the failures that should be avoided, the ones that should be tolerated and the ones that can be ignored. This categorization depends on the mean time between failures, the down time required to recover from the resulting errors as well as the cost of tolerating or avoiding them. Clearly, very rare failures that happen once a year can be ignored, if the system can be up and running again within about an hour with a simple system reset.

On the other hand, a more frequent transient fault [9-1] needs special attention. A fault is transient if it exists for only a short amount of time and disappears without an explicit recovery procedure. A transient fault might lead the system into an incoherent state which can be permanent. If this state is not detected and handled properly, it can eventually lead to a system failure which will, in turn, degrade the system's dependability and reliability[1]. A system with the ability to recover from inconsistent states caused by transient faults without outside intervention and regardless of the origin of the fault origin, is said to be a self-stabilizing system [9-2].

The plan to achieve a fault-tolerant design for the DAQ has been the following:

1. Catalogue an extensive list of possible faults together with their effects on the system, as well as possible detection and recovery methods.
2. Supplement the CMS-DAQ system design in places that have been identified as potential sources of faults, by adding additional functionality or improved components.
3. Establish that the newly supplemented design is fault-tolerant by demonstrating that in the resulting system all transient faults are handled in real-time and that, after relatively short periods of time, the system resumes its correct operation.
4. Test the correctness of the actual implementation of the established fault-tolerant protocols by injecting faults to prototype software and hardware developed in the context of the CMS-DAQ R&D.

Before giving the extensive list of possible failures and their detection and recovery methods for each sub-system, it should be noted that, there is no single point of failure in the DAQ system which would bring down the entire system until it is repaired. The proposed system, with its modular design consisting

1. Definition: The probability that a system will perform its specified function, without failure under stated environmental conditions, over a required lifetime.

of independent RU Builders[1], will continue to operate, albeit with less performance, after identifying and isolating the faulty component. If one RU Builder fails the system will continue to build and filter events with the remaining RU Builders after re-configuration of the FED Builders[2]. If a FED Builder fails, the system will also continue to build and filter events from other parts of the detector, again after reconfiguration of the RU Builders. The only single point of failure prohibiting further operation is the FED Builder for the Level-1 Trigger (see Section 4.2.5, "Data Flow in the FED Builder"),

The list of faults identified in the DAQ architecture and event building protocols is discussed in Section 9.2, "Error Analysis for the Event Building Protocol" along with the detection methods and the associated reaction to the faults.

9.1 Fault Tolerance Requirements and Design Choices of the DAQ System

This section lists the design choices and suggested fault-tolerance requirements of the DAQ system and discusses the underlying reasons for these choices. Some of the items listed below are more of a design choice than a requirement. The main motivation behind these choices is to maximize system availability. When the Event Building networks are fully loaded, any attempt to recover corrupted event data will decrease system availability such that the product of the system availability and the system reliability at best stays constant. As long as the mean time between failures is large, the number of events sacrificed due to error recovery will be acceptable.

1. All error states in the DAQ system shall be detected, logged and of the system shall be brought back to a consistent state.
2. Sub-systems shall have self-stabilizing behavior for all external faults.
3. Sub-systems shall tolerate transient faults.
4. All data corruptions shall be detected and logged.
5. Application processes shall not assume that the inter-process communications are reliable.
6. Sub-systems shall be capable of performing forward-error-recovery (go to a state to that the system would have gone to if the error had not occurred).
7. Sub-systems are allowed to execute exception-handling methods which corrupt event data of a limited number of events in order to maintain system availability.
8. Sub-systems could perform backward-error-recovery, however only if the system availability would not decrease.
9. Sub-systems could implement fault masking[3], provided this does not decrease system availability.
10. The system shall continue to operate with reduced performance in the presence of faulty components.
11. All persistent failures (e.g. a node crash) shall be detected and reported to the Problem Solver.
12. The system shall be reconfigurable in order to mask faulty components.

1. See Section 4.3, "RU Builder" and Section 4.4, "Partitioning".
2. See Section 4.2, "FED Builder".
3. Fault masking allows the subs-system to ignore specified faults.

13. Sub-systems shall provide monitoring and diagnostics services to facilitate the localization of faults.
14. Sub-systems shall monitor their resources and report any shortages to the Error Manager.
15. The event number should not be reset before O(1000) triggers[1] since the last reset to avoid having two events with the same event number in the Readout Unit Input module.

9.2 Error Analysis for the Event Building Protocol

This section gives a general description of the failure detection and exception-handling methods which will be implemented by functional units of the DAQ system. A detailed list of all failures that are handled by a unit, the details of their detection, and exception-handling procedures are described in the following sub-sections. Supplementary information is given in Appendix E, "Fault Tolerance of Distributed Systems".

Each node could produce a faulty message as a result of input failure and/or internal faults. For each message there are three possible faults, namely the loss of the message, a spurious message and the corruption of some information contained in the message. Each node in the system shall implement the following fault-tolerance behavior to handle input failures:

- A spurious message can be detected by comparing the message context either with the information stored in the receiver if the message is a response to a request, or to information it expects. For example, a event number received by the Readout Units from the Readout Manager must increase by N (or have a reset value), where N is the number of RU Builders in the system.
- All messages shall contain sufficient information to allow identification of spurious messages. Standard encoding techniques will be used to ensure that a value error in these fields will not cause a false detection of spurious messages. All detected spurious messages will be dropped by the receivers, thus they will not change the internal states of these receivers.
- Similarly, any message with a detected-value error can be safely dropped at a receiver. Thus, they appear to a receiver as omission failures. Well known techniques such as CRC checks will be used to detect corruptions during transmission. Whenever possible, reliable communication protocols will be used to recover from value-errors incurred during transmission (see Section 6.6.3, "Reliable Broadcast Protocol for the RCN"). It is noted that some of the messages with undetected value-error will be identified as spurious messages. Undetected spurious messages will have the same effect as an undetected value-errors.
- Omission failures will be detected by “watchdog timers” or local sequence numbers. Whenever a watchdog timer expires, the node will perform a forward recovery and release all its resources allocated to the event in question. All messages arriving late (after a watchdog timer expires for a given transaction) will be treated as spurious messages and will be dropped. In order to avoid false detection of omission failures, watchdog timer expiration will be long enough to cover most transmission delays.

Transactions with undetected value-error can not be differentiated from error-free transactions. Therefore, a node which receives a message with undetected value-error will not deviate from its standard behavior. However, it could generate omission failures, and spurious or corrupted messages.

1. Actual number of triggers between reset will depend on the concurrent events being build in the RUI which depend on the memory available for this purpose on the final system.

9.2.1 Failure Modes of the Front-End System

Front-End Systems provide data fragments from various detector elements to the DAQ system for event assembly. As with any node in the DAQ system, front-end nodes can also fail and produce omission failures, spurious or corrupted messages. Value-errors in the detector data have no effect on the correct operation of the DAQ system. Of course, in this case, the assembled event will contain the erroneous data.

Data from different detector elements belonging to the same bunch-crossing will be identified by two numbers, a event number and a bunch-crossing number. These event numbers will be generated by counters in the Front-End Systems from a common clock signal distributed by the Trigger and Timing system (see Chapter 5.2.1, "Trigger Timing and Control (TTC) System"). The event number increments continuously with each Level-1 accept and counts the number of triggers since the last resynch. It wraps around after about 1 million triggers. The bunch-crossing number is generated by a 40 MHz clock and gives the bunch-crossing of an accepted trigger. It is resetted every orbit. Since the DAQ system uses this pair of numbers (or an alias of them) to assemble the event, any value-error in these numbers leads to an assembly of data from different triggers[1].

A transient fault that causes the event number counter to over- or under-count (a value-error) does not stop the DAQ system. However, all assembled-events will be corrupted until the faulty counter is reset. This failure will be detected by the observation of data fragments with the same event number, but having a different bunch-crossing number. On the other hand, over- or under-counting of the bunch-crossing number does not lead to a faulty assembly of events. However, such events can not be distinguished from events assembled from data fragments with value-error in their event numbers.

In principle, over-counting or under-counting failures of the event or bunch-crossing numbers could be detected by the BU nodes, by checking the consistency of the numbers in an assembled event. In contrast to other transient failures which lead to event corruption at a small number of BU nodes, these failures are single-point failures which affect all BU nodes. Since their errors persist until a reset of the faulty counters, the late detection (in the order of seconds) of this class of failures results in the loss of a potentially large number of events. This may not be acceptable. The earliest nodes in the assembly chain which could detect these failures are the Readout Unit Inputs (see Section 9.2.2.2, "Readout Unit Input" for discussion on how this error can be detected.).

9.2.2 Fault Tolerance of the Event Building Nodes

The event building protocol described in Section 5.3, "Event Builder Protocols and Data Flow" together with the fault tolerance extensions summarized in the introduction of this section guarantees the end of the life cycle is reached for each event, even in the presence of transient faults. For all events in the system at a given time, if all nodes complete their activity and release all resources used in the event building process, then the DAQ system will be in a consistent state and ready to accept new triggers. In the process of recovering system consistency, intentionally or unintentionally, events being assembled could get corrupted. Since transient faults disappear from the system without requiring any intervention, an assembly of events entering the system at a later time will proceed according to standard system behavior.

1. In principle, the event number alone will be sufficient to assemble events when there are no failures. The event/bunch-crossing number pair will facilitate the detection of certain failures.

9.2.2.1 Front-end Readout Link

The “single input” Front-end Readout Link (FRL) boards with their associated Network-Interface Cards (NIC), behave as FIFO buffers and therefore do not deviate from their standard behavior with the occurrence of input failures. All input failures will be transmitted to the output without any permanent effect on the internal states of these nodes.

Due to the FIFO behavior of the readout, all input failures can be detected by comparing the event number of the event data with an internal sequence counter local to the FRL. Although this simple mechanism is sufficient to detect failures, it is not possible to localize the fault and invoke the proper recovery action. For example, the event number of the incoming event data will be different from the local FRL sequence number if either the event number or the local sequence number has a value error.

Since the routing of the event data depends on the event number, any value error of this field, either at the input of the FRL or generated due to an internal fault, can lead to miss-routing of the data fragment. Miss-routed data fragments will produce an omission failure for the intended destination node and spurious data for the node which will eventually receive it.

When a FRL merges two inputs, an omission failure at one of the inputs would lead to merging of data from different triggers. Although this error can be detected by comparing the event numbers from both inputs and a local sequence counter, the loss of synchronization persists until the resynch of the front-end nodes. Miss-routing of the data will be avoided by the use of majority logic, however in this case, the assembled fragment might contain data from two different triggers and therefore it will be marked as corrupted.

9.2.2.2 Readout Unit Input

The Readout Unit Input (RUI) module will assemble data fragments from multiple (typically 8) FRLs and present the super-fragment to a Readout Unit (RU), preserving the trigger time ordering. All omission failures will be detected with a watch-dog timer and the event in question will be flagged as incomplete and passed to the RU, and the resources allocated to it will be released.

The RUI implements a credit scheme based traffic shaping to avoid data loss due to memory shortages. Nevertheless, it may run out of memory if the RUI fails to transfer super-fragments to the RU memory. In this case, the RUI will delete the oldest super-fragment(s) in its memory to make room for fragments from new triggers. This behavior will result in an omission failure at the RU input.

It is possible to detect most spurious messages by comparing the event number and the expected sequence number. Undetected spurious messages will corrupt the fragment being built with the same event number. However, this corruption can be detected by the RU, Builder Unit and /or Filter Unit by checking the consistency of the event data. When valid data for an event from the same FRL which corrupted the event in question by generating a spurious message arrives, it will be detected as a spurious message and will be dropped with no side effects.

By comparing the bunch crossing numbers of fragments from different FRLs, the RUI is in principle the first node which could detect data miss-alignment errors. If the error is a transient error, simply marking the super-fragment as corrupted would be sufficient and no further action is necessary. However, if the error persist the system has to be resynched (see Section 5.2.2.2, "Synchronisation Recovery").

9.2.2.3 Event Manager

From the fault tolerance point of view, Event Manager (EVM) nodes maintain the state of Event-IDs they manage and allocate them to BUs for event assembly. Each Event-ID has three states: “Free”, “Open” and “Allocated”. Receiving a trigger message from the GTP in the EVM causes one of the “Free” Event-ID’s to change its state to “Open”. The EVM informs the RUs that a trigger arrived and the event data was assigned an Event-ID. Upon receiving an Allocate message from a Builder Unit (BU), one of the “Open” Event-ID’s will be assigned to the BU requesting an event for assembly and the state of the Event-ID is changed to “Allocated”. A Clear message is always associated with a given Event-ID and its arrival will trigger the release of an Event-ID. The Clear message changes the state of the Event-ID it is associated with to “Free”, thus making it available for reuse by future triggers.

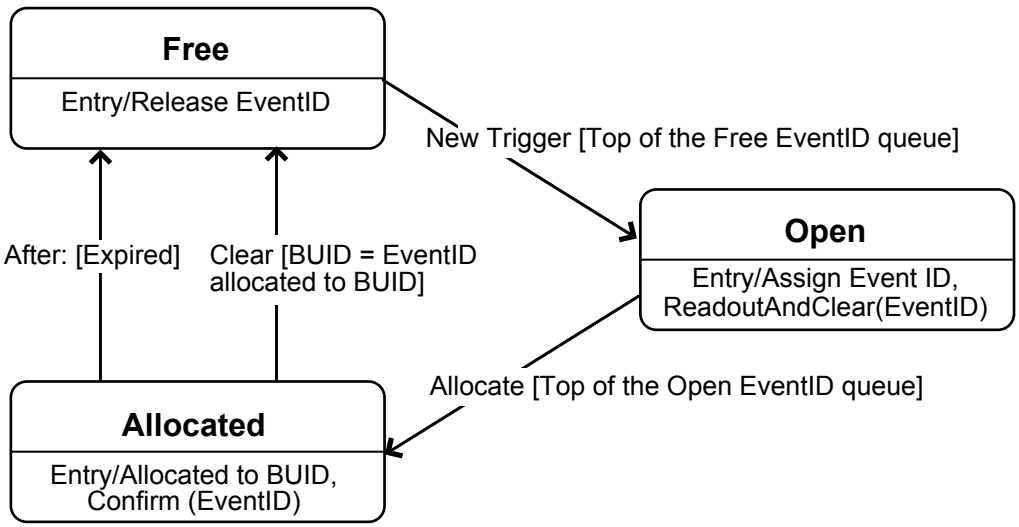


Figure 9-1 State chart of an Event-ID.

An omission failure of the trigger message has no effect on the internal state of the EVM. However, since no readout command is sent to the RUs, event data associated with this trigger stays in the RU memory. In this failure mode, since all RU’s miss a readout command, no data corruption occurs for the proceeding triggers. Again, the omission of the Allocate message has no effect on the EVM but generates an omission failure for the expected Confirm message.

Since Clear messages are always associated with a given Event-ID, their omission will cause an Event-ID to remain ”Allocated” until the EVM watchdog expires. If the total number of Event-IDs in the system is sufficiently large[1], the watch-dog timers for each Event-ID can be replaced by a periodic but less frequent garbage collection which checks the state of all Event-IDs.

In addition to the input failures mentioned above, an internal failure might trigger an unwanted state change of an Event-ID. Below (see Table 9-1) is the list of all possible faulty transitions and the resulting failure modes of the system.

1. The total number of Event-IDs in the system depends on the available memory in the RUs and BUs.

Table 9-1 List of all possible faulty transitions and the resulting failure modes of the system due to internal EVM failures.

Faulty transition	Failure modes
“Free” Event-ID is changed to “Open”	Previous event which was associated with the Event-ID will be reassembled. After successful completion of the assembly, Event-ID state will be restored to its correct state and the system returned to its standard behavior.
“Free” Event-ID is changed to “Allocated”	Event-ID and the associated resources stay blocked until the next garbage collection. No event is lost.
“Open” Event-ID is changed to “Allocated”	The event associated with this Event-ID never gets assembled. The next garbage collection will release the Event-ID for reuse.
“Open” Event-ID is changed to “Free”.	The event in the RUs will be over written by a new event and the original event will be lost.
“Allocated” Event-ID is changed to “Free”	The Clear message expected in association with this Event-ID will be identified as a spurious message and will be dropped. If this Event-ID is used by a new trigger and a Clear-and-Readout command is executed before the event previously assigned to this Event-ID gets assembled, the resulting over writing of the event fragments in the RU memories will corrupt the previous event.
“Allocated” Event-ID is changed to “Open”	Same event will be reassigned for assembly and selection. The Clear message from the first assembly will get identified as spurious and will be ignored. Completion of the second assembly will free the Event-ID. If the event is selected for archiving, the archive might have two copies of the same event if it does not catch this failure.

9.2.2.4 Readout Unit

A Readout Unit (RU) stores the event fragments received from its Readout Unit Input (RUI) node and provides random access to the event data during event assembly. A RU receives a “Clear-and-Readout” command from the Event Manager (EVM) and “Send” messages from the Builder Units which it replies to with “Cache” messages.

Due to the FIFO nature of the readout and the one-to-one relation between the incoming event fragments and the “Clear-and-Readout” messages, any loss of alignment between the command and the data, if not detected and corrected, results in assembling of event fragments from different events. The loss of alignment will be detected by comparing the event and bunch-crossing numbers in the command and data messages.

Upon receiving the “Clear-and-Readout” command, a RU releases all resources used by the previous event which was assigned the same Event-ID. The new event data are stored with this Event-ID only if the command and the data are aligned. Although this will prevent the assembling of fragments from different events, all events will have missing fragments from faulty RUs until the alignment is re-established.

Realignment of the command and the event fragment can be established by removing the data and/or commands from their respective input queues. This recovery will result in the loss of a few events, but will be more efficient than a full system resynch. Table 9-2 lists various failure which upset the command and data alignment along with the side effects of recovery.

Under normal operations, the number of event fragments stored in the memory of a single RU is equal to the number of Event-IDs. Each RU will have sufficient memory to hold all fragments. However, internal faults might lock some of this memory. A garbage collection will periodically reclaim the locked memory. In principle, the asynchronous Trigger-Throttling System (see Section 5.2, "Trigger-Throttling System (TTS)") will prevent a RU memory overflow. If the throttle system fails, memory is freed by removing the oldest event fragment from the RU memory.

BUs request a given event fragment using a “Send” message. The readout and transfer of the event fragment from the RU memory is not a destructive process and does not change the internal state of the RU. Therefore, an omission failure, a value error or a spurious message do not have any adverse effects on the state of the RU. However these failures will propagate as RU failures.

1. A “Send” omission will result in an omission of the corresponding “Cache” message.
2. A spurious “Send” message from a BU will be replied with a spurious “Cache” message.
3. A value error in the destination field of the “Send” message will result in sending the event fragment to a different BU.
4. A value error in the Event-ID field of the “Send” message will generate a spurious “Cache” message containing the wrong event fragment.
5. A value error in the resourceID field of the “Send” message will be copied to the “Cache” message which will be identified as a spurious “Cache” message in the BU.

In addition, due to internal errors, a RU can send the wrong event fragment. This failure can be detected by checking the consistency of the event and bunch-crossing numbers of the request and the reply at the BU.

Table 9-2 Various failures that upset the command and data alignment along with the side effects of recovery.

Failure	Side effect	Recovery
Omission of a Clear-and-Readout command	Input data queue has an additional event fragment. Readout command aligns with the next event fragment in the input data queue.	Remove the event fragment from the input data queue.
Omission of an event fragment	Input command queue has an additional command. Event fragment aligns with the next command in the input command queue.	Remove command from the input queue.
Spurious Clear-and-Readout command	Input command queue has additional command. Event fragment aligns with the next command in the input command queue.	Remove command from the input command queue.
Spurious event fragment from RUI	Input data queue has an extra event fragment. Command aligns with the next event fragment in the input data queue.	Remove event fragment from the input data queue.
Value error in the Event number field of the event fragment.	Next Command aligns with the next event fragment in the input data queue	Remove both event fragment and command from the input queues.
Value error in the bunch-crossing number field of the event fragment	Next Command aligns with the next event fragment in the input data queue.	Remove both event fragment and command from the input queues.
Value error in the Event number field of the Clear-and-Readout command.	Next Command aligns with the next event fragment in the input data queue.	Remove both event fragment and command from the input queues.
Value error in the bunch-crossing number field of the Clear-and-Readout command.	Next Command aligns with the next event fragment in the input data queue.	Remove both event fragment and command from the input queues.

9.2.2.5 Builder Unit

Following the Readout Unit event builder protocol described in Section 5.3, "Event Builder Protocols and Data Flow", a Builder Unit (BU) node sends “Allocate” and “Clear” messages to the EVM and “Send” messages to the RUs. It receives a “Confirm” message from the EVM and “Cache” messages from the RUs as reply to an earlier “Allocate” and “Send” messages.

Omission failures of “Confirm” and “Cache” messages will lock the resources allocated to the event in question. However, proper use of watch-dog timers will detect these resource dead locks and resolve them. Partially assembled events will be passed to Filter Units (FU) as incomplete events. Omission failure of a “Confirm” message also triggers an omission of the associated “Clear” message at the EVM which locks the Event-ID in question permanently if not detected and handled by the EVM.

A value error of the Event-ID field will not effect the event building functions. However, depending on the actual state of the Event-ID, a BU might assemble and later release an event assigned to another BU, or assemble an event incorrectly if there is an ongoing readout for this Event-ID. In the first case, one of the two “Clear” messages which is generated at a later time will appear as a spurious message to the EVM. The latter case will be identified by consistency checks of the assembled event at the FU or BU.

9.2.3 Fault Tolerance Analysis of Filter Unit - Builder Unit Communication

Filter Unit (FU) - Builder Unit (BU) communication is based on Allocate, Collect, Discard and Take messages as described in Section 8.3.2, "Data Flow". Table 9-3 lists errors that may occur in the BU-FU communication.

Table 9-3 Errors in FU - BU communication.

Message	Error	Effect on BU and solution
allocate	message lost	Undetected. FU requests again after timeout.
	duplicated (FU transaction id known in BU)	BU delivers data again
collect	message lost	Undetected. FU requests again after timeout
	duplicated	BU delivers data again.
	resource not granted to requesting FU before or resource is invalid	BU shall reply with an error to the requesting FU
	requested fragment set invalid	BU replies with error to requesting FU
discard	message lost	Stuck event remains allocated in BU. It will be cleared after a configurable timeout.
	resource not open/not allocated or invalid	Message ignored.
take	data delivery is lost	FU time-out and re-allocate.
	transaction/resource mismatch	FU ignores message and reiterates last undelivered data request.

9.3 Persistent Failures

The chosen exception handling mechanism described above is based on the assumption that the fault is transient and external to the node which detects the resulting failure. Therefore, simply bringing the system back to a consistent state will be sufficient to recover from these transient faults. However, if the failure is due to a permanent fault, the resulting failure will persist and the system will fall back into the same error state after each recovery. In other words, the recovery will not be successful. A permanent fault must clearly be localized and removed in order for the recovery to be successful.

Nodes that detect persistent failures do not have sufficient information for fault localization and fault treatment. These fault diagnostics functions will be executed by a central agent, the Problem Solver unit (see Section 12.4.6, "Problem Solver (PS)"), which has more global information about the state of the system.

All failures will be reported to the Information and Monitor Service (IMS) (see Section 12.4.4, "Information and Monitor Service (IMS)"). Persistent failures reported to the IMS will be passed to the Problem Solver for fault diagnosis. It will correlate various failure reports from different subsystems, execute specific test procedures if needed, and localize and perform the necessary repair action on the faulty unit. The Problem Solver can use the IMS to gather further information and use function managers to invoke actions.

9.4 Conclusions and Outlook

This chapter provided a summary of the present understanding of the failure modes, the fault tolerance of the DAQ system, and fault detection and recovery mechanisms. A crucial aspect of fault tolerance is the correctness of all code associated with detecting faults and recovering from them. Given that the occurrence of faults should be the exception, rather than the rule, fault-handling code is only rarely executed. It is thus crucial to have all systems including the fault-handling code extensively tested before commissioning. To establish the correctness and scaling of the fault handling algorithms, fault injection will be used at each level of development, from the prototype Event-Builder to the final system, and also in a simulation.

The simulation described in Appendix B.4, "Simulation of Myrinet Event Builder" is currently being extended to study the failure behavior of the DAQ system. For this purpose, a mechanism to inject errors into the system at any event building node is foreseen. The simulation was used to validate the fault recovery mechanisms. Presently, only omission failures for “send” and “receive” messages are implemented. Omission failures represent the largest fraction of failures since most of the described failures above will result in an omission failure. As the implementation of the event building nodes evolves, the simulation model will follow the scheme of error detection and will serve as a test bed to study undetected value errors and spurious messages.

Another important aspect of the simulation is the validation of the resource usage in the EVB components. By varying various parameters of the EVB process like fragment sizes, buffer sizes etc. the robustness of a given set of parameters against over-commitment of resources will be verified.

Throughout the design of the DAQ system described in previous chapters, emphasis was on the maximixng the number of events entering the DAQ system for event assembly. This objective requires that the data networks are utilized near 100% efficiency. All exception handling methods outlined in this chapter are designed with the same objective: maximizing the system availability even if it means sacrificing the dependability, i.e. event data could be corrupted in order not to decrease the system availability. For large class of failures, corrupted or lost event data can be recovered by retry, i.e. requesting the faulty information again. However, an increased reliability at the expense of decreased availability might have an overall adverse effect on the performance of the system. Thus the gain in reliability by the retry mechanism has to be carefully evaluated.

However, if the data networks does not have a spare capacity to absorb this additional load, number of events assembled without failure will not increase. If the failure probability increases with the network load, such as packet loss probability, the dependability will even decrease.

The study of fault tolerance can only act within the limits of the failure model of the system. The scope will improve towards integrity as the design advances from prototyping to the final design. As implementation details become known, new failure modes will be added to the model and through measurements on test-bench systems, the mean time between failures will be known more accurately. Extending the fault tolerance coverage is a continuous activity and will continue at each stage of construction, commissioning and even data taking.

9.5 References

9-1 H. Kopetz, “Real-Time Systems: Design Principles for Distributed Embedded Applications”, Kluwer Academic Publishers, 1997.

9-2 J. C. Laprie, “Dependability: Basic Concepts and Terminology”, IFIP WG 10.4 Dependable Computing and Fault Tolerance, Springer ,1992

Part 3
System Software, Control and Monitor

10 Online Software

This chapter gives an overview of the online software architecture. That is all the software tools and services needed for the transportation and processing of data as well as for the configuration, control and monitoring of all devices in the data acquisition system. A homogenous approach has been taken to the design and implementation of the online software. In accordance with this, all components participate in the data acquisition process through a uniform communication backbone, where a common set of rules have been defined for the format of exchanged data. The online software encompasses a distributed processing environment, data acquisition components, the run control and monitor system and the detector control system. These components are briefly outlined in the following sub-sections.

10.1 Overall System Architecture

The top level block diagram of the online software is shown in Figure 10-1. The architecture consists of four main parts:

- Cross-platform DAQ framework: XDAQ
- Data acquisition components
- Run Control and Monitor System (RCMS)
- Detector Control System (DCS)

The RCMS, DCS and data acquisition components interoperate through a distributed processing environment called XDAQ (cross-platform DAQ framework) to achieve the functionality required to operate the data acquisition system of the experiment.

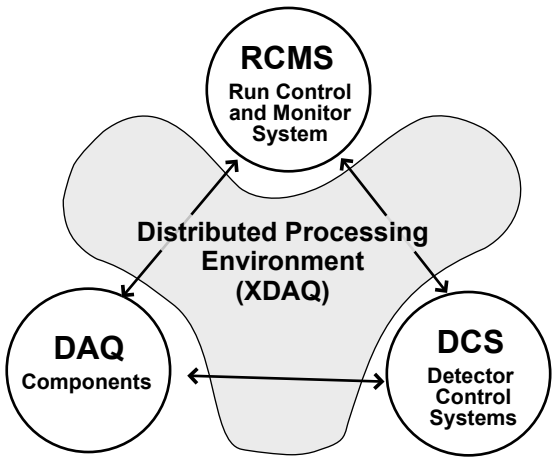


Figure 10-1 Overall online software architecture. Circles represent sub-systems that are connected via XDAQ.

10.2 Run Control and Monitor System

The Run Control and Monitor System (RCMS) drives the data acquisition system of the experiment for data-taking purposes. Figure 10-2 illustrates the context in which the RCMS operates and details its exter-

nal interfaces to other DAQ subsystems. Users operate the experiment through interfaces provided by the RCMS. To obtain information about detector status and to regulate the environmental sub-systems of the experiment, the RCMS communicates with the DCS system. The static and dynamic configurations of the system are maintained by the RCMS through the use of a database. The RCMS therefore has, at any moment in time, the overall state of the experiment. Interfaces to all data acquisition components and the trigger system are available to drive the data acquisition task. Finally, the RCMS also interfaces to the Computing Services described in Chapter 16.

The requirements, architecture, design and current prototypes of the RCMS are described in detail in Chapter 12, "Run Control and Monitor System".

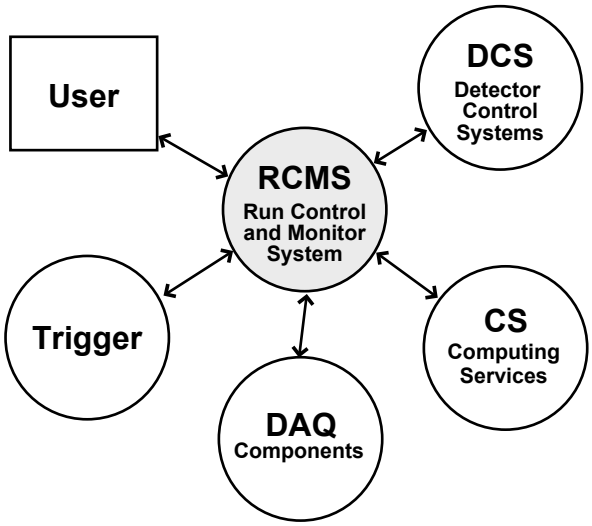


Figure 10-2 RCMS context diagram. Circles represent sub-systems internal to the DAQ; squares represent external systems.

10.3 Detector Control System

The Detector Control System (DCS) is responsible for maintaining and monitoring the operational state of the experiment. Figure 10-3 shows the system boundary of the DCS, which is connected to the operational infrastructure services of the experiment.

The DCS connects to the RCMS, the Computing Services, and the DAQ components. It treats these as sub-detector elements, controlling their power supplies and other environmental variables similarly to those of the dub-detectors.

During data-taking the DCS operates under the supervision of the RCMS, to which it is connected. A user interface is provided to access all DCS functionality for autonomous detector control and maintenance. All information needed to sustain the environmental conditions and their history, are stored persistently within a database. These data are provided on demand to the RCMS and to the data acquisition components. The DCS system is based on an industrial product.

Chapter 13 contains more information on the requirements and design of this system. The DCS integrates with all other components of the data acquisition through XDAQ.

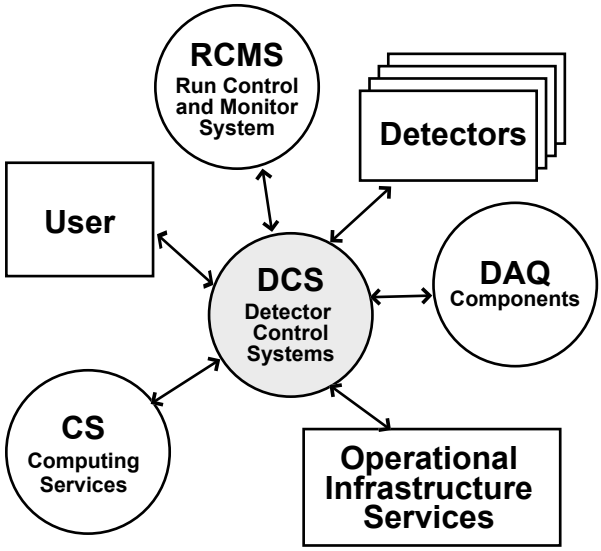


Figure 10-3 DCS context diagram. Circles represent sub-systems internal to the DAQ; squares represent external systems.

10.4 Data Acquisition Components

The collection of DAQ components includes applications such as the distributed Event Builder (EVB), subdetector electronics configuration and monitoring components (FEC and FED), and software-based trigger-throttling mechanisms (aTTS). These applications require interfaces as shown in Figure 10-4, to the Front-End Drivers, the trigger system, the high-level trigger services as well as the DCS. The relevant

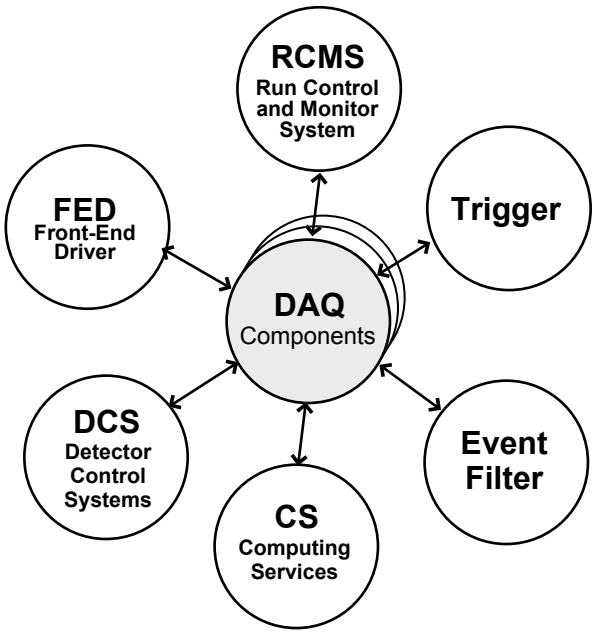


Figure 10-4 DAQ components interface. Circles represent sub-systems internal to the DAQ.

RCMS services connect to the data acquisition components to steer the data acquisition. Descriptions of all data acquisition components involved in the event building process can be found in Part 2.

10.5 Cross-Platform DAQ Framework: XDAQ

An overview of the online software infrastructure based on XDAQ is shown in Figure 10-5. The framework provides basic system services such as memory management, communication services and the executives in which applications are executed. Applications make use of these services in order to perform their functions. Additional utility components provide support for hardware access, database access and other general-purpose interfaces such as the connection to the Detector Control System. Finally, high-level tools support the configuration, control and monitoring of the system. These tools also include the user interface that serves as the gateway to the data acquisition system. It is described in detail in Chapter 12, "Run Control and Monitor System".

An important decision in the design of the CMS DAQ system has been the adoption of a homogeneous software infrastructure for all components in the DAQ. The decision is motivated by the expected ease of integration and configuration of multiple and diverse subsystems. The use of common and standardized software technologies reduces the effort associated with the maintenance and evolution of the system over the long lifetime of the experiment.

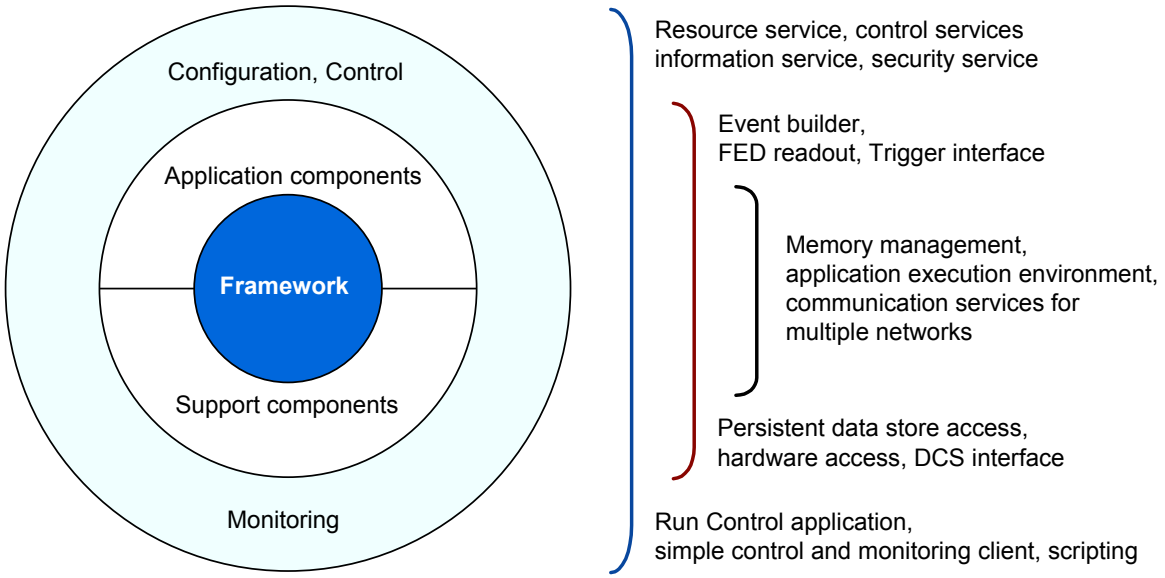


Figure 10-5 Overview of the online software infrastructure.

The above can be realized with a domain-specific middleware [10-1] which is designed for data acquisition and which offers its services through unambiguously defined interfaces. The role of this middleware is to ease the tasks of designing, programming and managing applications by providing a simple, consistent and integrated distributed programming environment. Such a platform abstracts from the complexity and heterogeneity of the underlying distributed environment. The requirements, design and technology foundations of this approach are described in Chapter 11, "Cross-Platform DAQ Framework (XDAQ)".

10.6 References

10-1 P. Bernstein, “Middleware: A Model for Distributed Systems Services”, Communications of the ACM, 39 (196) 86.

11 Cross-Platform DAQ Framework (XDAQ)

XDAQ is a framework designed specifically for the development of distributed data acquisition systems. It provides platform independent services, tools for local and remote inter-process communication, configuration and control, as well as technology independent data storage. To achieve these goals, the framework builds upon industrial standards, open protocols and libraries. This chapter starts by outlining the overall requirements of XDAQ, and gives a description of its design followed by a summary of the development status.

11.1 Requirements

Three types of requirements have been identified: functional, non-functional and testing. The first category includes all requirements related to the functionality of the system, whereas the second category includes all other requirements, with the exception of those related to the testing of the functionality and performance of the DAQ. The requirements are outlined in the following sections.

11.1.1 Functional Requirements

11.1.1.1 Communication and Interoperability

The framework must provide a set of facilities for the transmission and reception of data within and across subsystem boundaries. This functionality must be available for the communication both of applications on the same processing unit and for tasks that are distributed over physically distinct interconnected computers. Services must be available to retrieve all information that is needed to communicate to other application components.

The software infrastructure must also provide mechanisms to ensure true interoperability among application components within the system without regard to the protocol used.

The framework shall foresee the possibility of interaction with new external systems through the provision of facilities that allow the addition of communication protocols. These facilities shall be designed such, that no modifications in the applications that use the newly added communication method will be required.

11.1.1.2 Device Access

Various system and application software components need a set of functions to access custom devices directly for configuration and readout purposes. Access shall be provided in the same way for hardware devices that are directly connected to a computer as well as for devices that are accessed through bus adapters. This software layer must also include provisions for extensions to new bus adapter products. These requirements imply additional constraints on the portability, robustness and efficiency of the software, as discussed in Section 11.1.2, "Non-Functional Requirements".

11.1.1.3 Configuration, Control and Monitoring of Applications

The environment should provide the means to make the application parameters of any built-in or user-defined data types accessible to other applications within or outside the system. It shall therefore be possible to inspect and modify all such parameters.

To allow the coordination of application components, it must be possible to define their overall behaviour, that is their states and modes. A mechanism should be in place to allow the specification of the tasks that an application performs at a given time and which operations it accepts. In addition, services should be available to query and modify the state of applications.

A service must be present to record any structured information such as logging messages, error reports, as well as complex data types. Examples include statistics and charts that are generated during run-time. Such information items, termed “documents”, should be disseminated to subscribing client processes. The introduction of documents necessitates the presence of mechanisms for their persistent and transient storage. Documents can have a validity period after which they are discarded from the system and will no longer be accessible to clients. It shall be possible to associate a priority with a document, which determines the way in which it is processed and delivered to recipients.

The infrastructure must include facilities that allow the creation, maintenance, and persistent storage of information about all hardware and software components that can be used. The data must cover all configuration that is necessary for the applications to perform their specified functions.

Applications will be able to share physical resources such as computers or networks. The framework must support these operations by providing mechanisms to cope with allocation, sharing and concurrency conditions.

11.1.1.4 Re-usable Application Modules

The software infrastructure will include pre-built application components to perform the following tasks:

- Collection of data from a data link (see Section 7.3, "The Common Interface between the FEDs and the DAQ System")
- Event building according to the protocol described in Chapter 5, "Event Flow Control"
- Benchmarking of applications
- Diagnosis and validation of hardware and software components and their required functionalities
- Configuration, control and monitoring.

11.1.1.5 User Accessibility

In order to let a user interact with the system for configuration, control and monitoring purposes, a graphical user interface that is decoupled from the actual service implementations shall be provided. To perform repetitively occurring tasks it shall be possible to define re-usable procedures that automate all operations that the configuration, control and monitoring services offer.

11.1.2 Non-Functional Requirements

This section describes constraints that are placed on the system in order to achieve its goals within the environment in which it is embedded. The term “environment” includes the existing and foreseen hardware infrastructure, the data rate of the detector output, the lifetime of the experiment and various other issues that are not functions of the DAQ system.

11.1.2.1 Maintainability and Portability

Due to the variety of platforms that will be prevalent until the system is commissioned for operation, the infrastructure must provide all the means to allow portability across operating system and hardware platforms. This includes support for accessing data across multiple bus systems and the possibility to add upcoming custom electronics to the set of supported systems without the addition or removal of explicit instructions in user applications.

Operating system independence will be provided, but can only be maintained if applications do not directly use native system functions. Most important, the memory management tools of the underlying system should not be applied directly, since their uncontrolled use may affect the robustness of the system. In addition, the application code shall be invariant with respect to physical location and network. The rationale behind the latter requirement is to improve system testability and maintainability. Users can create applications in an environment with limited resources and move the validated application to the deployment system without imposing changes in the software. Common functions should be independently packaged in re-usable building blocks. This leverages development, integration and testing efforts.

11.1.2.2 Scalability

The application domain is characterized by data transmission requirements. In order to meet the overall system requirements (see Chapter 2, "Requirements") and not to introduce unpredictable behavior, the overhead introduced by the software must be constant for each transmission operation and must not affect the data throughput in a significant way. Factors that might affect latency and throughput are memory management, data copy operations and processing time that is proportional to the quantity of transmitted data. The system shall have the ability to scale, that is to operate within the specified requirements and to take advantage of additional resource availability, as its size increases.

11.1.2.3 Flexibility

With respect to the various networks and protocols that are involved in the data collection tasks, it is necessary to allow the application developer to use multiple communication channels over different networks and protocols concurrently.

11.1.2.4 Identification

All system components will be unambiguously identified within the system for communication, control and tracking purposes.

11.1.3 Testing Requirements

This section focuses on the requirements for testing the implementation of applications at the component and system levels.

All application and framework components will be tested independently of each other. Integration tests of collaborating components must be performed and results validated against the functional requirements. Test data will be used as input to exercise components, subsystems and the system under normal and abnormal operational conditions. These tests should be automated so that regression test procedures can be performed on every newly released software.

A set of separate application modules to exercise performance and scalability measurements must be included in the software environment. Performance measurements will then be carried out to verify that the system satisfies all requirements and to investigate its behavior under high load.

The collection of all test software and its documentation includes verification checklists, test data as well as supporting documentation like test plans, specifications, procedures, test cases, test data descriptions and reports. Testing is planned and carried out within the organizational framework outlined in Section 11.3.

11.2 Design

This section describes the design of the software infrastructure, named XDAQ, in direct response to the requirements.

11.2.1 Executive Framework

The distributed programming environment follows a layered middleware approach [11-1], designed according to the object-oriented model and implemented using the C++ programming language [11-2]. The distributed processing infrastructure is made scalable by the ability to partition applications into smaller functional units that can be distributed over multiple processing units. In this scheme each computing node runs a copy of an executive that can be extended at run-time with binary plugin components. The program exposes two types of interfaces: “core” and “application”. The core interfaces lie between the middleware and core plugin components, providing access to basic system functionalities and communication hardware. Core plugins manage basic system functions on behalf of the user applications, including network access, memory management and device access. The application interfaces provide access to the various functions offered by the core plugins and are placed between the middleware and the user application components as shown in Figure 11-1.

Middleware services include information dispatching to applications, data transmission, exception handling facilities, access to configuration parameters, and location lookup (address resolution) of applications and services. Other system services include locking, synchronization, task execution and memory management. Applications communicate with each other through the services provided by the executive according to a peer-to-peer message-passing model. This allows each application to act both as a client and a server. The general programming model follows the event driven processing scheme [11-3] where an event is an occurrence within the system. It can be an incoming message, an interrupt, the completion of an instruction, like a direct memory access transfer, or an exception. Messages are sent asynchronously and trigger the activation of user-supplied callback procedures when they arrive at the receiver side. Eve-

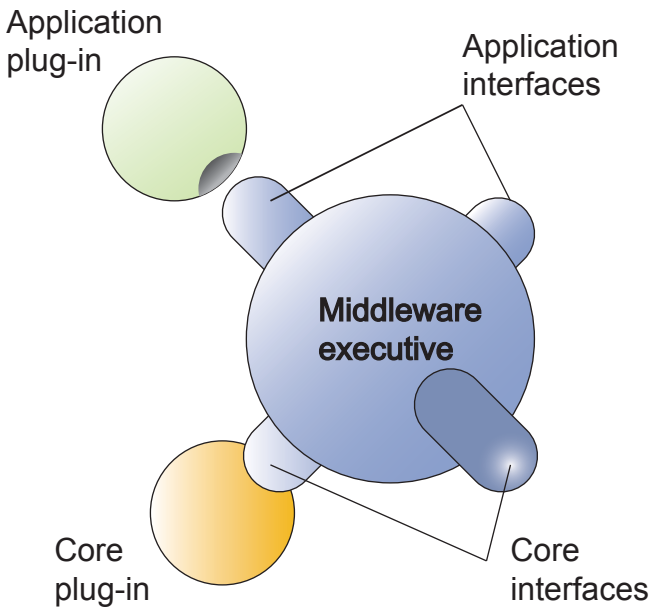


Figure 11-1 Middleware interfaces.

ry event therefore corresponds to a message that follows a standardized format. Initially, two formats are supported. One is based on the I_2O specification [11-4] and the other is XML [11-5].

The event-based processing model has been chosen because it is known to scale [11-6]. There is no need for a central place in which the incoming data have to be interpreted completely. It is the responsibility of each software component that listens to a given type of event (e.g. data received, timeout) to decide what it should do with the information received. Extensibility is thus achieved through the decoupling of the reception of a message and its processing. The procedure for a given message can be provided dynamically during run-time by downloading a software module that contains all code to react to an incoming message of a given type. Furthermore it is possible to add new functionality by defining new messages. The system provides a default procedure if for a given event no executable function has been supplied. This model results in a homogeneous structure of software components with an intrinsic fault-tolerant behavior.

11.2.2 Application Interfaces and State Machines

One responsibility of the execution model is to keep track of the local states of all running applications. The overall behavior of an application in the scope of the executive framework can be modelled as a state machine that is defined for each individual application. In order to facilitate the development of such interfaces, a common set of states and transitions is included with all applications and executives. State changes can be initiated by control commands that are sent to the applications and executives. The framework takes care of checking the consistency of the control commands. The success or failure of a state transition is reported to the initiator of the request. The current state of an application is a parameter that can be queried at any time from an external system. Dependencies between applications, which rely on state information are not foreseen and have to be managed by an external system (see Chapter 12, "Run Control and Monitor System").

The framework provides a mechanism, referred to as “reflection”, for the introspection of application interfaces in terms of exported variables and methods. Using this mechanism, an application can make var-

ious data elements visible to external systems. Both simple built-in data types as well as user-defined composite data structures are supported and the set of data types that are exported can be extended by the user. All functions defining the control interface of an application can be retrieved at run-time. This enables integration with external systems without the need to obtain a written specification of the application interface.

11.2.3 Memory Management

The executive program provides applications with efficient memory management facilities. These are based on a scheme called “buffer-loaning” which avoids fragmentation of memory over long run periods and presents a safe operation model that prevents extensive growth of memory consumption. With the buffer-loaning scheme, applications or core-plugins ask the executive for fixed sized chunks of memory from one of various buffer pools. The principle is displayed graphically in Figure 11-2.

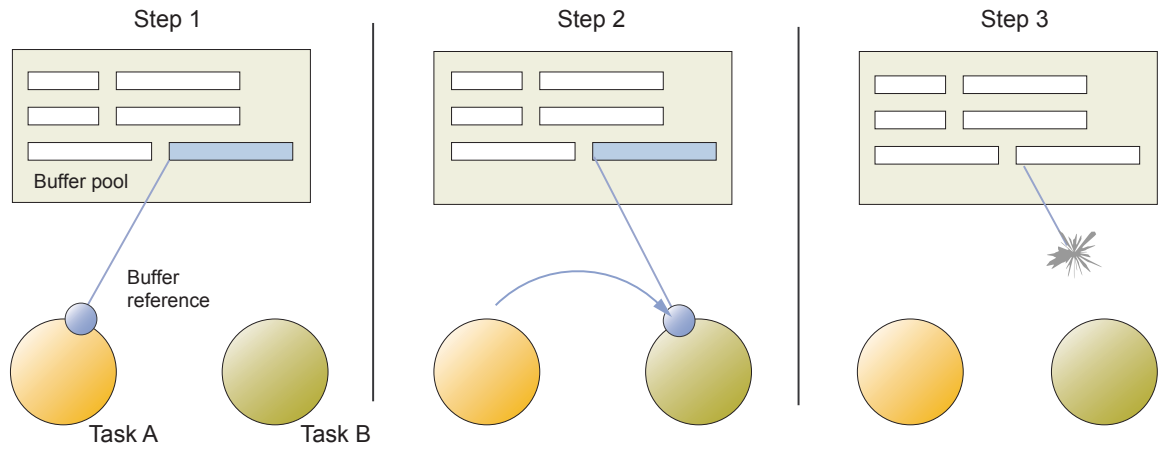


Figure 11-2 Illustration of the buffer-loaning mechanism. A task loans a reference to an unused buffer that matches the closest requested data size from a buffer pool (step 1). The buffer can be passed to another task by forwarding the buffer reference (step 2) without copying the data. The buffer is released to the pool by destroying the buffer reference (step 3). It can now be re-allocated to another task.

The executive manages various types of pools, including ordinary user-space memory, a reserved amount of physical memory (e.g. the Linux “bigphys” kernel extension) and memory on network cards. The pools can be configured such that exceptions are raised if too little memory is available. All memory buffers are allocated from the available pool and are accessible through a reference that can be further lent to other software components. For example, a user application prepares a message and passes it on to a transport component that handles the network transmission. Eventually, the buffer must be returned to its pool. Built-in reference counting ensures that a buffer is not returned into its originating pool before the very last user has released it.

Various buffers can be chained together to allow arbitrarily sized data. The mechanism not only enables efficient zero-copy implementations, but also provides the foundation for transparent operation across network boundaries. Various high-speed interconnects and custom built electronics rely on non-standard memory models, that would otherwise require the instrumentation of user programs with special instructions. Buffer pools can be added to the executive to offer specific allocation through an interface that is common to all memory types.

11.2.4 Data Transmission

Data transmission in the described distributed programming environment is carried out by special application components, named peer transports (see Figure 11-3). Peer transports register themselves with the executive as being capable of resolving addresses as well as transmitting and receiving data.

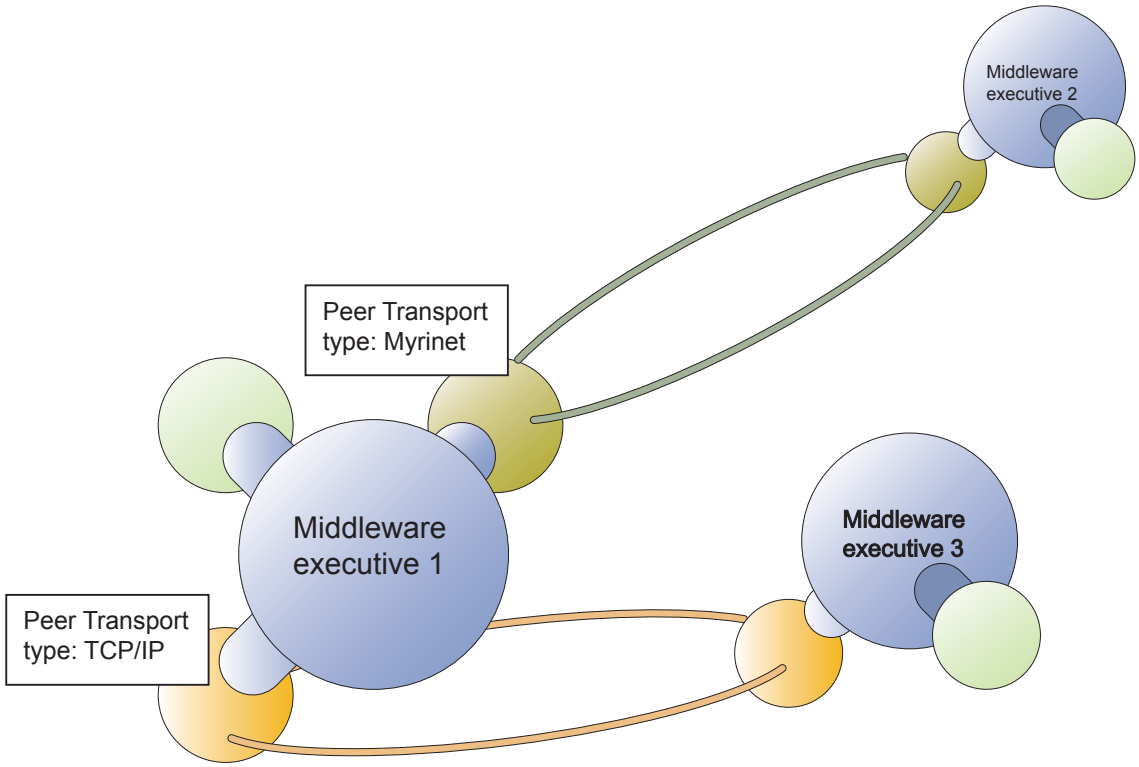


Figure 11-3 Communication over multiple networks through peer-transports.

Communication between ordinary applications is accomplished by means of an executive function. This function, when invoked, re-directs the outgoing message to the proper peer-transport that in turn delivers the data over the associated medium. In this way the framework is independent of any transport protocol or network and can be extended at any time to accommodate newly appearing communication technologies.

The baseline framework includes a number of peer-transports which are listed in Table 11-1. Applications can communicate according to two different models, unicast and multicast. Unicast is designed for transmitting data to a single destination, whereas multicast provides efficient handling of group communication within a network. The model extends a single line of communication over a network allowing messages to be sent simultaneously to a group of applications, i.e. all instances of a given class. If a peer-transport does not have the capability to perform true multicast (e.g. TCP/IP), it can tell the executive that it will emulate the required behaviour using normal point-to-point message exchange. This is an important feature to allow the verification of software in an environment with limited functionalities.

Table 11-1 Baseline peer-transports.

Transport technology	Description
TCP/IP	reliable messages over various physical networks
UDP (unicast and native multicast)	unreliable datagrams over various physical networks
SOAP (HTTP messaging with XML data)	reliable delivery of XML documents over various physical networks
local FIFO	communication among applications within the same executive
GM	reliable messaging over Myrinet
raw Ethernet (unicast and true multicast)	unreliable messages over Ethernet (VxWorks operating system only)

11.2.5 Protocols and Data Formats

The framework supports two data formats, one based on the I_2O specification and the other on XML. I_2O messages are datagrams with a maximum size of 256 kB. For sizes larger than this maximum, the data have to be split and sent in a sequence of multiple frames. The framework provides mechanisms to perform this task. I_2O messages are primarily intended for the efficient exchange of binary information, e.g. data acquisition flow. No interpretation of such messages takes place in the executive. The messages are used directly by the applications to perform the required computation tasks. However, the content is platform-dependent and the application designer must obey all alignment and byte-ordering rules. Moreover, the structures for I_2O messages are provided through C language definitions and a recompilation of the corresponding applications is required if the data content type changes. The use of the I_2O message frames can be combined with the buffer-loaning memory allocation scheme to create efficient applications. Thus, data that have been received do not need to be copied before they are passed on, since it is sufficient to work with references to the corresponding frames.

Despite its efficiency the I_2O scheme is not universal and lacks flexibility. A second type of communication has been chosen for tasks that require higher flexibility such as configuration, control and monitoring. This message-passing protocol, called Simple Object Access Protocol (SOAP) [11-7] relies on the standard Web protocol (HTTP) [11-8] and encapsulates data using the eXtensible Markup Language (XML) [11-5]. The adoption of SOAP naturally leads to the use of Web Services that standardize the way in which applications export their interfaces to possible clients[11-9]. The executive middleware exposes an interface to the application programmer that is based on the JAXM package [11-10] from Sun Microsystems and implements the W3C SOAP 1.1 specification for creating and parsing SOAP messages. Clients can use any SOAP-compliant communication package to interact with the executives and their application module plug-ins. This approach has several advantages over the transmission of statically declared, binary messages. First, the availability of a standardized interface to inspect and manipulate XML data, called the Document Object Model (DOM) [11-11] helps to quickly integrate the applications with a variety of data sources. XML data can be read from disk or remote Web sites; they can be processed by a number of tools for the majority of operating systems, they are machine and human readable and can be transmitted over any kind of transport protocol. XML data are understood across platform boundaries, because data are plain text and through the association of a grammar, it is possible to describe the contents of a message. Second, XML favors system robustness through the use of grammars and named elements that allow detection of malformed messages. Third, XML allows scalable configuration, monitoring and control, by avoiding large quantities of data to be sent across the network at once. Messages may contain links to data that can be resolved when needed. Applications may export information by transmitting an XML structure with the link to the location at which they have placed the generated data. It is important to note that there is no limit to the data content type. Any text, binary or complex structure can be de-

scribed in XML and any kind of protocol can be chosen to transmit these data. The framework supports an initial small set of data structures that can be extended gradually with additional capabilities (see Table 11-2).

Table 11-2 Some data types that XDAQ can export through XML messages.

Data type	Example
simple types (built-in C++ native types)	unsigned long, float, string
vectors of simple or composite types	vector<int>, vector< vector <string> >
bags of simple types (collections of types)	(unsigned long, float, vector<string>)
JAS [11-12] plot types	a histogram, including error bars

11.2.6 Application Components

In addition to the services that support distributed computing, the online software framework includes components that facilitate the building of data acquisition systems including general tasks like hardware access, database access and local data monitoring. These components are intended to be used in four domains:

- the global DAQ system of the experiment
- local DAQ systems
- the development and production environment of the FED
- testbeam DAQ systems

11.2.6.1 Data Acquisition Components

The online software infrastructure provides re-usable application components to perform event building according to the protocols described in Chapter 5, "Event Flow Control". This includes generic software for the Readout Unit (RU), Builder Unit (BU) and Event Manager (EVM) subsystems. The event builder components form a distributed application that can be configured to run on a set of networked computers. The application exposes three distinct interfaces: input from the detector readout, output to the High-Level Trigger system and input from the trigger system. In addition to the event builder, the software includes the application components necessary for the readout through the front-end links as well as the handling of trigger information.

The DAQ software components serve as building blocks for the local DAQ systems, where they can be used for commissioning part of the DAQ itself, and also for calibration and detector monitoring purposes. However, they may need adaptation to meet additional requirements that are specific to the operational environment. A similar procedure applies to testbeam and FED development applications.

11.2.6.2 Common Application Components

Components are included, to be used from within applications, to ease direct hardware access, persistent data store operations, communication with DCS and local monitoring.

The hardware access library is used to directly manipulate hardware devices for configuration purposes. It provides access to registers and memory areas for various bus systems through named objects that can be defined dynamically at run-time.

The database access library exhibits an interface to read information from persistent data stores and to present it to the user through the standardized Document Object Model (DOM) of the World Wide Web Consortium (W3C). The DOM renders information as trees that can be traversed by the application code. Each node can contain a named data element or may be a link to other information that is accessible through one of the provided protocols. The library provides the required authentication and access regulation procedures for each type of persistent data store that is supported. No particular requirements are imposed on the database schema for using this package. The baseline framework provides support for Oracle [11-13], MySQL [11-14] and the file system.

The DCS access component enables applications in the online environment to communicate with the experiment's Detector Control System. Any control system that has been built with the PVSS-II SCADA [11-15] product can be seen by an application through this component as just another application component. Conversely, an application in the online domain can be seen by the PVSS-II system as soon as it registers with the system. Thus, bi-directional communication between the DAQ and DCS domains is achieved. The interface between the two systems is described in detail in [11-16]. An online application component can read and write values and subscribe for change notifications. The PVSS-II system can read and modify values that are made visible by an online application. The PVSS-II system can also request to be notified when value changes occur, but it is left to the designer of the online application to explicitly respond to such a request. The PVSS-II system may also access the interface of a registered application in order to invoke actions on them. Provision of functionality to perform the described tasks with the PVSS-II system belongs to the DCS domain.

The data monitoring library is used to perform basic statistical analysis for monitoring application and system operation on a local processing node. It allows the structuring of the results for storage with database systems or for external visualization toolkits.

11.2.7 Configuration, Control and Monitoring Components

Configuration, control and monitoring correspond to tasks that affect the distributed programming environment and the applications that run in it. All three tasks involve components both in the executive programs and in the applications. The configuration of the executives comprises all information that is needed in order to

- identify which applications must be run in a certain executive
- communicate to other executives and applications, including external subsystems
- find the location of the binary objects needed for all described tasks.

The configuration of the applications is limited to providing default values for their exported parameters.

To manage configuration data, a service that keeps track of all hardware and software resources is provided. This service acts as a database and does not yet allow the control and configuring of the running system. This will be carried out by control services. “Control” in this context is defined as all the actions necessary for the deterministic execution of state transitions that have been defined for the executive and all configured applications. These services are responsible for initiating state transitions in executives and applications, thus including the issuing of configuration commands (see Section 12.3.1, "State Definitions and System Synchronization").

“Monitoring” corresponds to the collection of data that have been made visible by the executives and applications through exported parameters or supplied functions that can be invoked through their corresponding messages. Two modes are supported:

- pull mode: the collection of data is performed by repetitive inquiry of the involved components
- push mode: all components actively send data to the monitoring service

The software infrastructure provides the basic primitives to implement application-specific monitoring through the Information Service as described in Section 12.4.4, "Information and Monitor Service (IMS)".

11.3 Software Process Environment

The process via which the evolution of the system occurs in an orderly fashion is termed the software process. It is an iterative process of synthesis, development and operation of the software that satisfies in a near-optimal manner the full range of requirements for the system and thus helps to increase the likelyhood of being successful.

The development of the CMS online software has adhered to established standards of software processes. A short summary of the main elements is provided here.

11.3.1 Documentation Guidelines and Procedures

Guidelines based upon existing standards are being defined to provide support for the documentation of the system under development, including all artifacts developed during the life cycle of the project. The format, content and categories of documents for the software process and development shall be defined. Guidelines will help in preparing technical documentation, provide a reference for uniformity of style, format and content, with the goal of enhancing consistency of documentation and visibility into the completeness of the project.

11.3.2 Configuration Management Tools

Configuration management is supported by various tools that are freely available. CVS [11-17] is a widely used tool for version management and GNU Make [11-18] is a tool for system building. Both of these are normally supplied with the UNIX system. Configuration management relies on a database that records information about system changes and change requests that are outstanding. It also requires some formal means of requesting changes. A combination of CVS and the SourceForge.net [11-19] facilities fit these needs. Distribution of the software to end users is accomplished through a packaging and installation tool that must be capable of covering all platforms supported by the software. Currently InstallAnywhere is used for this purpose [11-20]. In addition, CVS server access is granted to users that require immediate updates. These include co-developers from remote institutes and users with the need for tailored components, e.g. testbeam applications.

11.3.3 Software Development Environment

The online-software is based on various existing industry standards to reduce development and maintenance efforts as well as to achieve interoperability with external systems. The selected technologies increase the probability of achieving these goals, but cannot guarantee success by themselves. A firm understanding of the technologies is required together with expertise in system integration.

The implementation language of the executive and its application components is C++. The reasons underlying this decision are fourfold. First, the distributed programming environment has been designed according to the object-oriented model and C++ maps well to this approach. Second, C++ is the implementation language of the offline reconstruction framework. Hence, integration with high-level trigger code that is based on the offline software interfaces is facilitated. Third, the online software environment relies on efficient program execution that can at present only be achieved with compiled binary executables. Fourth, C++ is fully integrated with C, the systems programming language of the majority of operating systems. The currently supported platforms are listed in Table 11-3. Configuration, control and monitoring services are currently implemented in Java and are therefore available on any Java enabled platform..

Table 11-3 Platforms and languages currently supported by the online software infrastructure.

Type of software	Platform	Languages/Compilers
Executive and application components	Sun/Solaris on Sparc processors, Linux on Intel processors, Mac OS X on PowerPC processors, VxWorks on Intel and PowerPC processors	C++/GNU g++, C/GNU gcc
Configuration, control and monitoring services	Any Java/Tcl enabled platform	Java/Sun JDK, Tcl

The software is available for, but not limited to the Linux (Red Hat distribution), Mac OS X (Apple Inc.), Solaris (Sun Microsystems) and VxWorks (Wind River) operating systems. VxWorks is a real-time system that finds its deployment wherever requirements occur that cannot be met by a time-sharing operating system, such as the UNIX family. This includes rapid prototyping of embedded environments and its use in highly constrained environments. VxWorks has been chosen because of its capabilities to be tailored to a variety of hardware platforms, its proven efficiency and wide acceptance in industry. Solaris and Linux are two major players in the UNIX marketplace and excel through the vast amounts of documentation. Last but not least, most third party products to which the online software infrastructure has to interface are supported on these target platforms. All what has been said for Solaris and Linux also applies to the MacOS X platform. In addition, a variety of software development and system administration tools are available that exploit the full range of functions that this system provides.

For processing SOAP messages, use is made of the freely available Xerces parser from the Apache project [11-21] library that implements the Document Object Model (DOM) interface for manipulating XML data in C++ and Java. DOM is a W3C (Word Wide Web Consortium) standard that defines language bindings for various programming languages including C++ and Java.

Relational databases have proven robustness and scalability, and have therefore been selected as the primary means for persistently storing configuration data in the online software environment. The market also suggests that these products will be around for the entire lifetime of the experiment. The format of the data entered into and retrieved from different database products has been normalized to XML and DOM. This design choice allows the development of applications independently of the database

back-end. The online software infrastructure presently supports the following back-end technologies: Oracle, mySQL, file system and native XML databases[11-22][11-23].

The software infrastructure includes a graphical user interface, entirely written in Java, which serves simple configuration, control and monitoring purposes. It uses the JAXM [11-10] package for performing SOAP communication and interfaces to the Java Analysis Studio (JAS) [11-12] to demonstrate the capabilities of interfacing to third party products through the XML language. Scripting is supported in the simple control program through a built-in Tcl language interpreter [11-24]. No limitation exists on the use of other scripting languages for interfacing to the distributed programming environment as long as SOAP communication is available.

11.4 System Management

These tools are intended to manage the set of all computers that make up the DAQ system. The management functions include installation of the operating system, installation and upgrade of the application software as well as the boot strap process. A secondary function is the continuous monitoring of the computing infrastructure. Detailed information can be found in Section 16.6.1, "Cluster Management".

11.5 Prototype Usage and Experience

Various prototypes of the described software infrastructure have been used in testbeam environments for the Muon and Tracker subsystems. The gathered experience served as an input for additional requirements and technology decisions. The software is also actively used in FED development projects and for the preparation of front-end configuration systems.

In addition, the software infrastructure has been installed at institutes that participate in the development of the DAQ system. At CERN, the software environment serves as an infrastructure for DAQ system evaluation (see Section 7.7, "Readout Column Prototypes and Test-benches") and for integration tasks with the high-level trigger processing framework (see Section 8.6, "Prototypes and Results"). For a detailed analysis and basic efficiency measurements of the communication subsystem implemented in XDAQ, see [11-25]. The existing downscaled DAQ column prototype proves that the design can be successfully applied in a “real-world” environment and that there are no observable deficiencies introduced by the chosen technologies. In the context of design, the prototype also shows that the correct balance has been made between flexibility and efficiency.

The deployment of the software in various real scale application scenarios demonstrated the viability of the approach taken. The following sub-sections outline two of these applications, the tracker local DAQ system for front-end control and calibration, and the muon chamber validation system.

11.5.1 Tracker Testbeam

The silicon microstrip detectors are readout via an analog optical link connected to the analog to digital converters of the FED. The control commands for the front-end chips, together with the clock and level 1 trigger, are propagated to the detectors through a proprietary token-ring network. This task is carried out by a Front-End Controller (FEC) interacting with several communication and control units (CCUs). Commissioning of detectors is a dedicated data acquisition task including the communication channel de-

scribed above. For example, timing calibration is required, because the trigger is propagated sequentially along the ring to the detectors and the digitization time depends on the fibre lengths to the FEDs.

Additional calibration tasks include pulse shape adjustment, optical link gain setting and front-end gain calibration. Test setups include slow control facilities, such as thermistors and I^2C driven humidity probes, and HV/LV control, which are driven by XDAQ applications. For this purpose, a local data acquisition system that supports the calibration loop process in addition to configuration and control operations must be available.

A high-level diagram of the system in operation is shown in Figure 11-4. XDAQ together with the generic event builder components has been successfully used to implement the system described above. Additional specialized software components were developed to interface to detector specific electronics and the persistent data storage technology. Online visualization facilities were implemented with Java Analysis Studio and interfaced to the system through the XDAQ SOAP messaging system. The implementation of the system took 4 FTE months.

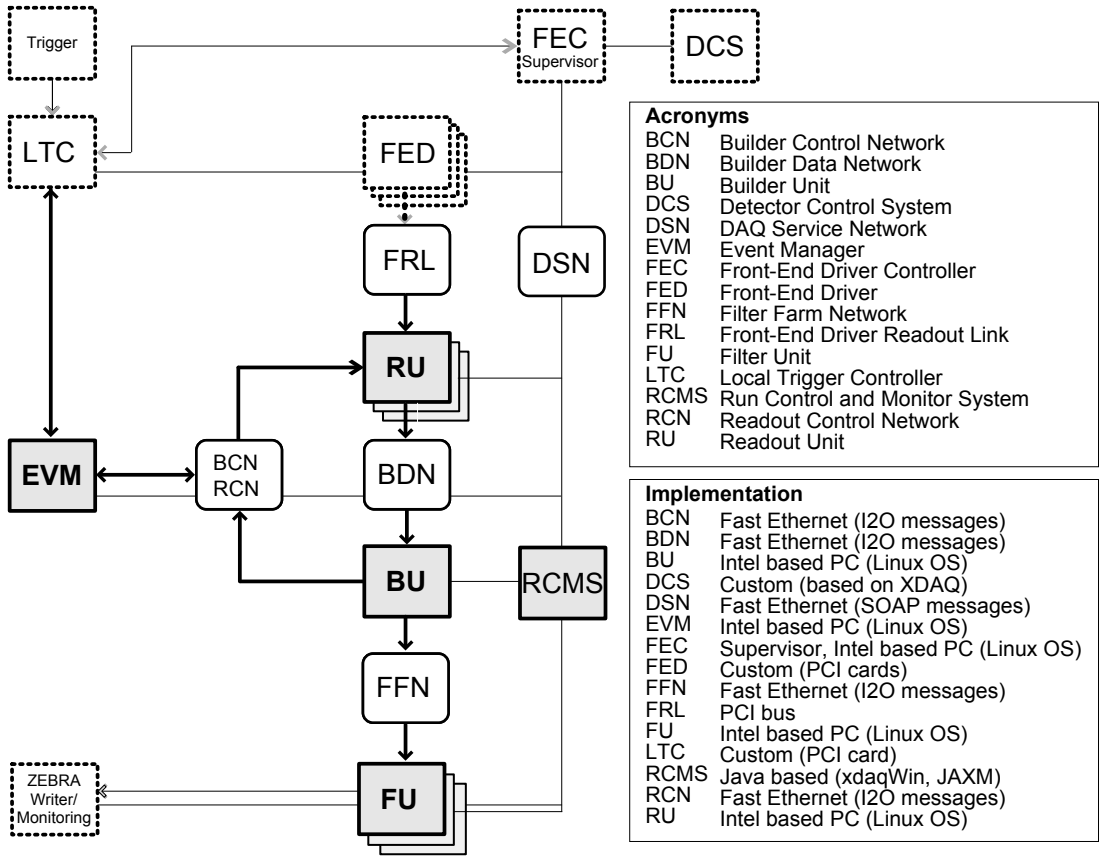


Figure 11-4 Tracker testbeam and calibration system. Dashed rectangles denote systems, developed by the subdetector collaboration, grey shaded rectangles identify XDAQ supplied components. Interconnects are shown as rounded rectangles.

The flexibility and scalability of XDAQ was demonstrated by its use in different configurations. It was possible without modifications to transfer an existing small setup at PSI (Zurich, Switzerland) to tracker subsystem (Rod, Petal) tests. This system was easily scaled up in terms of computer resources and interfaces to instrumentation in the testbeam environment at CERN. Commissioning of the new system took 2 hours as opposed to 30 hours with previous systems. At a rate of 2000 events per spill (500 Hz with an av-

erage event size of 20 kB), a maximum data throughput of 100 Mb/s was achieved, which corresponds to the available Fast Ethernet capabilities. During one operation, that consisted of five days of continuous and uninterrupted data taking, around 600 GB of data were produced for analysis. Users gave positive feedback on the simple interface for controlling the distributed system. It was noted that novices could easily operate and configure the system themselves, in particular the application of configuration changes to re-distribute processing tasks to multiple computers.

11.5.2 Muon Chamber Validation

The 250 chambers that make up the muon barrel spectrometer need to be tested through cosmic ray acquisition runs before shipping to and after their arrival at CERN. Tests with muon beams at high data rates together with coupling to other subdetectors are also necessary to validate the detector behavior under realistic conditions. The observables include occupancy calculations for determining the particle hits for each channel, drift time histograms and the response to dedicated test pulse events. Thus, a data acquisition system had to be put in place to perform these tasks. XDAQ was used as the generic platform to implement this system. As outlined in Figure 11-5, the software required customization of three parts, the local trigger controller (LTC), readout of custom electronics through the VME bus and interfacing to a data storage/analysis back-end. For the acquisition task, the generic event builder components were used. Run control was implemented by the prototype system as described in Section 12.6.1, "RCMS for Small DAQ Systems". As opposed to the tracker experience, slow control was performed by a separate, Windows based system.

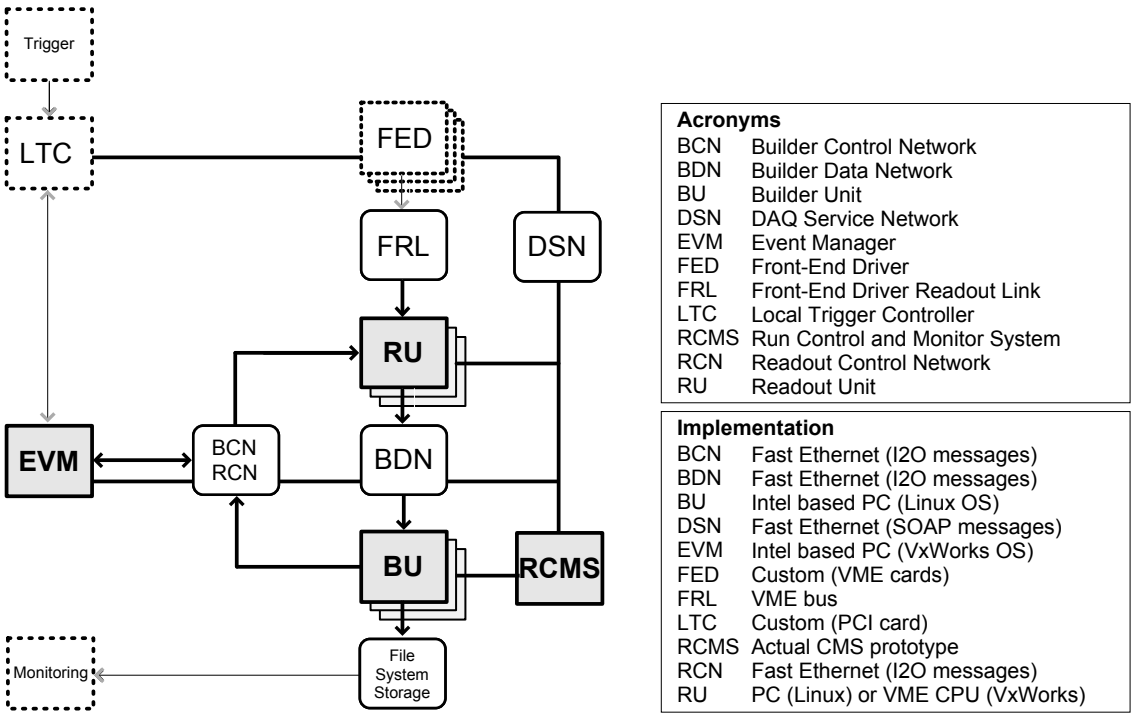


Figure 11-5 Muon testbeam and validation system. Dashed rectangles denote systems, developed by the subdetector collaboration, grey shaded rectangles identify XDAQ supplied components. Interconnects are shown as rounded rectangles.

A similar setup was used during a test period in 2001 for the Gamma Irradiation Facility (GIF) facility at CERN. In this case, a muon chamber was coupled to a RPC detector and a beam chamber to verify the simultaneous response of the RPC and muon chambers. Update of the existing system went smoothly.

As a result of this project, several requirements were identified, mainly covering configuration, control and monitoring. Hardware platform heterogeneity (different bus systems, byte ordering and data alignment rules), as well as the presence of two different operating systems platforms, posed a challenge to the interoperability among all system components. The supported platforms included PowerPC based VME processors running the VxWorks real-time operating system, and Intel based personal computers with both Linux and VxWorks.

The provided abstraction fitted the need to switch between processor and operating system types without extra work. The required event rate of 10 kHz and the peak output of 4 MB/sec were well absorbed by the hardware and software installation in place. The variance of the event sizes stressed the buffer management system of XDAQ and thus proved the robustness of the design. Uptime measured over two weeks was 60%, including penalties from changing hardware configurations. From the initial intent to create the system to its completion, including the learning phase, six man months had to be invested. A system developed from scratch in the same time scale would not have provided the seamless integration with later appearing components (e.g. silicon beam telescope) and the ability to efficiently carry out modifications and configuration changes.

Through these experiences, confidence has been gained that the proposed design of the software infrastructure can fulfill the enumerated requirements.

11.6 Summary and Conclusions

This sub-section summarises the present status of the XDAQ framework development. The existing software is a proof of concept of the proposed design. The working prototype has limited functionality that covers the minimum requirements for demonstrating a simple data acquisition system. Confidence has been gained through the use of the software in test beams and laboratory environments such as the DAQ column that such a design approach can be scaled to the final data acquisition system. All dimensions of the system were investigated from readout and network communication to the graphical user interface for control and configuration.

A C++ framework has been developed that incorporates preliminary support for binary message-passing following the I_2O standard, and SOAP messages for platform independent communication. In this context several technologies have been exercised, including Apache Xerces for the DOM representation of XML messages, and Java web services for interfacing to users and external systems. Pre-production peer transports for network communication on all supported platforms (Intel based Linux, Power PC Mac OS X, and Power PC and Intel based VxWorks) now exist for TCP/IP, UDP and a subset of HTTP. FIFO communication supports applications running within the same XDAQ executive/process. A peer transport interfacing Myrinet GM has been implemented for tests on Intel based Linux PCs, as well as a simple peer transport for testing raw Ethernet communication on VxWorks to support the muon testbeam. A library for direct hardware access for PCI and VME devices has been implemented for the Linux operating system. A Linux driver together with a XDAQ application component to interface the generic PCI card for reading out FED devices has also been developed.

A prototype application component for interfacing XDAQ applications to various databases technologies in a standard but technology independent manner has been developed. The component currently supports MySQL, Oracle 8i and the file system. A user interface has been written in Java to exercise access (con-

trol and configuration) to a XDAQ distributed system through SOAP messages. Two prototype event builder applications have been developed to test and study two event builder protocols that are currently used for test beam systems. A third event builder is being developed for the fully functional data acquisition system.

The implementation of the presented design requires a significant development effort. All the required functionalities outlined in this section must be put in place with the level of robustness that ensures the proper operation of the CMS data acquisition system. The proposed layered architecture of the XDAQ middleware will allow the porting of all application components to other platforms, should that become necessary.

11.7 References

11-1 P. Bernstein, “Middleware: A Model for Distributed Systems Services”, Communications of the ACM, 39 (1996) 86.

11-2 J. Gutleber and L. Orsini, “Software Architecture for Processing Clusters Based on I_2O”, Cluster Computing 5(1):55-65, Kluwer Academic Publishers, 2002.

11-3 B.N. Bershad et al.,”Extensibility, Safety and Performance in the SPIN Operating System”, in Proceedings of the Fifteenth ACM Symposium on Operating System Principles, pp. 267-284, 1995.

11-4 I_2O Special Interest Group, Intelligent I/O (I_2O) Architecture Specification v2.0, 1999, at http://www.intelligent-io.com

11-5 J. Boyer, “Canonical XML Version 1.0, W3C Recommendation 15 March 2001”, http://www.w3.org/TR/xml-c14n

11-6 P. Pardyak and B.N. Bershad, “Dynamic binding for an extensible system”, in Proceedings of the 2nd USENIX Symposium on Operating Systems Design and Implementation, 1996, pp. 201-212.

11-7 D. Box et al.,” Simple Object Access Protocol (SOAP) 1.1, W3C Note 08 May 2000”, http://www.w3.org/TR/SOAP

11-8 R. Fielding et al.,”Hypertext Transfer Protocol - HTTP /1.1”, IETF RFC 2616, June 1999, http://www.w3.org/Protocols/rfc2616/rfc2616.html

11-9 G. Glass, ”Web Services: building blocks for distributed systems”, Prentice Hall, 2002.

11-10 “Java API for XML Messaging (JAXM) Specification v1.0”, Sun Microsystems, 901 San Antonio Road, Palo Alto, California 94303, USA, October 2001.

11-11 A. Le Hors et al.,”Document Object Model (DOM) Level 3 Core Specification, Version 1.0, W3C Working Draft 13, September 2001”, http://www.w3.org/TR/2001/WD-DOM-Level-3-Core-20010913

11-12 The Java Analysis Studio, http://jas.freehep.org

11-13 Oracle database product documentation: http://oradoc.cern.ch, provided by the CERN Oracle support (http://wwwinfo.cern.ch/db/oracle/)

11-14 The MySQL open source database: http://www.mysql.com/

11-15 The PVSS II system from ETM provided through the CERN IT division, http://itcobe.web.cern.ch/itcobe/Services/Pvss/

11-16 DCS-DAQ Interface Requirements, see http://cern.ch/xdaq

11-17 Concurrent Versions System, http://www.cvshome.org/

11-18 GNUMake project, http://www.gnu.org/software/make/make.html

11-19 http://sourceforge.net/projects/xdaq/ , hosted by the Open Source Development Network, 50 Nagog Park Acton, MA 01720, USA.

11-20 Zero G Software Inc., 514 Bryant Street, San Francisco, CA 94107, USA. See also http://www.zerog.com

11-21 Apache Software Foundation XML project, http://xml.apache.org

11-22 Exist DB, http://exist.sourceforge.net and Apache XIndice, http://xml.apache,org/xinidice

11-23 http://www.xmldb.org

11-24 R. Johnson, “Tcl and Java integration”, Sun Microsystems Laboratories, 901 San Antonio, MS UMTV-29-232, Palo Alto, CA 94303-4900, USA, 1998. See also http://www.tcl.tk/software/java

11-25 J. Gutleber et al.,”Architectural Software Support for Processing Clusters”, in Proceedings of the IEEE International Conference on Cluster Computing, pp. 153-161, November 28 - December 1, 2000, Chemnitz, Germany, IEEE Computer Society Press, Los Alamitos, California.

12 Run Control and Monitor System

The Run Control and Monitor System (RCMS) is the collection of hardware and software components responsible for controlling and monitoring the CMS experiment during data taking. It provides physicists with a single point of entry to operate the experiment and to monitor detector status and data quality. The interface enables users to access and control the experiment from any part in the world providing a “virtual counting room”, where physicists and operators can perform all programmable actions on the system, effectively taking shifts from a distance.

In order to achieve its goals, the RCMS interoperates with the Detector Control System (DCS), the data acquisition components and the trigger subsystem as shown in Figure 10-2, through the services provided by the distributed processing environment (see Section 10.5, "Cross-Platform DAQ Framework: XDAQ"). For configuration, user administration and logging, the RCMS makes use of a database management system.

The RCMS is the master controller of the CMS DAQ when the experiment is taking data. It instructs the Detector Control System (DCS) to act according to the specific needs of a data-taking session.

The three main functions of the RCMS are

- control and monitoring with the support of the DCS to ensure the correct and proper operation of the CMS experiment
- control and monitoring of the data acquisition system
- provide user interfaces and allow users to access the system from anywhere in the world

The RCMS views the experiment as a set of partitions when performing these three functions. A partition is the smallest grouping of entities that can be configured and operated independently. This includes configuration, monitoring, error handling, logging and synchronization with other subsystems.

The RCMS architecture faces similar requirements to applications in the Internet world, in particular:

- connecting many clients distributed in a large geographical area to a set of servers and databases
- providing secure access
- scaling up in terms of connectivity to the servers and transactions per second of the databases

The design and prototyping of the RCMS has followed these rapidly developing environments. Many of these widely-spread software tools follow an open-source approach, thus facilitating the integration with other software packages. As discussed later in this chapter, it is planned to maximise the use of standard software technologies in the RCMS.

12.1 Requirements

This section presents the requirements of the RCMS grouped by the following functionalities: configuration, control and monitoring. It also includes the requirements for the user interface.

12.1.1 Configuration Requirements

The system will provide means for defining and storing persistently the configuration of any subsystem and collections of subsystems. It shall be possible to retrieve and apply stored configurations. The system must also support run-time inspection and modification.

The RCMS views the experiment as a collection of partitions, where a partition is the smallest configurable grouping of units. The use of partitions satisfies the requirement that independent sections of the experiment can be run concurrently. In order to support partitions, the following requirements must be met:

- It must be possible to define, configure, control and monitor any possible partition of the CMS apparatus.
- Multiple partitions can run concurrently. Resources can be allocated to and shared between multiple partitions.
- It must be possible to store and retrieve all partition configurations to and from persistent storage.
- It must be possible to group several resources into a partition according to a set of pre-defined rules. Validation must be possible based on the application of these rules.

12.1.2 Control Requirements

The RCMS shall perform actions on any sub-system within a partition in an orderly manner through the definition of states and state transitions. As described in Section 11.2.2, "Application Interfaces and State Machines", it must be possible to query the current state of any sub-system within a partition. The RCMS will communicate with the sub-systems of a partition according to the protocol specified in Section 11.2.5, "Protocols and Data Formats".

The RCMS should be able to perform actions within a given maximum time period. Configuration and setup of partitions will take in the order of minutes, their control (state change, execution of commands) in the order of seconds, and monitoring will range from microseconds to minutes depending on the amount of required data.

It shall be possible to control any individual sub-system within a partition. As an example, it will be possible to suspend and resume the execution of a sub-system.

It will be possible to run RCMS in “user training” mode. Actions on the sub-system resources in this mode are not actually performed but emulated.

The RCMS shall be responsible for keeping the operation of the detector in a coherent state in the presence of concurrent accesses from multiple users and systems.

12.1.3 Monitor Requirements

Monitor information is subsystem dependent, and in the context of the RCMS refers to performance parameters, statistics, errors, warnings, debugging information and raw event data. Facilities must be provided for the logging of all monitored data, to categorize and visualize the information history of detector operation. The level of detail must be sufficient to unambiguously identify the actions performed on or by individual subsystems including type, severity, time stamp, sub-system identification and action description. In order to correlate current and previous system states, a common synchronized clock with an accuracy of milliseconds must be available.

12.1.4 User Interface Requirements

The user interface shall be able to access all of the functions provided by the RCMS, and will allow multiple users to access the experiment concurrently. Users shall only be presented with the functionalities for which they are authorized. Therefore the user interface must have access to authorization and authentication mechanisms. Visual components must be customizable by the user to reflect and focus on the sub-systems being controlled and monitored. It should be possible to record and playback sequences of user actions performed on the graphical user interfaces. It must be possible to automate all interactive user operations via scripting technologies, hence the RCMS can be run without the use of graphical user interfaces.

The available interfaces must be accessible remotely from different platforms and systems. The interface will not be limited to scripting and graphical interfaces, but shall also include an API (Java and C++) and a set of protocols to support such media as e-mail, SMS and voice recognition. The execution of user interface processes should be independent of the execution of the RCMS services. This enables a partition to continue running when no user interfaces are active.

12.2 Architecture

The architecture of the DAQ system implies that there are roughly $O(10^4)$ objects that need to be controlled. The RCMS architecture must be capable of scaling up to this order of magnitude. The approach of constructing hierarchies of distributed control applications will be taken in order to achieve this goal.

The logical layout of the RCMS is shown in Figure 12-1. It consists of four types of elements, a Session Manager (SMR), a Sub-System Controller (SSC), a user interface and a set of services that are needed to support specific functions like security, logging and resource management.

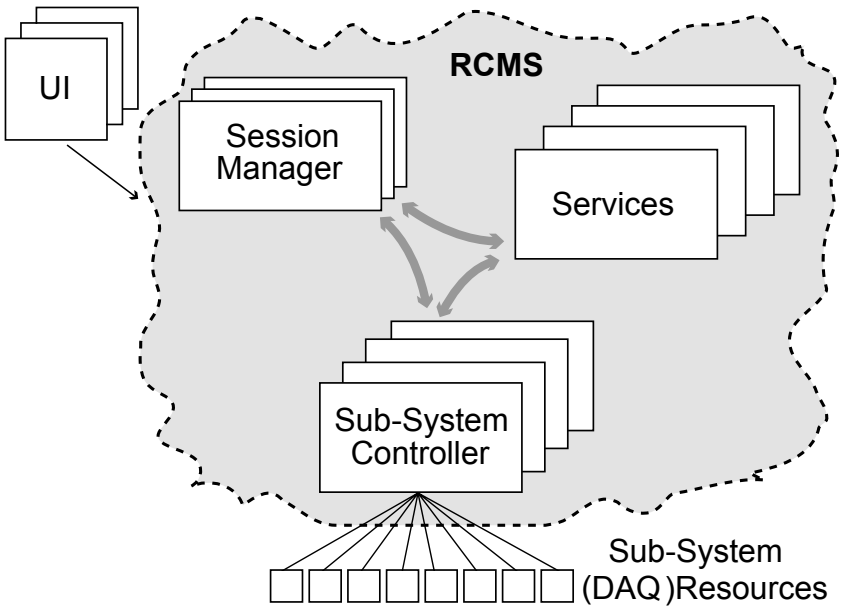


Figure 12-1 RCMS logical layout.

The execution of a partition is defined as a “session”. A session is the allocation of all the hardware and software of a CMS partition needed to perform data-taking. Multiple sessions may coexist and operate concurrently. Each session is associated with a Session Manager (SMR) that coordinates all actions. The SMR accepts control commands and forwards them through a sub-system controller (SSC) to the components under its control.

Single commands that are initiated by the user will be interpreted, expanded into sequences of commands where applicable, and routed to the proper sub-system resources by different RCMS components. Status information resulting from the actions corresponding to the commands submitted is expected asynchronously and is logged and analysed by the RCMS. The RCMS also handles all information concerning the internal status, malfunctions, errors and, when required, monitor data from the various DAQ subsystems. The sub-systems controlled by the RCMS, corresponding to the main elements of the DAQ are shown in Figure 12-2.

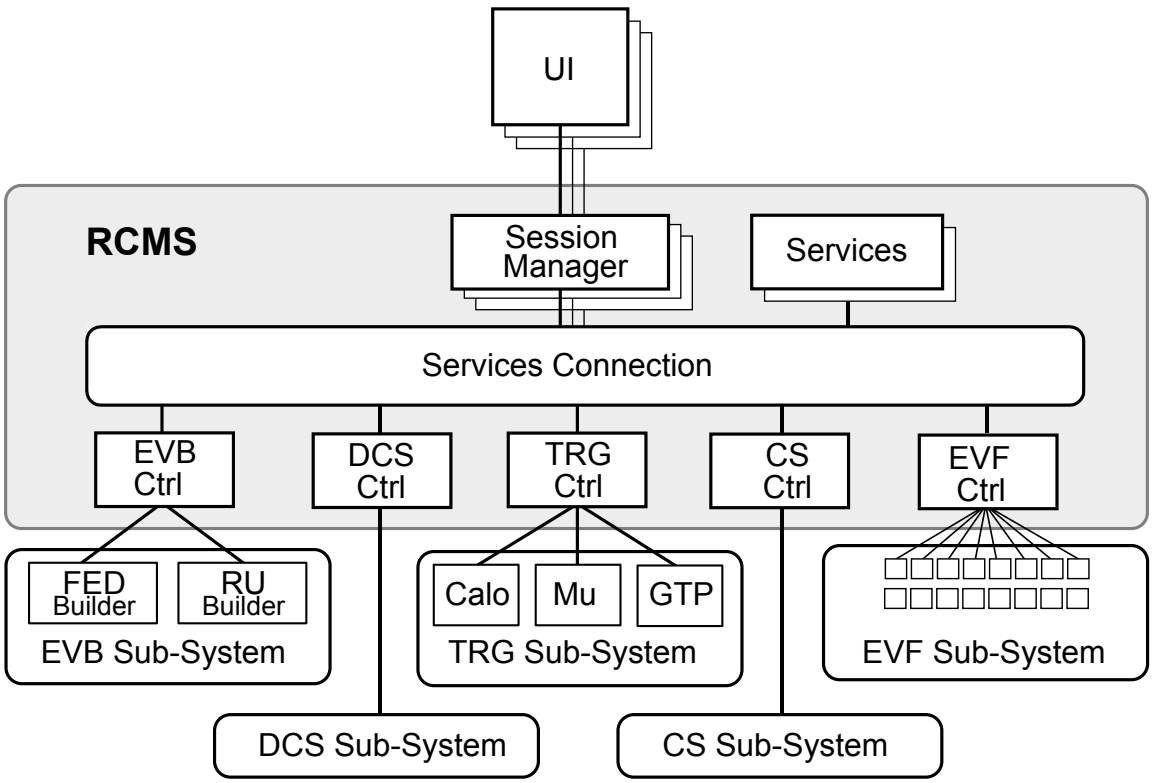


Figure 12-2 Session Managers and Sub-Systems defined in the RCMS.

Subsystems are connected for control and monitor purposes via the “DAQ Service Network” (DSN), which is based on switched Ethernet. A subsystem is composed of DAQ resources which are any hardware or software components that can be managed directly or indirectly through the DSN. The resources currently defined for the different subsystems are listed in Table 12-1. The DSN can also be used for local data acquisition.

Table 12-1 Subsystems and their resources, as defined in the RCMS.

Subsystem	Resources defined
Event Builder (EVB)	FED Builder, RU Builder (RU, BU and EVM)
Event Filter (EVF)	Filter Units
Trigger (TRG)	Calorimeter Trigger, Muon Trigger, Global Trigger
Detector Control System (DCS)	Values accessible through the SCADA product
Computing Services (CS)	Storage, Monitor Services

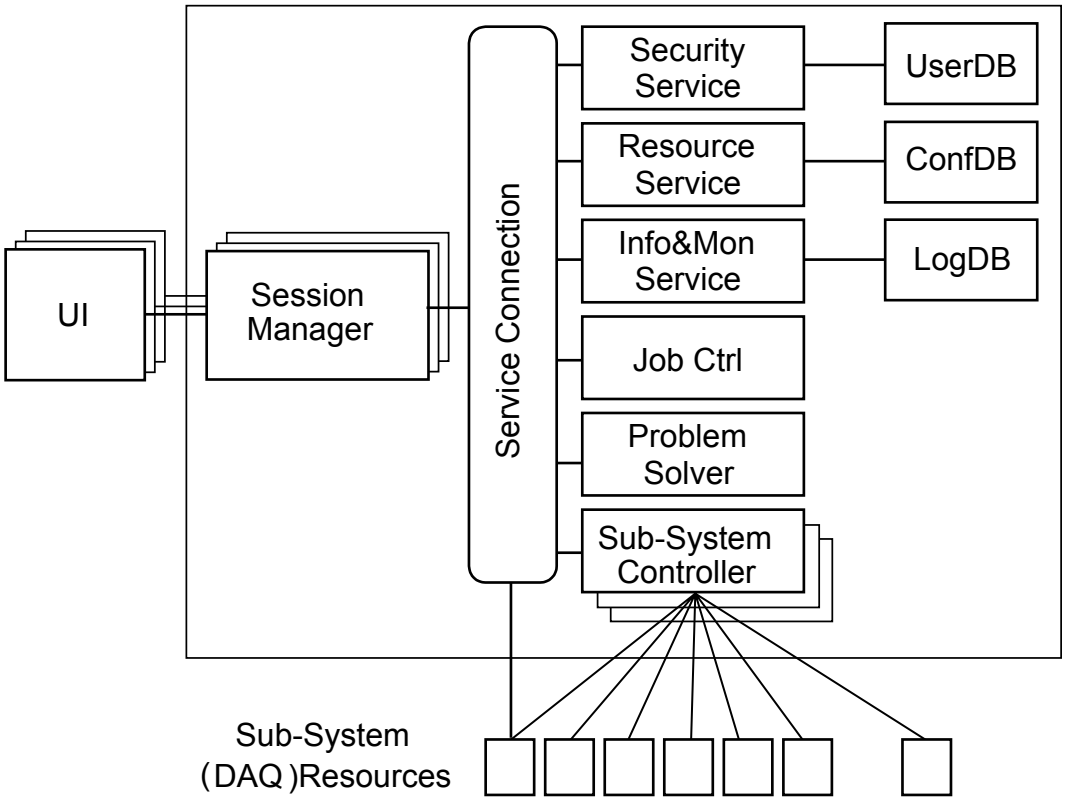


Figure 12-3 Block diagram of the Run Control and Monitor System.

The block diagram of the RCMS, showing the various services to support interaction with users and to manage subsystem resources, is shown in Figure 12-3. These services are:

- Security Service (SS). Provides login and user account management functions.
- Resource Service (RS). Provides access to the configuration database (ConfDB) that stores all the information about partitions and DAQ resources. The RS also handles the global “alive heart beat” of the CMS DAQ.
- Information and Monitor Service (IMS). Collects messages and monitor data coming from DAQ resources or internal RCMS components and stores them in a database (LogDB). The IMS can distribute these messages to any external subscriber, with the possibility of filtering them according to a number of criteria, e.g. the message type, the level of severity and the source of the message. A

similar mechanism is also used to distribute monitor data. The IMS also supports logbook facilities.

- Job Control (JC). Starts, monitors and stops, when necessary, the software elements of the RCMS, including the data acquisition components.
- Problem Solver (PS). Uses information from the RS and IMS to identify malfunctions and attempts to provide automatic recovery procedures where applicable.

The above set of services can be distributed over a number of computing nodes for scalability and fault tolerance.

12.3 RCMS and DAQ Operation

The CMS data acquisition system can be operated in one of two ways:

- Partitioned mode, in which one or more DAQ partitions are used (see Section 4.4). All subsystems are partitioned according to the specific data-taking requirements, e.g. calibration runs, debugging of a specific subdetector. In this mode, multiple DAQ sessions can run concurrently. A special case of this mode is a “physics run” in which all operational detectors are used in a single partition. A physics run includes all five DAQ elements, namely the trigger, the event builder, the event filter sub-systems, the detector control system and computing services.
- Stand-alone mode, in which the detector partitions work independently using an individual trigger and individual data acquisition system. The stand-alone DAQ uses FED supervisor processors as Readout Units and the DSN for the event builder[1]. FED supervisor processors will be equipped with XDAQ software both to control and setup the FEDs and to acquire data when they are working in stand-alone mode. In this case BU and FU functionalities (including event storage capability) can be embedded in the FED supervisor.

12.3.1 State Definitions and System Synchronization

All RCMS subsystems can handle basic commands like “halt”, “configure”, “enable”, “disable”, “suspend” and “resume” that correspond to the state transitions defined for the XDAQ executive (see Section 11.2.2, "Application Interfaces and State Machines" and Figure 12-4). The responses to the execution of these commands are reported to the IMS. All subsystems implement a state machine similar to the one in Figure 12-4

Controlling a large number of resources increases the probability of one or more resources failing. Initialization and configuration must tolerate these failures if they are related to non-critical components. The synchronisation of state changes within a large number of resources also requires an efficient control mechanism to fulfill timing requirements.

Faults can significantly change run conditions, and therefore their occurrence together with the state transitions of all resources must be recorded. Critical faults will stop the system and raise alarms, whereas non-critical faults will be tolerated. However a combination of non-critical faults can generate a critical system fault.

1. These are the BDN, BCN and RCN described in Chapter 6, "Event Builder Networks".

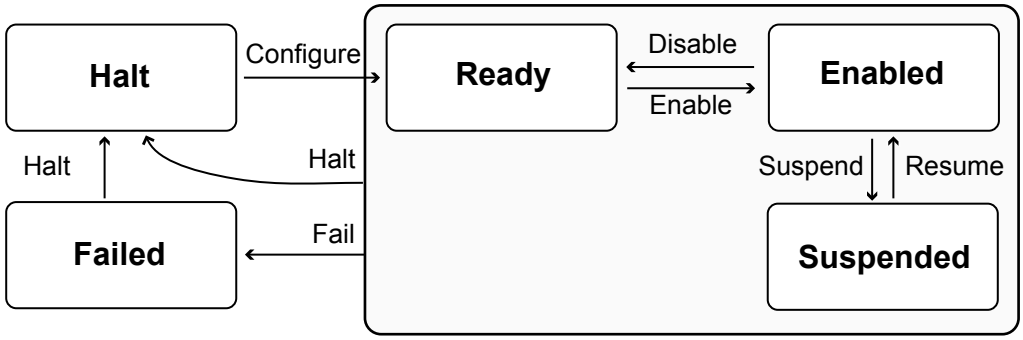


Figure 12-4 State diagram of a generic DAQ component.

12.3.2 Run Definition

A data run is defined as the interval between the starting and stopping of data taking. Traditionally, a new run was started either because of major changes in run conditions or because of DAQ operations. Run conditions include such information as calibration constants and trigger tables, whereas DAQ resets and tape changes are examples of DAQ operations.

The start of a new run entails a flushing of the event data in the system. This has to be avoided, because the buffer sizes and number of components to be controlled in the CMS experiment imply a significant amount of deadtime.

With current database technology, it is possible to continuously record all significant variations in the run conditions that affect the quality of acquired data. This capability no longer necessitates starting a new run when a significant change in the run conditions occurs, thus reducing the number of times a system flush has to be performed. Time tags will be used to associate run conditions with event data.

12.3.3 Sessions and Partitions

The subdetectors, such as the muon detectors, tracker and calorimeters, represent “natural” partition boundaries within the DAQ system. These partitions are reflected in those DAQ sub-systems that have to deal directly with the detectors and their electronics like DCS and the event builder. These physical partitions impose constraints on the definition of the logical partitions that will be stored in a database. The same is true for the trigger where the partitions are imposed by the TTC distribution topology. However, this is not the case for the event filter where partitions are freely configurable, as they are only influenced by the computing power required by the session to which they belong.

The DCS, from the point of view of the RCMS is an external system with its own independent partition management. However, during data-taking, the RCMS instructs the DCS to set up and monitor partitions corresponding to the detector elements needed for the data-taking run in question. The RCMS also has access to all partition information in the DCS.

Before a partition can be used it must be validated. It is invalid if it does not include a complete and consistent set of elements needed for a DAQ run. The completeness and consistency check depends on the sub-system for which the partition is defined. Taking an event builder as an example, the check includes

the verification that at least one RU, one BU and one EVM are included. A valid partition is activated by allocating physical sub-system resources to it. The activation can fail if the partition tries to use unavailable resources. The activation of a partition corresponds to the creation of a session ready to accept commands. A running session can be joined by several user interfaces.

12.3.4 Interface to DCS

The RCMS instructs the Detector Control System (DCS) to act according to the specific needs of a data-taking session. At all other times, the DCS runs in stand-alone mode. The DCS, as described in Chapter 13, "Detector Control System", can be accessed through a specific XDAQ application (see Section 11.2.6.2, "Common Application Components").

12.4 Design of the RCMS Components

This section describes the design of all RCMS components. For each component an API (C++, Java) has been defined and is described in [12-1].

12.4.1 Session Manager (SMR)

The SMR is created to coordinate a session by the RS when that session is opened. The RS also provides the SMR, at the time of its creation, with all the needed information concerning the subsystems and the composition of the partitions making the session. The lifetime of the SMR corresponds to the lifetime of the session and it is deleted when the session is closed. Once the session is opened, it can be joined by several users. The SMR coordinates user access to the session and propagates commands from the users to the SubSystem Controllers (SSCs) which belong to the session. Figure 12-5 shows a typical interaction diagram where a command is issued from a user interface that has already joined a session. The example shown in the figure includes two SSCs (A and B). In this case SSCs inform the Information and Monitor System (IMS) of the status change of the subsystems they control. The diagram also shows a system state display that retrieves, at regular time intervals, the global status of the system.

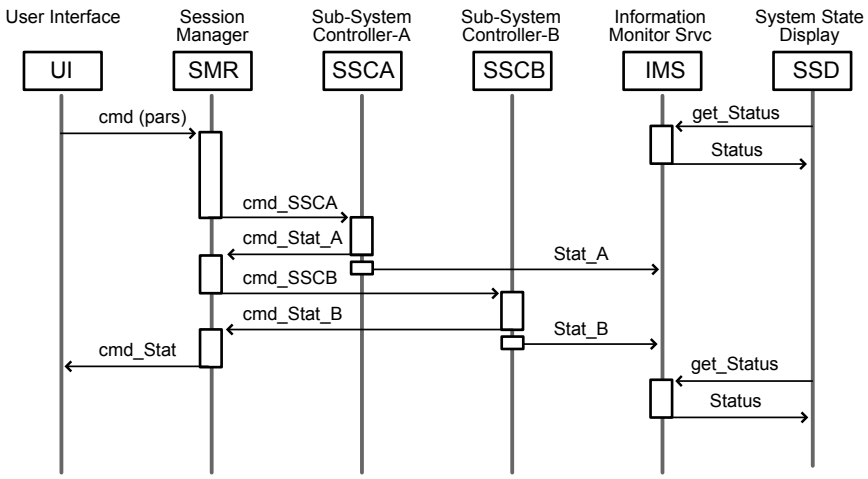


Figure 12-5 SMR interaction with the other RCMS components.

12.4.2 Security Service (SS)

This service provides the RCMS login facilities for user authentication and authorization, providing data encryption when demanded. The SS manages a database (UserDB) that stores profiles for each user including access rights and a personalized work history, e.g. open sessions and defined partitions. The SS keeps track of the user context such as the connection and client interface type (browser, stand-alone application, etc.).

12.4.3 Resource Service (RS)

This service handles all the resources of the DAQ system through partitions and their sessions. Resources can be discovered, allocated and queried. A resource can be any hardware or software component that can be managed directly or indirectly through the DAQ service network. Figure 12-6 shows the block diagram of the Resource Service.

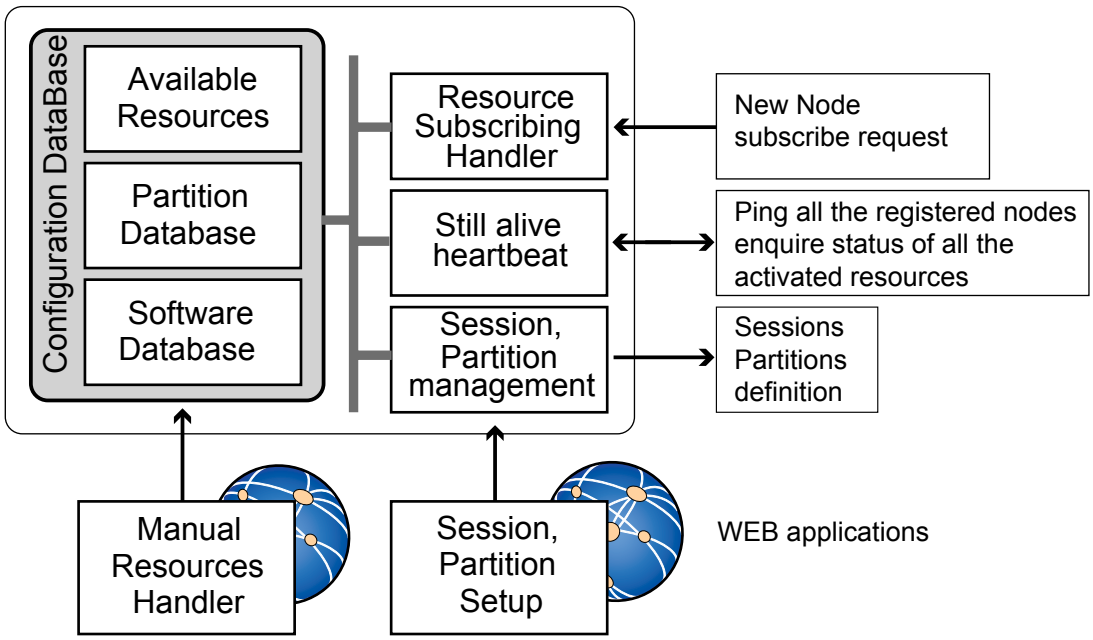


Figure 12-6 Block diagram of the Resource Service.

In accordance with the requirements, partitions can only use “available” resources whose state (running or down) is verified when the partition is activated for data-taking. It is the responsibility of the Resource Service to check for resource contention with other active partitions. The RS must also signal temporary unavailability when activating the partition.

A periodic scan of all the required registered resources, the “alive heartbeat”, will keep the database of available resources up to date. Two types of scan are foreseen:

- ping scan: Checks if resources are reachable, and is expected to take a time of O(s)
- status inquiry: the status is retrieved and then dispatched to the IMS, described in the following section. This type of scan is expected to take a time of O(minutes)

12.4.4 Information and Monitor Service (IMS)

This service is responsible for collecting information from all subsystems in the DAQ. Figure 12-7 shows the block diagram of the IMS.

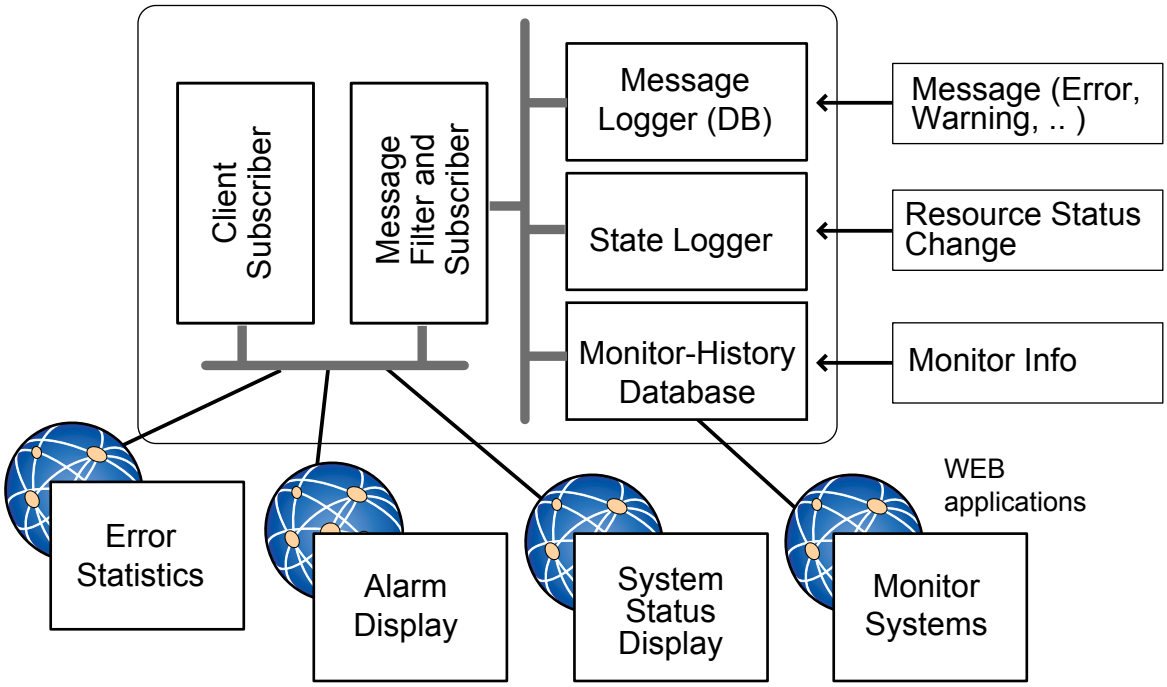


Figure 12-7 Block diagram of the Information and Monitor Service.

There are several types of messages collected from the subsystems that are catalogued according to their severity level and type such as: informational, monitoring, warning and errors. Data can be provided in numeric formats, histograms, tables and other constructs.

Each record of information sent to the IMS contains the following items:

- time stamp: when the message/monitor information was generated
- message identification
- the source of the information, uniquely identified by the session, sub-system, partition, software application and hardware board
- information type: informational, monitoring, warning and error
- specific fields according to the information type

The time stamp facilitates tracking of different events, as well as correlating them. All resources are synchronized with time servers providing a global time accuracy of O(ms).

The IMS collects and organizes the incoming information in an internal database and publishes it to the IMS subscribers. Subscribers can register for specific messages that can be selected on the basis of the information source, the error severity level, the type of messages or a combination of these tags. The IMS also monitors event data providing permanent recording and snapshot buffers for subscribers. Any information can also be retrieved by a client via simple query procedures.

The IMS is also responsible for storing and maintaining the run logbook. This type of information is managed by the Session Manager that uses the IMS back-end facility to store them.

12.4.5 Job Control (JC)

Job Control is a utility that allows the remote execution and supervision of processes as outlined in Section 16.6.1, "Cluster Management". This tool is necessary to start, monitor and stop, when needed, the software infrastructure of the RCMS on the main management nodes and subsystem resources., i.e. the various managers and the databases.

12.4.6 Problem Solver (PS)

The Problem Solver (PS) is a central service of the RCMS with the main task of collecting alarms from the DAQ system and potentially determining recovery procedures. Figure 12-8 shows the block diagram of the PS.

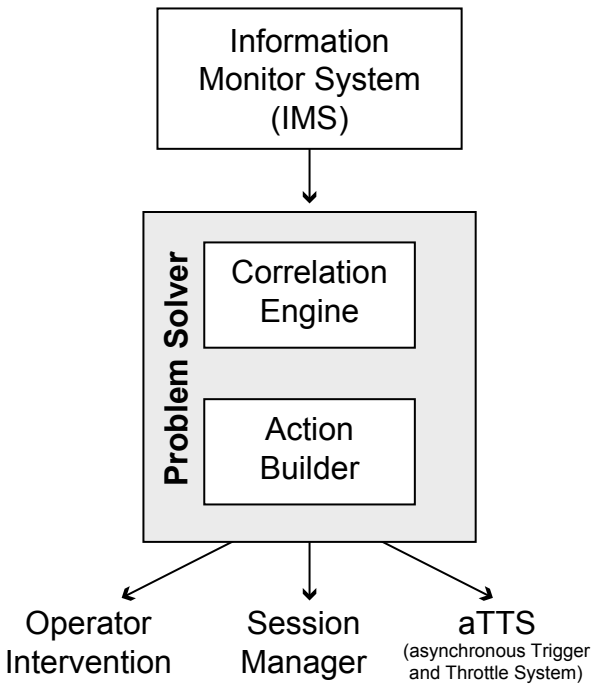


Figure 12-8 Block diagram for the Problem Solver.

The PS subscribes to the IMS to receive the information it requires from the DAQ components it is interested in. This information is processed by a correlation engine and the result is used to determine a potential recovery action. When the PS cannot determine such an automatic recovery procedure, it simply informs the system operator, providing any analysis results the PS may have obtained.

The PS must interact with the asynchronous Trigger-Throttling System (aTTS, see Section 5.2.3) to provide the back-pressure signal to the trigger to block data taking while the DAQ system recovers from failures. This may be used to allow the detachment from the DAQ chain of a given component, e.g. a

Readout Unit or a Builder Unit, and to reconfigure the system. Other faults (e.g. a misalignment of the event fragments read out of the FEDs) could involve a reset of the front-end electronics, which implies that access to the fast reset signal is necessary.

Automatic recovery procedures are communicated to the Session Manager that controls the component in question. The procedures vary from controlling single DAQ components to the entire system.

12.4.7 Sub-System Controller (SSC)

An SSC, as shown in Figure 12-9, consists of a Function Manager (FM) and a local database (DB) service. There is one FM per partition that receives requests from an SMR and transforms them into the corresponding requests for actions that are sent to the subsystem. Since partitions can be nested, the SSC can be organized in a hierarchical manner.

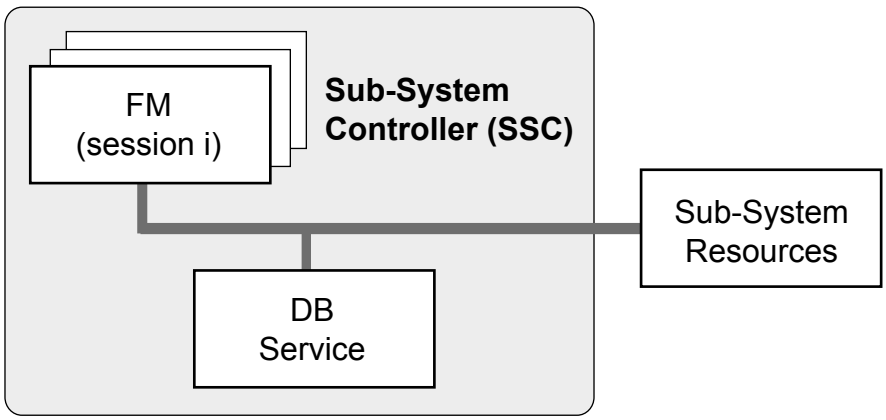


Figure 12-9 Block diagram of a SubSystem Controller.

A hierarchy can map to a corresponding physical architecture, with a shared load among different processors. An example of this is shown in Figure 12-10. The local database (DB) service can be used as a proxy to the RS and IMS to facilitate software and configuration download and monitoring.

12.5 Software Technologies

Web technologies and related developments play a fundamental role in the implementation of the proposed design. In this section we review some of the present trends that have been followed.

12.5.1 Web Technologies

Web technologies and in particular Web Services have been chosen because they have been designed to interconnect highly heterogeneous and distributed systems, therefore implying a rich choice of available tools and solutions. Among the available technologies, the XML data format and the W3C standard SOAP protocol have been adopted (see Section 11.2.5, "Protocols and Data Formats") as the main means for communication. For more information about these and their associated tools see Section 11.2.2, "Application Interfaces and State Machines".

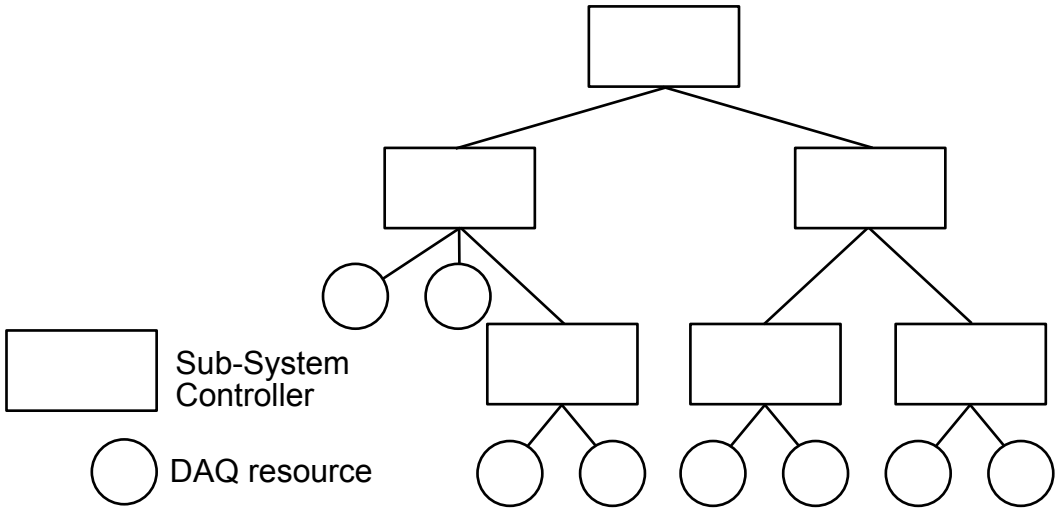


Figure 12-10 Structure of a generic sub-system.

12.5.2 Security

Security issues are crucial for a system such as the RCMS that has clients distributed around the globe. A weakness in security can have serious consequences on the correct operation of the system. Security is a problem in its own right and the RCMS has therefore chosen to leverage existing solutions in the scope of widely distributed systems. The Grid Security Infrastructure (GSI) of the Grid Forum has submitted an Internet Draft [12-2] which provides some preliminary guidelines to GRID infrastructures. This is a potential candidate to address the security needs of RCMS, because it has been proposed for the security infrastructure of the LHC Computing Grid (LCG) project [12-3].

12.5.3 Database

Databases will be used by the RCMS to handle different types of information, ranging from hardware and software configurations to user identification parameters and system book keeping information. Relational database management systems are appropriate for this task and furthermore guarantee the required level of robustness and scalability. It should be noted however that native XML database technologies exist and are being investigated because they remove the need to transform between XML data and relation table structures. For more information about the chosen technologies see Section 11.3.3, "Software Development Environment".

12.5.4 Expert Systems

As described in Section 12.4.6, "Problem Solver (PS)", the PS is a central service of the RCMS. Its main task is to correlate different types of input information (errors, warnings, messages, etc.) to identify global error conditions in the DAQ system and to propose, wherever possible, appropriate recovery actions. The PS can be implemented by conventional procedural languages, however artificial intelligence techniques like rule-based expert systems permit modelling at a higher level of abstraction, thus allowing the formulation of problems in almost human terms. Commercial [12-4] and public-domain [12-5] expert systems

exist. For example, Jess [12-6] could be easily integrated into the current Java-based infrastructure, and is currently being evaluated.

12.6 RCMS Prototypes

Prototypes have been developed to provide proof of principle for the proposed RCMS design and the technologies it uses. A prototype has been implemented for the small DAQ systems used for experiment production and the test beams. Two prototypes addressing specific functions, hereafter referred to as demonstrators, have also been developed to study scalability issues. Medium scale O(100) test-bed systems are used in this case to emulate parts of the hierarchical structure of the final DAQ system.

12.6.1 RCMS for Small DAQ Systems

A prototype RCMS system has been developed for test beam setups, and the detector validation, trigger and front-end electronics test-benches. The system is based on the XDAQ framework as described in Chapter 11 and implements all services shown in Figure 12-3 with the exception of the “Problem Solver”. Multiple sessions can run concurrently. DB proxies and management systems are not implemented in the sub-system controllers, due to the small number of nodes. The Job Control service (JC) is also not used, since XDAQ executives were started automatically at boot time. The following paragraphs give a brief report on the design and tools used to implement the prototype. A full description of the system can be found in [12-7].

The services and managers have been implemented as a collection of Java servlets using the Apache Tomcat platform. Figure 12-11 displays a graphical representation of the system when a single Tomcat [12-8] is used. SOAP messages are used as the primary communication mechanism. Relational (MySQL) and native XML (XML:DB [12-9]) database technologies have been exercised for persistent storage. Figure 12-12 shows the implementation of the Resource Service. The Castor [12-10] tool has been used to map XML data structures to Java class definitions.

The session and function managers incorporate a Finite State Machine (FSM) to track the status of the corresponding components. The FSM is defined by an XML document and actions are declared through Java classes. Both definition and implementation are managed by the Resource Service (RS). When a session is opened, the RS launches the session and function managers, passing them the corresponding state machine definitions along with the Java code as well as the information about the hardware and software configuration.

The described system has been integrated into the data acquisition system that is used to validate muon chambers for the CMS detector at the Legnaro production center and at the CERN storage area. A similar system is also used for the chamber tests with muon beams in the H2 area and at the Gamma Irradiation Facility at CERN.

12.6.2 RCMS Demonstrators

Two demonstrators have been set up to prove the scalability of the RCMS architecture and the technologies it uses.

The first demonstrator uses the basic elements of the prototype RCMS system developed for small DAQ systems, to explore the ability to command a set of DAQ nodes running XDAQ executives. Both flat and

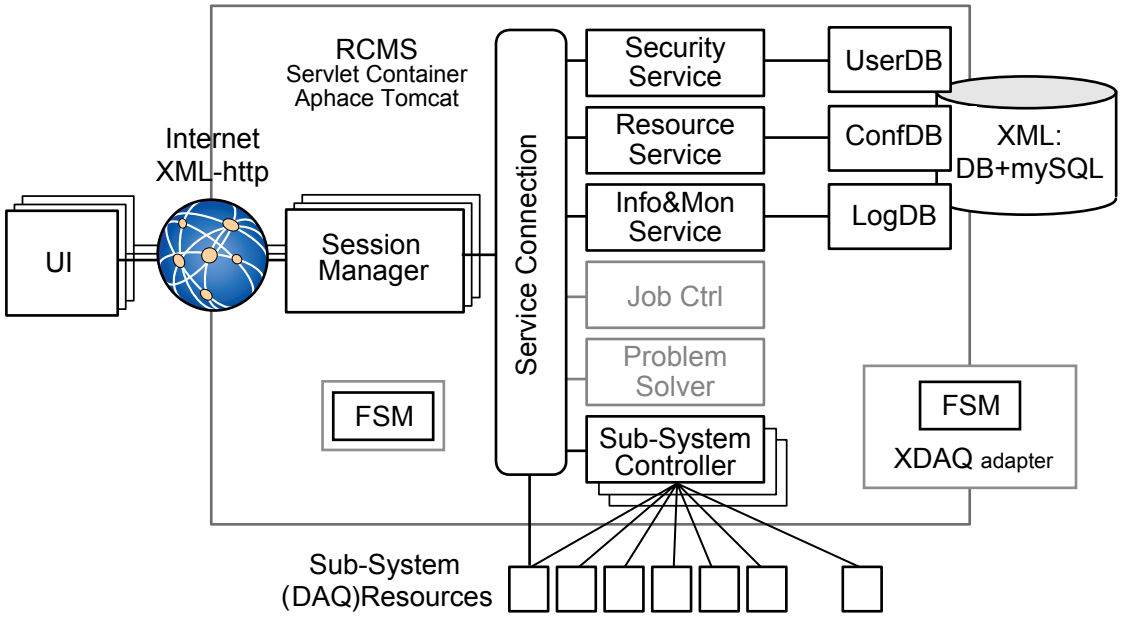


Figure 12-11 Block diagram of the RCMS prototype.

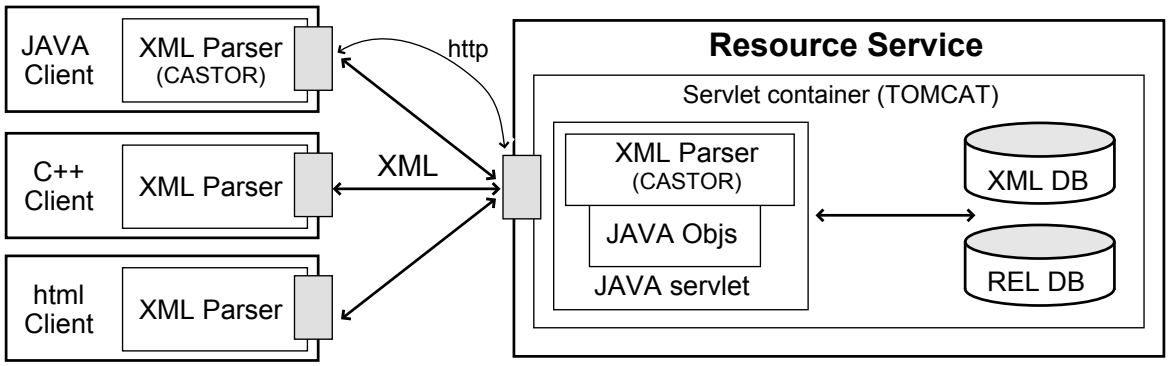


Figure 12-12 Block diagram of the Resource Service Prototype.

hierarchical network topologies are used to prove that the RCMS fits the requirements of a distributed system with the size foreseen for CMS. A 120 node PC cluster, corresponding to the size of a full readout unit builder, has been set up with a single RCMS host. All PCs are 1 GHz Pentium-III dual-processor machines. The XDAQ executives perform dummy actions, merely implementing a state change acknowledge when a command is received. The time measured in the tests therefore represents the time required to perform a state change of the entire cluster. Figure 12-13 shows the measured time as a function of the number of nodes addressed. Line A represents the behaviour when the function manager works sequentially, sending a command and waiting for an acknowledgement before sending the next command to another node. Line B shows a multi-threaded implementation where multiple nodes are sent commands concurrently. In this case a command can be sent to all nodes within 150 ms. Further optimization can be performed through a hierarchical configuration as shown in Figure 12-14. Intermediate nodes with FM functionality are controlled by the master node, to relay the commands to the leaf nodes. Line C in Figure 12-13 shows the performance of this approach (as in Figure 12-14) with two intermediate nodes, each one controlling 64 leaf nodes.

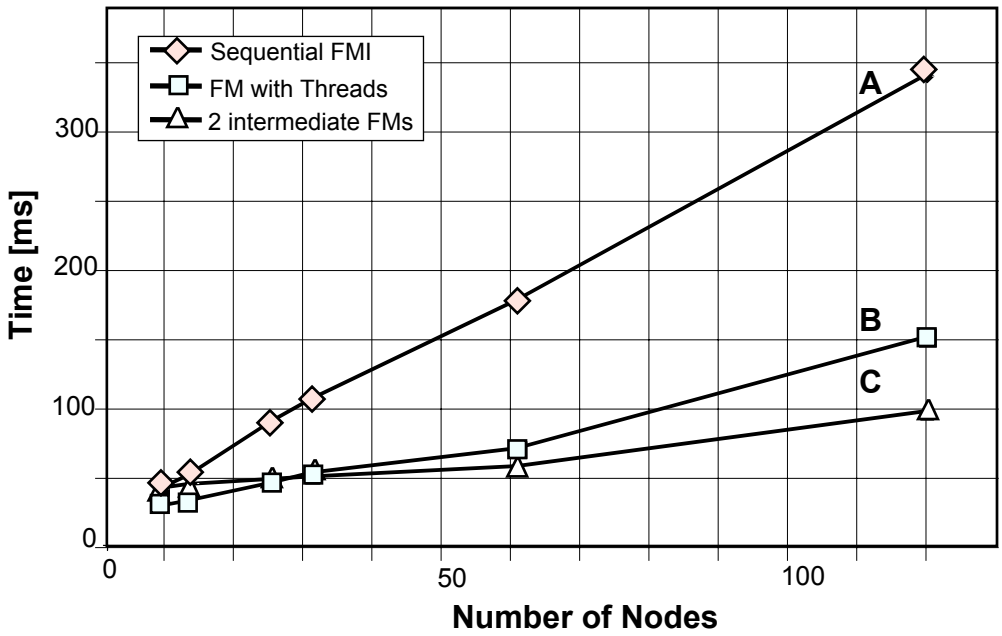


Figure 12-13 Time spent to send a command to the PC cluster vs. the number of nodes of the cluster.

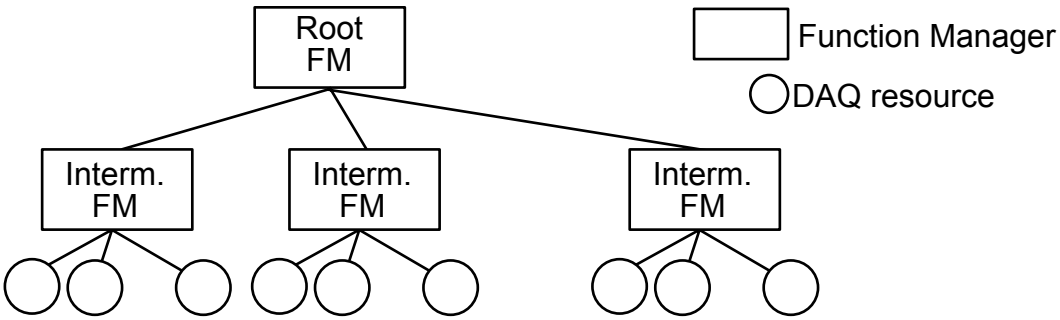


Figure 12-14 PC nodes organized in a hierarchical architecture with an intermediate layer.

The second demonstrator is a simplified version of a log message service based on Web Service technology. The PCs used for deployment were from the same cluster as the first demonstrator. Multiple instances of the service hosted by different physical machines have been implemented to show scalability. Every instance exports a logging interface through the Web Service Description Language (WSDL) [12-11]. This information is then published using Universal Description, Discovery and Integration (UDDI) registries [12-12] located in a dedicated host. Clients looking for the log services, query first the UDDI server to discover the location of the service and retrieve the access method. Up to 4 web services and a fixed set of 15 clients have been used. The Web Service infrastructure, including the UDDI registry, has been developed using the Glue platform [12-13]. The performance of the logging service scales linearly with the number of instances of the service available (Figure 12-15).

The second demonstrator shows the following advantages of Web Service technologies:

- rapid application deployment
- run-time configurability and maintenance through the addition or the removal of services via UDDI

- standardization of service descriptions and access

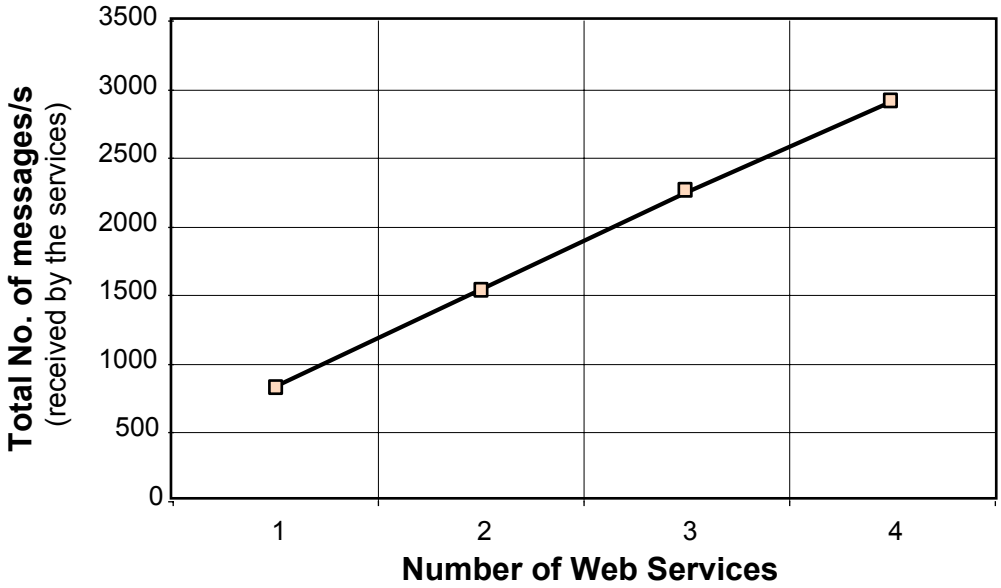


Figure 12-15 Performance of the Log Service in function of the number of servers are providing the service

12.7 Summary and Outlook

In this chapter the design and architecture of the RCMS have been presented. A service driven architecture has been adopted due to its intrinsic modularity and scalability. This approach leads to a design which can exploit the modern concept of world-wide interoperable services found in existing Web technologies.

Prototypes and demonstrators have been developed using state-of-the-art technologies to provide a proof of principle of the proposed approach. However, some products such as XML databases and Web Service platforms are very recent and require further investigation into their stability and robustness.

The prototypes developed did not attempt to test the scalability of the database technologies used. However, different approaches to cope with scalability requirements exist and have to be validated in the scope of the RCMS system.

Distributed architectures, like the one proposed, are intrinsically weak in terms of security. This requires further investigation, for example the tracking of other projects (e.g. LCG) based on large distributed systems.

Recent trends in Grid technology like the Open Grid Service Architecture (OGSA) [12-14] demonstrate the interest in extending distributed computing to include Web services and related technologies. The adoption of OGSA could have positive effects on the design, deployment and maintenance of the system. Profit can be made from synergies with other OGSA based projects. Thus the present OGSA implementation is currently being evaluated for the RCMS.

Some of the RCMS functionalities have been tested in the context of small DAQ systems used for test beam setups and detector validation. This activity will continue in the future and will be extended to test-benches for the trigger and electronic Front-ends. At present, the setups for the drift tube muon cham-

bers are driven by the RCMS, while the integration work is in progress both for the tracker and DAQ-column.

The presented prototypes show that a system based on distributed services that are interconnected by open and standard protocols and have the ability to export their service interfaces, has the modularity and scalability required to configure, control and monitor a CMS DAQ segment (128 nodes). This result is encouraging because modular systems, in terms of FED and RU Builders, map well to the scale of the actual system to be implemented.

The transition from the prototype phase of the RCMS to the final implementation, which requires focusing on software robustness and adherence to production quality rules, will be an important effort in the following years.

12.8 References

12-1 RCMS API documentation can be found at http://RCMS.lnl.infn.it/doc/index.html

12-2 I. Foster et al., “A Security Architecture for Computational Grids”, in Proceedings of the 5th ACM Conference om Computer and Communications Security, 1998

12-3 The LHC Computing Grid (LCG) Project, http://lcg.web.cern.ch/LCG/

12-4 Blaze Advisor,http://www.blazesoft.com

12-5 CLIPS, A tool to build expert systems, http://www.ghg.net/clips/CLIPS.html

12-6 Jess, The rule engine for Java platform, http://herzberg.ca.sandia.gov/jess/

12-7 L. Berti et al.,” Run Control and Monitor System for the CMS Experiment”, LNL-INFN(Rep), 2002, to be published.

12-8 The Jakarta Project: Apache Tomcat, http://jakarta.apache.org/tomcat/

12-9 XML:DB database, see http://www.xmldb.org/

12-10 The Castor Project, http://castor.exolab.org/

12-11 Web Service Definition Language, http://www.w3.org/TR/wsdl

12-12 Universal Description, Discovery and Integration (UDDI) project, http://www.uddi.org/

12-13 GLUE, a Java Web Service Platform, http://www.themindelectric.com/

12-14 I. Foster et al., ”The Physiology of the GRID”, http://www.globus.org/research/papers/ogsa.pdf

13 Detector Control System

The main aim of the Detector Control System (DCS) is to ensure the correct operation of the CMS experiment, so that the data taken with the apparatus is of high quality. The scope of the Detector Control System (DCS) is therefore very wide and includes all subsystems and other individual elements involved in the control and monitor of the detector, its active elements, the electronics on and off the detector, the experimental hall as well as communications with the accelerator. The DCS also plays a major role in the protection of the experiment from any adverse events. Many of the functions provided by DCS are needed at all times, and as a result selected parts of the DCS must function continually on a 24-hour basis during the entire year.

13.1 Introduction

The primary function of the DCS [13-1] will be the overall control of the detector status and its environment. In addition DCS has to communicate with external entities, in particular the RCMS and the accelerator.The integration of the DCS system into the overall online system is illustrated in Figure 13-1. The

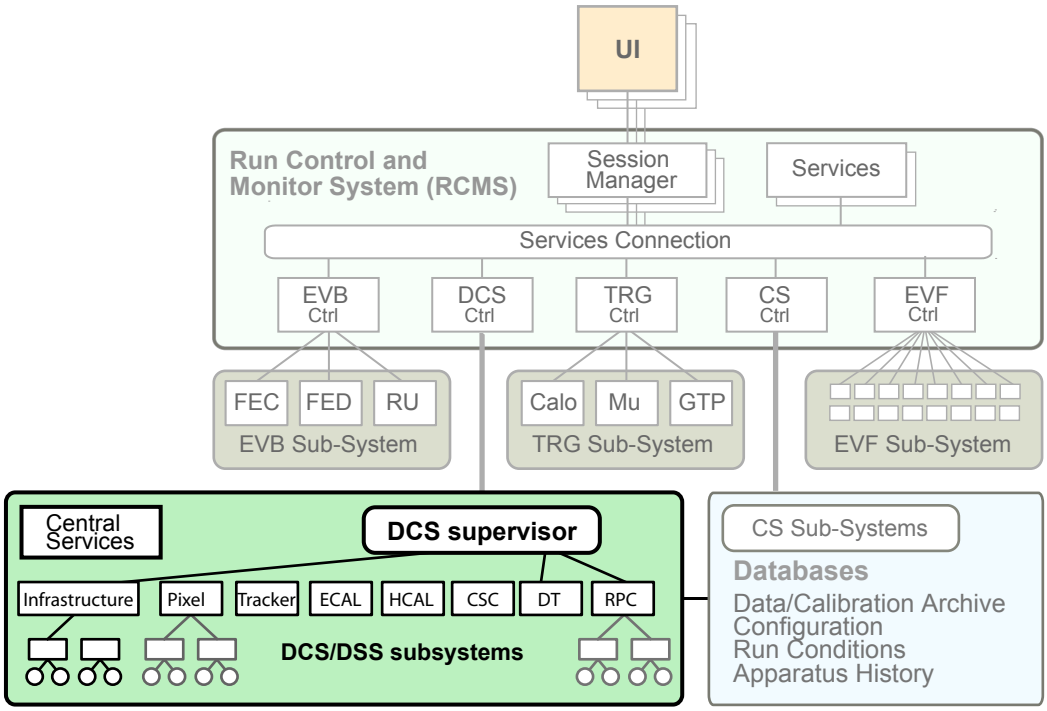


Figure 13-1 Integration of DCS in the online system.

DCS will be a slave to the Run Control and Monitor System which will be in charge of the overall control and monitoring of the data-taking process. When the DCS will operate outside data-taking periods it will act as the master of the active detector components and other parts of the system. The major tasks in this set will be:

- central supervision of all DCS elements,

- communication with external systems (RCMS, accelerator),
- user interfaces, and
- bookkeeping of detector parameters and used DCS hardware/software components.

An integral aspect of the DCS is its provisions for three important safety-related functions, namely

- Detector security (DCS/DSS)
- Alarm handling (DCS)
- Software access control (DCS)

The highest level of safety, “Level 3”, is established for the most severe situations where a human life may be at threat. This critical level will be handled by the CERN Safety Alarm Monitoring (CSAM) [13-2]. Less severe alarms, referred to as “Level 1” and “Level 2”, will be handled either by the Detector Safety System (DSS) [13-3] or by the DCS system itself. The exact boundary between the two will be defined in the future.

Another major task of the DCS is the control and monitoring of the systems regulating the environment at and near the experiment, a set of tasks traditionally referred to as “Slow Controls”, and which include, among others:

- Handling the supply of electricity[1] to the detector,
- Other DCS-related detector electronics (e.g. calibration systems),
- Control of the cooling facilities and the overall environment near the CMS subdetectors,
- Supervision of all gas and fluids subsystems,
- Control of all racks and crates

13.2 Requirements

This section discusses the main requirements on the DCS. It is not meant to be exhaustive, in the sense that it is sufficient to obtain an engineering blueprint for the system. Rather, the discussion highlights some of the major challenges that the system has to address, leaving more standard requirements, e.g. like the programmability of the system, to the actual implementation, under the assumption that such secondary requirements are well understood through work on previous High Energy Physics experiments.

13.2.1 General System Requirements

The major system-wide requirements on the CMS DCS are:

- Most importantly, the system has to be reliable: this requires safe power, redundancy and reliable hardware in numerous places.
- The system must be modular and hierarchical.

1. This task is usually referred to as “High and Low Voltage Control”.

- It has to be partitionable in order to allow independent control of individual subdetectors or parts of subdetectors. Conversely, integration of different parts into the global system must be possible with ease.
- The system needs automation features in order to speed up the execution of commonly executed actions and also to avoid mostly human mistakes in such repetitive actions.
- The system has to be easy to operate: a few non experts should be able to control the routine operations of the experiment.
- The system must be scalable: it must be possible to integrate new subdetectors or subdetector parts.
- The system must provide generic interfaces to other systems, e.g. the accelerator, the CMS Run Control and Monitor system.
- The system must be easy to maintain. This requirement strongly favours the usage of commercial hardware and software components.
- The system has to be homogeneous, which will greatly facilitate its integration, maintenance and possible upgrades, and has to have a uniform “look and feel” throughout all of its parts.
- The system has to provide a possibility for remote control via the internet to allow non CERN resident detector groups to do maintenance and control from their home station.

Beyond these general system-wide requirements there are a few additional ones resulting from the specific design and implementation of the various CMS subdetectors. These requirements are the subject of the following section.

13.2.2 Subdetector Requirements

The Detector Control Systems of individual subdetectors will be connected to the central DCS supervisor hierarchy. These systems will handle all the individual detector electronics which are only partially based on standard components like, for instance, the CAEN high-voltage power supplies. The low-voltage system as well as the gas system will be common for all sub-detectors, whereas the cooling systems will built individually by each sub-detector. For the front-end control links, the major part of the sub detectors has adopted the solution developed for the CMS tracker (CCU). Figure 13-2 illustrates the current situation for the front-end control links. A summary of the DCS items to be controlled in each sub detector is given in Table 13-1. Some particularities of the different sub-detectors are sketched in the following paragraphs.

13.2.2.1 Magnet

The magnet will be an independent system communicating with the central DCS supervisor. The control of this system will be carried out by a dedicated group responsible for the magnet as a whole. The central DCS will only receive status information from the magnet which will thus appear as a sub-detector which does not accept commands but only provides status information.

13.2.2.2 Tracker

Stringent requirements in the tracker system arise from the required cooling. To maintain the tracker within its working temperature, a reliable controls system built of Programmable Logic Controllers (PLC) will be installed. Although this system is part of the DCS, it will work independently and will only report its

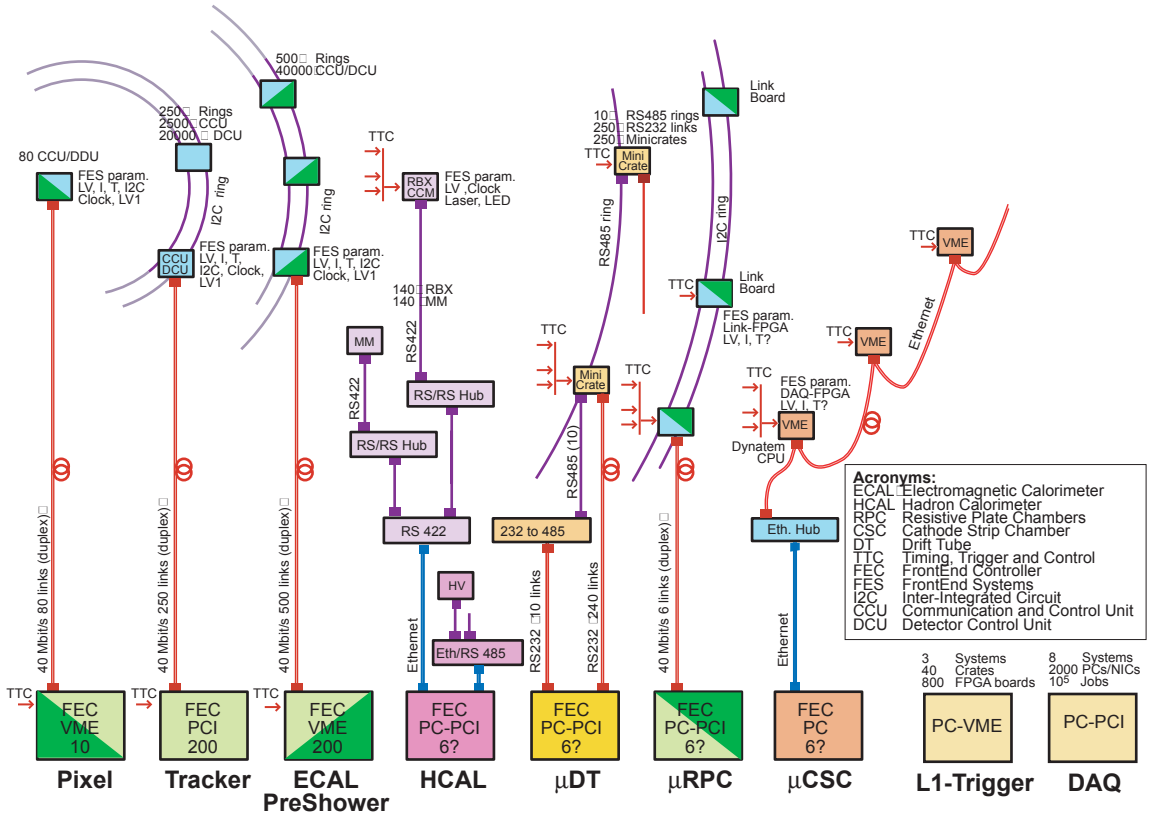


Figure 13-2 Implementation of the control links for the front-end electronics.

Table 13-1 Summary of DCS tasks, broken down into those that are common across sub-systems and those that are specific to each sub-detector.

Detector	Common	Specific
Pixel tracker	HV, LV, gas, FE links (CCU)	Cooling, temperature monitoring
Strip tracker	HV, LV, gas, FE links (CCU)	Cooling, temperature monitoring, thermal screen, alignment
ECAL	HV, LV, gas, FE links (CCU)	Cooling, temperature monitoring, laser
HCAL	LV	HV, radioactive source, laser, LEDs, FE links, cooling, temperature monitoring
DTs	HV, LV, gas, cooling, temperature monitoring	Alignment, FE links, cooling
CSCs	HV, LV, gas	FE links, alignment
RPCs	HV, LV, gas, FE links (CCU), cooling, temperature monitoring	

status to the DCS system. It will be possible for this to take direct emergency actions (e.g. switch off the power supplies) hard-wired connections.

An important element of the tracker control system is the link to the front-end electronics control [13-4], which uses a network similar to the IBM token-ring. The Front-End Controller (FEC) which will be housed in a VME crate in the counting room is the master on the ring and the Communication and Control

Units (CCU) in the front-end electronics on the detector are the slaves. With its two redundant network lines and 40Mb/s transmission speed, the link offers a fail-safe solution with high throughput. Due to those advantages and its open architecture it will be used, after slight adaptations, by other detectors as well.

13.2.2.3 Electromagnetic Calorimeter (ECAL)

The ECAL requirements on the cooling system are also very stringent: its temperature has to be kept stable within 0.1º. For this reason, the ECAL will have three sets of temperature probes. Many thousands of temperature probes will provide a fine-grained temperature map which will be used for corrections to the data. At present, it is foreseen to transmit these temperature readings via the normal data acquisition path. Coarse-grained temperature information will be sent via the control links and will be used for the supervision of the cooling system. A third group of temperature probes will be used for automatic, hard-wired action in the event of a failure in the cooling system. The links for the front-end electronics control will be the same as the ones for the tracker.

13.2.2.4 Hadronic Calorimeter (HCAL)

The amount of data relevant to the control of the HCAL is relatively small. However, a system for passing a radioactive source through the detector will have to be controlled as well as a laser system for calibration.

13.2.2.5 Muon Systems (MUON)

The alignment system will be part of the Muon system (see Section 13.5.4 for further details). Pieces of the barrel alignment system will be mounted on the barrel muon chambers and the alignment control will use the controls links of the barrel muon system. Those links will therefore handle not only the data of the two systems, but also the ownership of the controls of the two systems.

13.3 Architecture and Framework

A commercial SCADA (Supervisory Controls And Data Acquisition) system PVSS II [13-5] has been chosen by all LHC experiments as the supervisory system of the corresponding DCS systems. PVSS II is a development environment for a SCADA system which offers many of the basic functionalities needed to fulfill the tasks mentioned above. PVSS II components for use by all LHC experiments are either already developed and maintained, or are currently under development centrally within the framework of the CERN-wide “Joint COntrols Project”, JCOP. The framework provides the tools to build a partitionable controls hierarchy as well as the PVSS II implementation of certain hardware devices.

13.3.1 Command Hierarchy

Detector controls will be organized in a hierarchy as shown schematically in Figure 13-3. The implementation of this organisation will be made purely in software and only represents the logical structure of the detector. A prototype of the hierarchy mechanism is already implemented in the JCOP framework. The interfaces of all nodes in this hierarchy are made such that each node can be connected to any other node.

Therefore each node can be placed anywhere in the hierarchy. This allows for independent development of the sub-detector systems while it will greatly facilitate their integration into the full system.

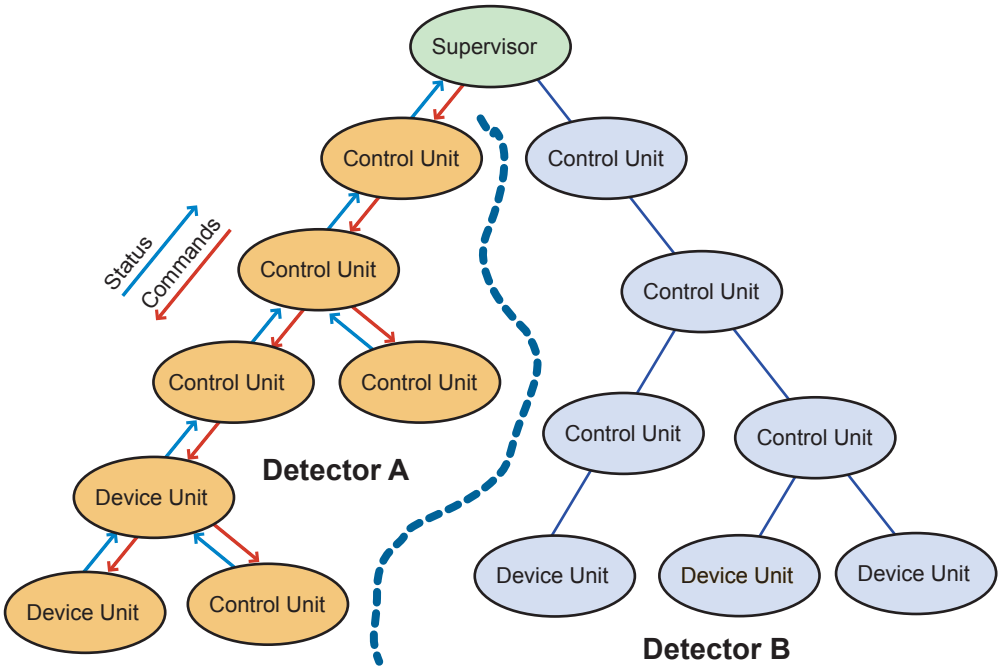


Figure 13-3 Illustration of command hierarchies in the DCS. Commands flow downwards, status flows upwards.

The DCS supervisor, at the topmost point of the hierarchy, offers global commands like “start” and “stop” for the entire detector. The commands are propagated towards the lower levels of the hierarchy, where the different levels interpret the commands received and translate them into the corresponding commands specific to the system they represent. As an example, a global “start” command is translated into a “HV ramp up” command for a sub detector. Correspondingly, the states of the lower levels are summarised by the upper levels and thus define the state of the upper levels. As an example, the state “HV on” of a sub-detector is summarized as “running” in the global state. The propagation of commands ends at the lowest level at the “devices” which are representations of the actual hardware. The SCADA system ensures that every action on the devices is transmitted to the hardware and that the state of the devices represent the hardware status. Several possible ways for carrying this out are described in Section 13.4.

13.3.2 Partitioning

The ability to partition the command hierarchy is essential for a detector like CMS which comprises a large number of subdetector elements. Partitioning implies that a sub tree is cut off from the command hierarchy. In this way components can be operated independently from the rest of the tree thus rendering the operation of the corresponding subdetectors independent of the rest of the system. This mode of operation will be used mainly for maintenance, calibration, system testing and trouble shooting. Under normal operating conditions, i.e. during a “physics run”, there will be only one partition, comprising all subdetectors participating in the data-taking.

The mechanism of partitioning is illustrated in Figure 13-4. Partitioning is introduced into the command hierarchy by declaring a node as partitionable. This adds three parameters to each node that provide the

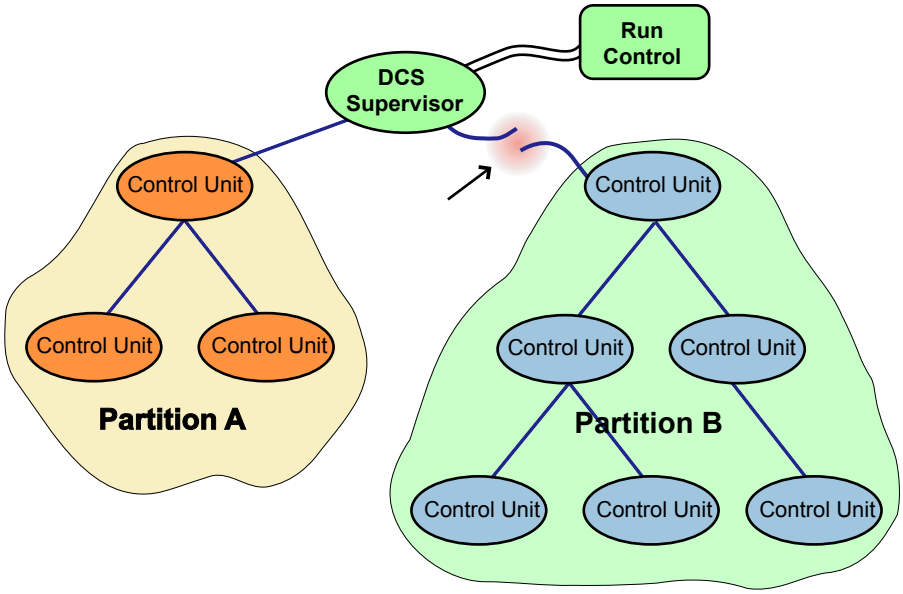


Figure 13-4 Illustration of the mechanism for partitioning the DCS.

- actual owner of node
- possibility to ignore states of lower level nodes
- possibility to ignore commands of upper level nodes

Tools to change the status of all three properties exist in the JCOP framework. The tools ensure that all nodes in a partition are owned by the same user and that the control of nodes in a partition can only be exercised through the partition's root node. The root node is the top most element in a tree or a branch and has therefore only children and no parents. For instance in Figure 13-4 the root node of detector B is the top most control unit in detector B after partitioning it from the DCS supervisor. However the owner can decide to share a partition and so a different user can also send commands to it.

13.3.3 Alarm Design Issues

A "Level1" or "Level 2" alarm indicates either an error condition or the occurrence of an undesired state of the detector or the other elements monitored by the DCS. Most of these alarms will be handled by the DCS using the alarm-handling mechanism of PVSS II. An alarm is created when a parameter value leaves a definable value range and it is cleared when the value re-enters this range. Alarms will be distributed to all entities, be they computing systems or humans, that are in charge of the system in question. An alarm has several properties:

- source
- possibility/necessity of an acknowledgement by the operator as well as a specification of the delay of this acknowledgement
- severity level
- time of occurrence
- system state that led to the alarm

- additional details like a help text describing the alarm and a potential reaction or even cure

Alarms will be logged persistently in the PVSS II database. PVSS II tools, which include a graphical display for active and archived alarms, can be used to browse this database.

13.3.4 Software Access Control

CMS will use communication techniques based on the Internet to offer remote maintenance and surveillance of the detector cavern and all its counting rooms. This implies a security risk. An access control mechanism must be put in place in order to prevent any damage or other disturbance caused by such incidents. The DCS system will utilize standard encryption and authentication techniques. On the other hand access control will be needed to allow for certain actions depending on the expertise of the user. For instance an experimenter on shift might only be allowed to switch a detector on or off, while an expert might be allowed to also change various key voltage settings.

PVSS II offers an intrinsic access mechanism where actions can be restricted on a per-user basis. However, more sophisticated access mechanisms may eventually be required and therefore, in the scope of the JCOP framework, an extension to the current access control system is planned. A future version of PVSS II, which will provide a simple, web-like remote access is expected to also include access control, encryption and authentication.

13.3.5 Configuration

Many DCS front-end components and their power supplies will have to be configured. This will be needed either to bring them into a running condition or to simply define their running mode. These configurations will be stored in the DCS configurations database outside the PVSS II system. The configuration data for environmental setup will be read and applied by the PVSS II system directly.

Another issue is the configuration of the DCS subsystems themselves. In the context of the JCOP project, mechanisms for automatic configuration of PVSS II systems are currently being developed. A possible future extension would be a mechanism to take configurations from a database and automatically apply them.

13.4 Hardware and Software Components

13.4.1 SCADA

Systems used for the supervision level of industrial controls applications are called SCADA (Supervisory Controls And Data Acquisition) systems. The selection of a single commercial standard framework for the controls of all the sub-detectors in the experiment allows for the development of a homogeneous system. A typical SCADA system is based on the experience of many people in this field, of perhaps many hundreds of man-years, in the development and debugging before becoming a stable and mature system. Support and maintenance, including documentation and training, is provided by the company.

The four LHC experiments have chosen PVSS II as the SCADA system for the control of the experiments. This system is essentially a toolkit that delivers the building blocks of a control system. It will be the basic skeleton of the DCS by housing many central components like:

- Database
- Network interconnection
- Graphical user interface
- Hardware connectivity

Every PVSS II system can consist of all these components. Several PVSS II systems can be connected together to enhance the capabilities and offered resources and to distribute the load. A system can be spread over several PCs or more than one system can be run on one PC. This design makes PVSS II highly scalable and allows independent development of sub-parts of the DCS. PVSS II offers many features, which are needed for a control system, e.g.:

- Access control
- Alarm handling
- Logging
- Archiving
- Trending (i.e. plotting PVSS II data versus time or versus other PVSS II data)
- C-like scripting
- API (Application Programmable Interface)
- Redundancy manager
- HMI (Human Machine Interface)
- Networking
- Development tools

Each PVSS II system has always one event manager and one database manager. It can have multiple control managers (for the scripts), API managers (named later API), drivers (connected to the hardware) and user interface managers. Each of these managers can run on a separate processor if required. The mix of ready-to-use functional components and easy and open programming capabilities simplify the development and offer the flexibility required. The PVSS II system runs on Windows and Linux and one can mix both operating systems even within a single PVSS II system, as long as no external component used, like OPC, restricts to one.

13.4.2 OPC

OPC (OLE for Process Control, where OLE stands for “Object Linking and Embedding”) defines a set of interfaces, based on OLE/COM and DCOM technology [13-6], for truly open software application inter-operability between automation/control applications, fieldbus/device and business/office applications.

OPC is managed by the OPC Foundation [13-7], which is comprised of more than 220 companies and institutes as members. The foundation has defined and released five sets of interfaces:

- OPC Data Access
- OPC Alarms and Events
- OPC Batch Interface

- OPC Historical Data
- OPC Security

The most interesting interface, for the applications in the DCS system, is the OPC Data Access. This interface prevents the need for specific drivers to connect to commercial hardware. Essentially all industrial hardware (e.g. PLCs) are marketed with an interface of this type. The software that provides this interface is called an “OPC server”. Control software, like a SCADA system, are “OPC clients” and can be easily connected to an OPC server. OPC, which is only available on the Windows platform, is currently the JCOP recommended way to connect commercial controls hardware to the software level. OPC servers for custom hardware can also be easily developed using dedicated toolkits.

13.4.3 Databases

13.4.3.1 PVSS II Internal Database

PVSS II includes its own proprietary database which is used to store the values, along with their history, of all the parameters which define the current state of the system (e.g. HV settings, alarms, etc.). The configuration of PVSS II itself is also stored in this database. A tool to back up the value history is included. The database is optimized for fast access, a feature that is required because these data are stored in real time. On the other hand it is not designed for static data and should not be considered as a database manager server.

13.4.3.2 External Database

For static and large amounts of data an external database is more convenient than the PVSS internal one. These data sources can be accessed via XDAQ (see Section 13.6.1) or by standard PVSS primitives. PVSS offers a connection from the SCADA GUI and control scripting language to relational database management systems through the ODBC standard. This connectivity is, however, limited to the Windows platform.

Any database can be accessed from a custom C++ library that can be called from the PVSS scripts and the GUIs. SCADA applications, written in C++ and using the SCADA API, can also access any database using the database libraries.

External Database access will be used mainly to store configuration data and to export data as, for example, in the case of the Conditions Database.

13.4.3.3 Conditions Database

The conditions database will hold the data describing the detector environment at data taking necessary for the offline reconstruction. This will allow for fine tuning the physics event reconstruction as well as for getting the status of all detector components at any point in time. Since the DCS system will control the environmental setup of the detector (HVs, LVs, GAS,...), selected data from the DCS will have to be written into the conditions database. It is foreseen to provide two write mechanisms:

1. on change: a parameter is written whenever it changes.
2. periodically: A parameter is written periodically after a certain time interval.

Whether a parameter has to be written and by which mechanism will be a configurable property of the parameter. To ensure the synchronization between the parameters in the conditions database and the appropriate ones in PVSS II, it will be possible to force an update of the parameters in the conditions database beyond the mechanisms described above.

13.4.4 PLCs

PLCs are simple and reliable front-end computers, which are widely used in industry for controls applications. They support several specific standardised programming languages. Normally PLCs are connected to devices via fieldbuses. They can also be interconnected with each other either via fieldbuses or Ethernet. The common way to connect SCADAs to PLCs is as well via a fieldbus or Ethernet. CERN recommends to use PLCs from two companies, for which support is provided. Simpler and less expensive PLCs (mini-PLCs) have recently become commercially available.

13.4.5 Sensors and Actuators

Sensors and actuators will be standardized as much as possible:

- Temperature measurement
- Humidity measurement
- Gas control equipment such as valves and gas mixers
- Radiation measurement
- Strain gauges
- Others

Note that these devices have to work mostly in a hostile environment with radiation and a magnetic field.

13.5 Applications

In the following, some standard full vertical slices of auxiliary systems which will be used within different sub-detectors are described. A full vertical slice covers all elements starting from the sensor/actuator level, which is connected to devices, and ending at the supervision level.

13.5.1 Power Supplies

CAEN has developed an OPC server for their power supplies [13-8], which will be used by the major part of the sub detectors. The DCS will then communicate with the CAEN systems using its built-in OPC client. The corresponding implementation based on PVSS II is delivered within the JCOP framework. Home made power supplies like in the case of HCAL will be integrated into the framework similar to the CAEN ones, but using home made software instead of OPC.

13.5.2 Gas and Cooling Control

The control of all CMS gas systems covering hardware and software will be designed, installed and maintained centrally by the CERN gas working group [13-9]. A PVSS II system for the experiments gas control will be provided by them with specific user interfaces for all the sub detectors using gas.

The control and regulation of all CMS cooling systems will be covered in an analogous fashion to the gas systems by the CERN cooling and ventilation group. To this end the JCOV (Joint COoling and Ventilation) project [13-10] has been created. In first design studies, the only difference with the design of the gas control system is be that no fieldbus between the PLCs and sensors/actuators is used. The sensors and actuators will be connected via I/O modules embedded in the PLC frame.

13.5.3 Rack and Crate Control

The control of the racks will be done twofold. The control of the main power distribution to the racks will be done using the equipment of the global power infrastructure. A small control unit housed in the turbine case will measure the air flow and temperatures in the racks and do smoke detection. It will report those values to the DCS system via a fieldbus. In case of emergency (i.e. overheating or fire) the power to the rack will be cut via a hardwired safety line to the main power distribution.

Crate control in CMS is foreseen to be done through the same type of fieldbus network used for the racks that house the crates. In this way, the same interface between the crate controller and PVSS II can be used for the racks. Remote control of the fan unit and the power supply has to be provided.

13.5.4 Alignment Control

The different alignment systems of the CMS detector have to steer lasers and sensors. These tasks will be carried out by a custom software layer. The expected number of resulting coordinates, approximately 10000, calculated by alignment software, can be stored in the PVSS II internal database. Therefore the features of PVSS II for data analysis and data presentation, alarming, access control and filtering of unchanged data can be used. As with all other data that are taken by the DCS and are required for the reconstruction of events, alignment constants will be transferred upon the detection of a significant change in the conditions database.

13.6 Connections to External Systems

13.6.1 Interface to the DAQ System

A communication mechanism between the DAQ and the DCS system is needed for several purposes. During normal physics data-taking the DCS will act as a slave to RCMS and will therefore have to receive commands and send back status information. Partitions will also have to be synchronized between the DAQ and the DCS system: the RCMS will instruct the DCS system to set up specific partitions which will be put under the control of the RCMS. Furthermore alarms will be sent from one system to the other.

Following these requirements PVSS interfaces to XDAQ through a native interface (API manager) implemented as a plug in component as described in Chapter 11, "Cross-Platform DAQ Framework (XDAQ)".

Through this interface the DCS system can access various functions of XDAQ, including direct access to PCI and VME systems, database products and run control services. This solution provides access to the whole PVSS II system, so that any kind of data can be exchanged between XDAQ and PVSS II in both directions. This fits all current and future requirements since it uses the most generic interface to PVSS II.

13.6.2 LHC and Technical Services

The LHC Data Interchange Working Group (LDIWG) [13-11] has been created to define a communication mechanism for the exchange of information between independent services at CERN. This mechanism will be used to receive information about the LHC machine and the technical services in the experiments. An interface in PVSS II for this mechanism will be provided by JCOP.

13.7 Summary and Outlook

The requirements for the central and the sub detector DCS have been identified. Although many system parts are similar and can be controlled by commonly developed tools, some of them still require specific software implementations to be developed and delivered by the groups responsible. The commonly used tools will be either industrial solutions or developments by the CERN JCOP framework. Since the beginning of JCOP, three years ago, PVSS II has been chosen as the core SCADA development tool and first implementations of common tools based on this product have started. Sub-detectors have recently started to tackle the implementation issues of their DCS systems, and therefore these systems are just entering their development phases. First prototype systems are expected to exist around summer 2003. Performance and scalability tests on the DCS tools will be carried out in the future. JCOP will evolve from feedback provided by the LHC experiments. A working central DCS system is expected for spring 2005. The Detector Control Systems of individual sub-detectors are expected within a similar time frame.

13.8 References

13-1 CMS Detector Control System (DCS), see http://itcowww.cern.ch/ITCO/DCS-CMS/.

13-2 CERN Safety Alarm Monitoring (CSAM), see http://st-div.web.cern.ch/st-div/Groups/ma/se/CSAM/CSAM.htm

13-3 Detector Safety System (DSS), see http://itco.web.cern.ch/itco/Projects-Services/JCOP/SubProjects/DSS/welcome.html

13-4 Tracker front-end electronics control, see http://cmstrackercontrol.web.cern.ch/CMSTrackerControl/

13-5 PVSS II system from ETM, see http://www.pvss.com

13-6 COM/DCOM information, see http://www.microsoft.com/com/default.asp

13-7 OLE for Process Control (OPC), see http://www.opcfoundation.org/

13-8 CAEN Corp., see http://www.caen.it

13-9 LHC gas working group: http://lhcexp.web.cern.ch/LHCExp/GasWG/gwg_main.htm

13-10 Joint cooling and ventilation project (JCOV), see http://proj-jcov.web.cern.ch/proj-JCOV/

13-11 LHC data interchange working group (LDIWG), see http://vac.home.cern.ch/Controls/WG/LDIWG/

Part 4
High-Level Trigger

14 Detector Simulation and Reconstruction

This chapter describes the software used to generate Monte Carlo events, to simulate the passage of particles through, and their interactions with, the CMS detector, and to simulate the response of the detector elements. It also describes the first steps in the reconstruction of the recorded information. The main goal is to present those details which relate to the accuracy and realism of the input to the HLT selection which is described in Chapter 15, "Physics Object Selection and HLT Performance".

The chapter is organized as follows: the first section gives an overview of event generation, together with details specific to some samples. The CMS detector configuration, and the simulation of the passage of particles through, and their interactions with, the detector elements is described in Section 14.2. Many pp interactions (the number depending on luminosity) are used to simulate the response of the detector elements; these interactions can be both from the same bunch-crossing as the triggering interaction or from previous or subsequent bunch-crossings. This is described in Section 14.3, along with a description of local detector reconstruction. The chapter concludes with a description of the reconstruction of the tracks and vertices of charged particles.

14.1 Monte Carlo Event Generation

14.1.1 Event Generation (CMKIN)

Standard Monte Carlo generators, such as PYTHIA [14-2] and ISAJET [14-3], are used to simulate the collisions between two protons at a centre-of-mass energy of 14 TeV. The events produced are stored in the standard HEPEVT structure in HBOOK ntuples with an average event size of 50 kB. The interface to the user, the interface to the Monte Carlo generator, and the input/output software is steered by the CMKIN program. It is written in FORTRAN as is the detector simulation program, CMSIM. These are the only software packages in the analysis chain that are still based on FORTRAN code. In the near future they will be replaced by C++ equivalents.

14.1.2 Generation of Muon Samples

The generation of the minimum-bias background for HLT studies involving muons is treated somewhat differently than for studies without muons since every charged pion or kaon with a momentum above a few GeV/c can potentially decay into a muon, penetrate into the muon system, and be mismeasured as a high-P_T muon because of multiple scattering through the iron yoke. So, in principle, to calculate the trigger rates the full inelastic cross section at the LHC needs to be simulated. In practice, however, the CPU requirement would be prohibitive; so an event-weighting technique is used.

The weighting procedure, described in detail in reference [14-4], assigns a probability to all potential muon parents, such as pions, kaons, *D*- and *B*-mesons, to decay into a muon within the tracking volume of CMS, based on their kinematics. The probability for a minimum-bias event to have *n* muons in the final state (n=0,1,2,...) is computed, and the sum of all probabilities for muon multiplicities satisfying a given selection criterion (e.g., two or more muons inside the detector acceptance for a di-muon background sample) is taken as the event weight. Then one of the possible muon multiplicities is chosen at random, according to their relative contribution to the event weight. After this choice, the final state is randomly chosen among the possible ones in the event leading to the selected multiplicity, and this final state is

forced in the PYTHIA generation. The final result is a sample of weighted events with at least *n* muons (*n*=1,2 depending on the background sample to be generated) in the final state. Only muons with P_T above a predetermined threshold are kept. This weighting procedure neglects any contribution to the trigger rates from hadronic punch-through, but this contribution has been shown to be small in separate studies.

Additionally, the generation of the minimum-bias background is broken up into several samples with differing cutoffs for the minimum transverse momentum of the hard-scattering (P_T^{hard}). In particular, one sample is generated with P_T^{hard} = 0 GeV/c and with minimum muon P_T = 1 GeV/c (from the weighting procedure described above), another sample with P_T^{hard} = 0 GeV/c and with minimum muon P_T = 4 GeV/c, and the last sample with P_T^{hard} = 10 GeV/c and with minimum muon P_T = 10 GeV/c. These samples are combined together with an event weight that depends on the P_T of the hard scattering as well as the muon event weight. Figure 14-1 shows the integrated rate of single muons obtained from PYTHIA as a function of the muon P_T threshold, for a luminosity of 10^{34} $cm^{-2}s^{-1}$. The muons are restricted to the Level-1 muon acceptance of $|\eta| < 2.1$, and the breakdown of the muon sources is also shown. The contribution from minimum-bias events uses the weighting procedure described. Charged kaons and pions contribute as a source of muons as much as bottom and charm quark decays at low P_T, and *W* decays dominate at high P_T. Figure 14-2 shows the integrated rate of di-muons at the generator level, and their source, for a luminosity of 10^{34} $cm^{-2}s^{-1}$. Decays of *Z*-bosons dominate over minimum-bias backgrounds at high P_T.

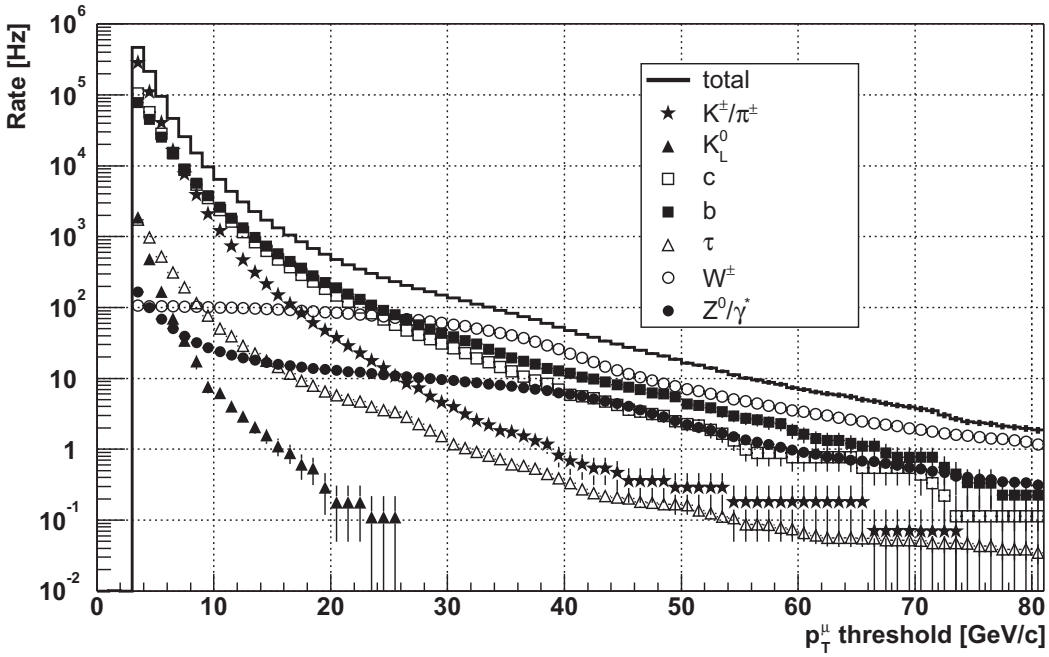


Figure 14-1 Integrated rate of single muons from PYTHIA as a function of the muon P_T threshold and for a luminosity of 10^{34} $cm^{-2}s^{-1}$. The breakdown of the muon sources is also shown.

14.1.2.1 Pile-up

Special care must be taken in simulating the pile-up of minimum-bias collisions within a bunch crossing for the muon HLT studies. A procedure, described in more detail in reference [14-5], is applied to mix the weighted minimum-bias collisions into a sample of beam crossings with the proper distribution of pile-up collisions and muons for a given luminosity. HLT rates for *n* muons from minimum-bias collisions, with

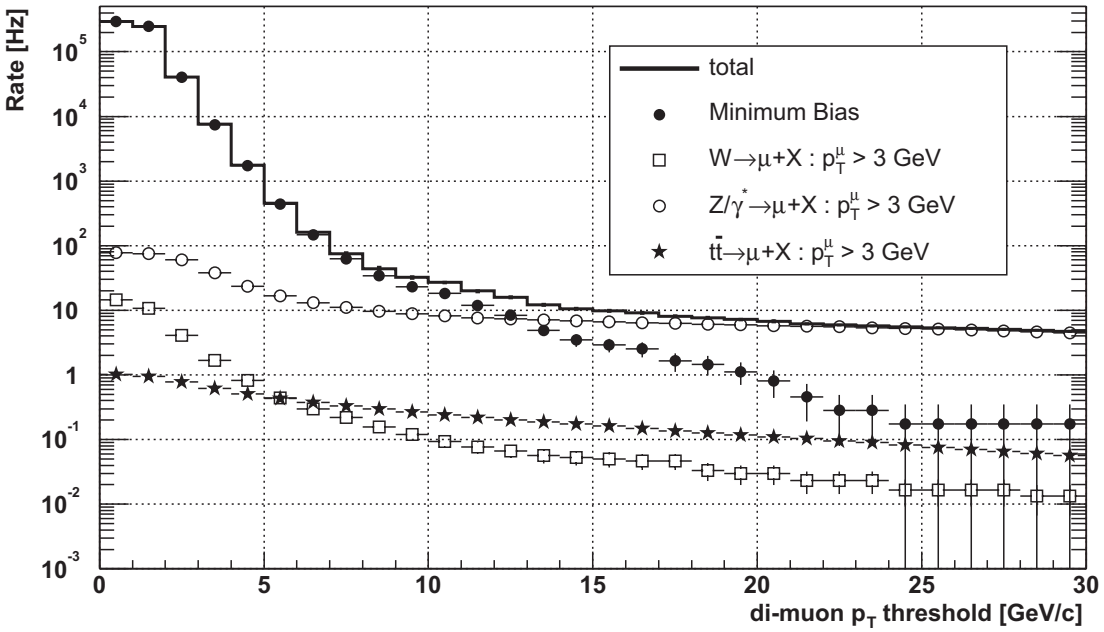


Figure 14-2 Integrated rate of di-muons from PYTHIA as a function of the muon P_T threshold and for a luminosity of 10^{34} $cm^{-2}s^{-1}$. The breakdown of the muon sources is also shown.

all correlations, can thus be obtained. For rate estimates from *W, Z* or top-quark decays, no event weighting was applied, and the pile-up within a bunch crossing is treated normally.

Muon detectors are also sensitive to thermalized neutrons (produced in collisions up to millions of bunch crossings earlier), which, when captured by a nucleus, may yield a gamma-ray that converts into an electron-positron pair close enough to an active gas layer to produce a signal. Although parameterizations of this long-time pile-up exist, they were not enabled for the studies reported here mainly due to the excessive CPU time spent on digitizing the large number of random hits. A dedicated Level-1 Trigger study showed negligible impact of this background on the performance of the Level-1 Track-Finder for the CSC system [14-6]. The neutron background effect is also expected to be negligible on the performance of the Level-1 Track-Finder for the DT system, where the neutron flux is expected to be 10–100 times smaller [14-7]. On the other hand, the output of the Level-1 Pattern Comparator Trigger of the RPC system is sensitive to the neutron background in a way that is dependent on the intrinsic detector noise assumed [14-8], and further dedicated Level-1 Trigger studies are underway.

14.1.3 Generation of Jet Trigger Samples

Because the jet energy resolution of the CMS calorimeter system is modest, and because of the very large amount of hadronic energy in the events due to the large number of pile-up interactions, the backgrounds to jet and missing transverse energy triggers are dominated by jet production. The data samples for study of these triggers thus consist mostly of inclusive jet events.

Inclusive di-jet events were generated with PYTHIA 6.152. In general, PYTHIA tends to underestimate the rates of multi-jets, as multi-jet production is simulated through a showering model of gluon radiation and not from a true matrix-element calculation. The events were generated in 21 bins in P_T^{hard}. A large number of events were generated with P_T^{hard} around 100 GeV since many of the jet triggers require a jet of transverse energy around 50 GeV. A large number of events with low P_T^{hard} were generated as well,

since these events turn out to give a surprisingly large contribution to the E_T^{miss} rate even after moderate E_T^{miss} and jet E_T cuts. Events with large P_T^{hard} were also generated to estimate the constant term in the jet resolution.

In the limit of very low luminosity, when the probability of having more than one interaction per crossing is very small, or when the trigger is on a high P_T object, which again has a very small probability per crossing, it is easy to calculate trigger rates from a background sample of inclusive di-jet events created in several disjoint bins in P_T^{hard}. The rate is just the sum of the rates from the individual bins: $R = L\Sigma\varepsilon_i\sigma_i$ where L is the instantaneous luminosity, ε_i is the efficiency for events in bin i to pass the trigger conditions, and σ_i is the cross section for inclusive di-jets within the P_T^{hard} range for that bin. However, when calculating rates for missing transverse energy triggers, since pile-up events make a non-negligible contribution to the trigger object (the E_T^{miss}), the relative cross sections of the pile-up events must be taken into account as well.

Let N_{bin} be the number of bins in P_T^{hard} and consider a given crossing which has $(n_1, n_2, \ldots, n_{Nbin}) = \vec{n}$ (in time) events from those bins. The mean expected number of interactions from each bin is then $\mu_i = f_i\mu$ where $f_i = \sigma_i/\sigma$. The probability of this distribution of interactions is just the product of the Poisson probabilities, $p(n_i;\mu_i) = \exp(-\mu_i)(\mu_i^{n_i}/n_i!)$, to obtain n_i events in bin i when μ_i events are expected on average. In the Monte Carlo production, N_j crossings in which the first event was required to be in the j^{th} P_T^{hard} bin are actually generated. Given these numbers, the weight assigned to each crossing with a given $\vec{n}$ is given by[1]

$$W(\vec{n}) = 31.578\mu \Big/ \left(\prod_{j=1}^{N_{bin}} N_j \frac{n_j}{f_j} \right) \text{MHz}.$$

The importance of using the proper weights is shown in Figure 14-3. These figures show the trigger rates as a function of E_T^{miss} and the 4-jet threshold with this type of weighting and with the low-luminosity weighting. The rate is overestimated when naive weights, which do not take into account the weights of the pile-up events, are used. A typical Level-1 Trigger has a rate of 1 kHz.

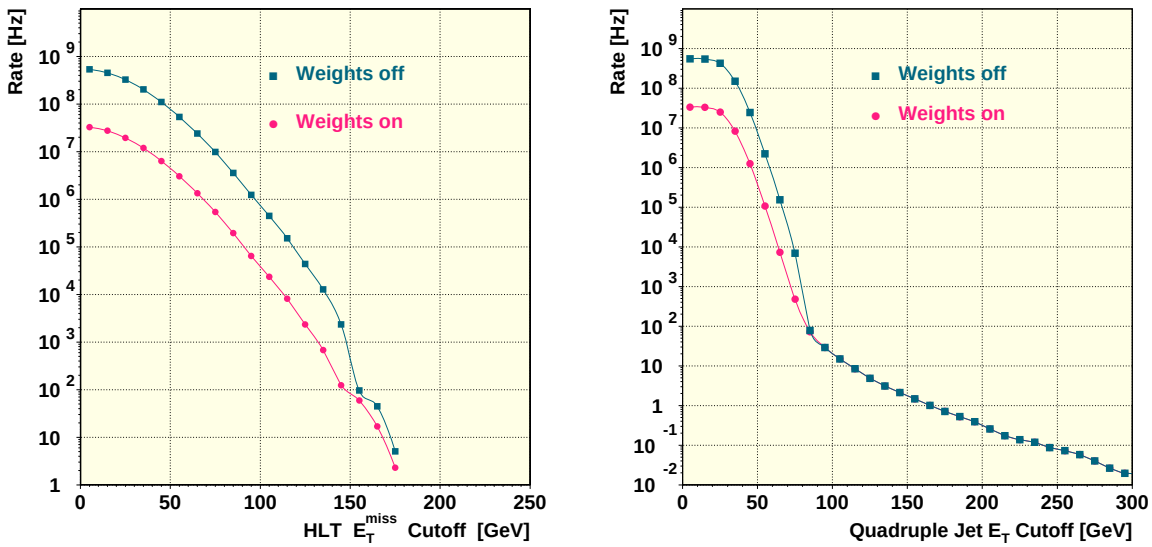


Figure 14-3 Event rates for E_T^{miss} (left) and for 4-jets (right) with and without pile-up cross-section weighting at high luminosity. The kink shows the unphysical results obtained using the incorrect weights.

1. When the gaps in the bunch structure are taken into account there are 31.578×10^6 bunch crossings/s in the LHC.

14.1.4 Generation of Electron Trigger Samples

Jet background events for the electron/photon triggers are generated with PYTHIA in five bins of P_T^{hard}. A preselection is made at generator level to enrich the sample in events which will pass the Level-1 electromagnetic trigger. It has been verified on a large unbiased sample that this preselection does not introduce significant bias.

A complication to the simulation of pile-up for the event sample used for electron and photon studies is caused by minimum-bias events, generated as pile-up, that fire one arm of the Level-1 electron trigger. The events in the pile-up sample are reused many times. Generation of sufficient minimum-bias events to allow their use only once for the simulation of pile-up at high luminosity is not currently feasible, nor would it be an effective use of resources. But the reuse of pile-up events which themselves trigger leads to statistical problems. The problem has been circumvented by excluding such events from the pile-up sample. But this selection excludes a possible source of background. The magnitude of this excluded background contribution has been estimated by calculating the probability, using a large unbiased sample of fully simulated events, that a double electromagnetic trigger results from crossings in which one event fires one arm and another event the other arm of the two electron trigger. For the high luminosity double electron trigger threshold at 14 GeV, the excluded rate is found to be about 5% of the total.

14.2 Detector Description and Simulation (CMSIM)

The CMS simulation package, CMSIM [14-9], is an application of the GEANT3 [14-10] detector description and simulation tool[1]. CMSIM is used to describe the detector geometry and materials. It also includes, and uses, information about the magnetic field.

CMSIM reads the individual events from the CMKIN ntuple and simulates the effects of energy loss, multiple scattering and showering in the detector materials with GEANT3. The information is stored in the form of “hits”. A simulated hit combines information about energy depositions in the detector, their magnitude, the time at which they occur and their location. The information stored in a hit is detector-dependent and contains all the details needed to simulate the detector response.

For tracking detectors (inner tracker and muon system) the information stored is the entry and exit points of the track, the momentum vector at the entry point, the type of the particle, the energy loss in the sensitive volume, and the time offset with respect to the trigger interaction time. For the calorimeters, the hits contain summarized information about the energy (for HF the number of photoelectrons) deposited by a shower in a crystal or a scintillator and the time slice during which that happened. A time slice is an interval of 1 ns. Finally, a number to identify the detector element that contains the hit is stored. Two calorimeter hits are distinct when they originate from different particles — even if this happens at the same time and in the same detector element.

In addition to the hits, CMSIM also produces simulated tracks and vertices. These are not used for detector response simulation or for reconstruction, but are required for analysis of the reconstruction results. They constitute the “Monte Carlo truth” about the interactions and decays of particles in the detector.

The results of the simulation are stored in GEANT3/CMSIM data structures in the form of ZEBRA banks. Interface routines similar to GS- and GF-routines in GEANT3 are provided for users to access data structures without knowledge of the ZEBRA memory management system. The information stored in the ZEBRA files is then imported into ORCA and stored in the Objectivity/DB databases. The average stor-

1. CMSIM also includes code for reconstruction and analysis which is, however, not used any more.

age size of a single event is 2 MB. The CPU time required to simulate one event on a 1 GHz Pentium-III CPU ranges from 60 s for a minimum-bias event, to 500 s for a 1 TeV jet-jet event.

All results presented in this TDR are, unless otherwise specified explicitly, simulated with version CMSIM125 of the CMSIM package.

In what follows, the CMS coordinate system is defined as follows: the z-axis is parallel to the beam axis and the magnetic field of the solenoid, and the polar angle theta is defined with respect to it. The azimuthal angle around the z-axis is denoted by the symbol ϕ, and the pseudorapidity -log(tan(θ/2) is denoted by the symbol η. The symbol P_T is used to denote the component of momentum transverse to the z-axis.

14.2.1 Tracker Geometry and Simulation

In the original design, the CMS Tracking detector consisted of a pixel vertex detector, a silicon Microstrip detector and MSGC gas detectors in the outer layers. In 2000 an addendum to the original design of the CMS tracker was approved. In the new design, the outer MSGC layers were substituted by other layers of silicon microstrips leading to an all-silicon configuration. The layout of the all-silicon tracker was subsequently optimized, before finally being frozen in the fall of 2001. It is this final layout that is used in the simulations described in this chapter.

In this configuration, the tracker detector has three pixel layers and ten silicon strip layers in the barrel, plus two pixel layers, three inner and nine outer forward disks of silicon detectors in each end-cap, as shown in Figure 14-4. The outer radius of the CMS tracker extends up to 107–110 cm. The length of the tracker is approximately 540 cm.

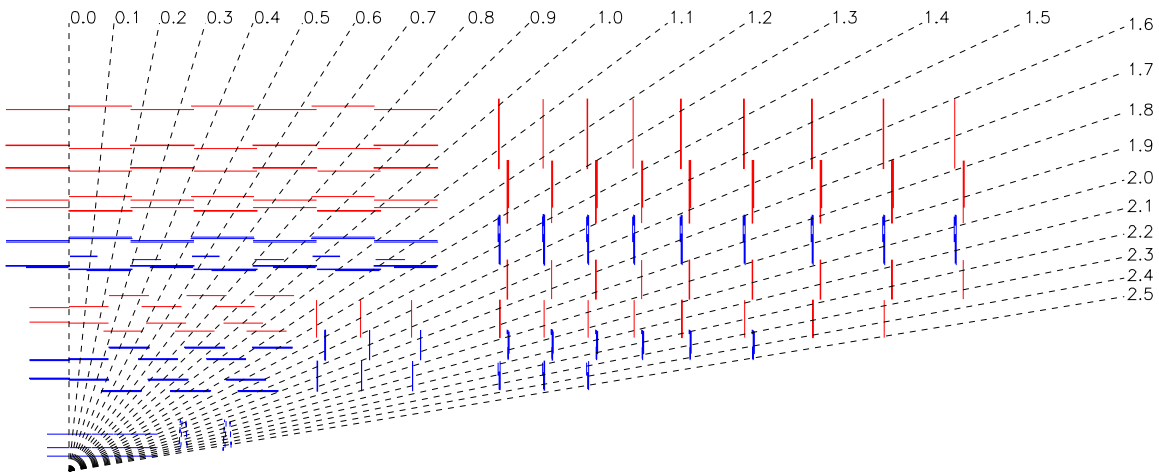


Figure 14-4 r-z view of a quadrant of the silicon part of the tracker. The inner barrel (TIB) has four layers, the outer barrel (TOB) has six layers. The inner (TID) end cap is made of three small disks on each side of the inner barrel. The outer end-cap (TEC) is made of nine big disks on both sides of the tracker.

The tracker contains a total of 15148 silicon strip modules with sensors whose thickness and pitch are listed in Table 14-1.

The pixel detector provides high resolution 3D space points required for the charged track pattern recognition. Due to its high position resolution (≈15 μm in both coordinates) it can determine the impact parameter of charged particles with high precision. It will also allow vertex reconstruction in three dimensions and will permit identification of b and tau-jets.

Table 14-1 Detectors types in the silicon tracker.

part	detectors	thickness (μm)	pitch (μm)
Inner barrel	2724	320	81/118
Outer barrel	5208	500	123/183
Inner disks	816	300	97/128/143
Outer disks	2512	300	96/126/128/143
Outer disks (2)	3888	500	143/158/183

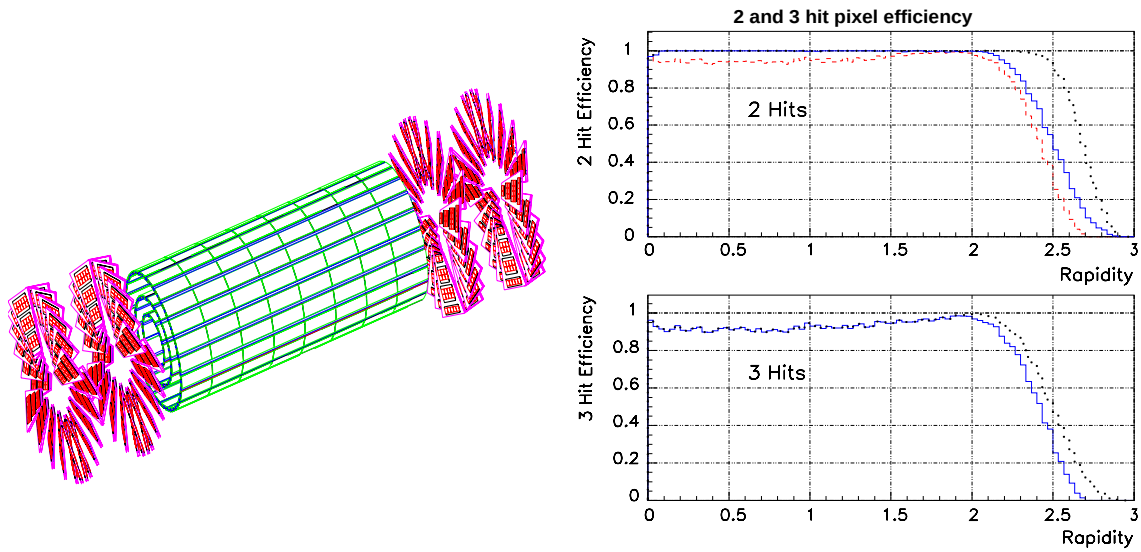


Figure 14-5 Left: illustration of the pixel detectors in the CMS tracker. Right: the efficiency for obtaining 2 (upper plot) and 3 (lower plot) pixel hits as a function of rapidity. The solid line is the three barrels + two disks configuration, the dashed line is for two barrels + one disk and the dotted line for three barrels + three disks.

The pixel detector consists of three barrel layers with two end-cap disks on each side (Figure 14-5 left). The three barrel layers will be located at mean radii of 4.4 cm, 7.3 cm and 10.2 cm respectively, and will be 53 cm long. The two end disks, extending from 6 to 15 cm in radius, will be placed on each side at z=34.5 cm and 46.5 cm. For the initial low luminosity runs the pixel detector will be staged to two barrel layers and one end-cap disk only.

In order to achieve the optimal vertex position resolution in both the (r, ϕ) and the z coordinates a design with a square pixel shape 150×150 μm^2 was adopted. To enhance the spatial resolution by analogue signal interpolation the effect of charge sharing induced by the large Lorentz drift in the 4 T magnetic field is used. Hence the detectors are deliberately not tilted in the barrel layers but are tilted in the end disks resulting in a turbine-like geometry. The whole pixel system consists of about 1400 detector modules arranged into half-ladders of 4 identical modules each in the barrel and blades with 7 different modules each in the disks. To read out the detector about 16000 readout chips are bump-bonded to the detector modules. The total number of readout channels (pixels) is about 44×10^6.

The pixel detector has been designed to provide 2 hit coverage up to rapidities of about η=2.2 (for tracks originating within 2σ around the centre). This is illustrated in Figure 14-5 (right, upper plot) where the efficiency for 2 hits is shown for the 3 barrel + 2 disk configuration. The efficiency is calculated for high transverse momentum muons originating from primary vertices smeared along the z axis with σ=5.3 cm,

but with a cutoff at 2σ. The staged, 1 disk configuration, limits the coverage to |η|=2.0. Note that the full coverage up to |η|=2.4 can be achieved with an upgrade to 3 disks. For stand-alone pixel tracking capabilities one needs at least 3 pixel hits per track. This can be achieved with the 3 barrel layers available for the high luminosity runs. The 3-hit coverage is also shown in Figure 14-5 (right, lower plot).

14.2.1.1 Tracker Material

The CMS tracker clearly includes both sensitive detector volumes and non-sensitive ones. Due to the fact that the tracker requires ~15 kA of low voltage current supply and dissipates approximately 60kW of power, a large fraction of the material consists of electrical supply cables and cooling services. Other non-sensitive material parts are cables, support structures, electronics, the beam-pipe and the thermal screen outside the tracker. The GEANT3/CMSIM simulation of the CMS tracker is tuned to describe all of these components to a high level of accuracy. Figure 14-6 (left) shows a simulated TOB detector rod, with the mother cable, cooling and electronics.

Figure 14-6 (right) shows examples of the simulated tracker volume (TIB and TOB). The readout electronics, the cooling and support structure reflect the current technical designs.

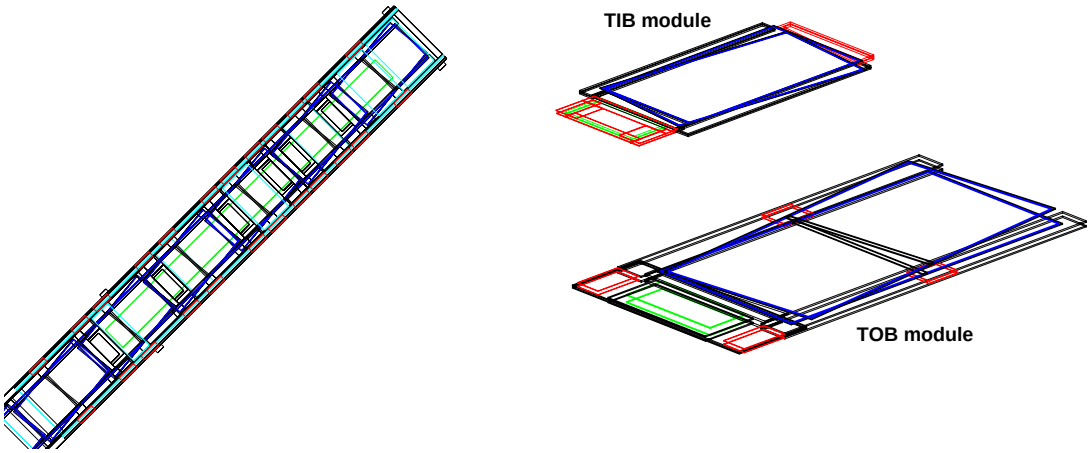


Figure 14-6 Examples of tracker volumes simulated in CMSIM. Left: a simulated TOB Rod; right: the Inner Barrel (TIB) and Outer Barrel (TOB) modules.

The decomposition of the tracker material in units of radiation lengths and in units of absorption lengths versus η for the different subdetectors is shown in Figure 14-7

14.2.2 ECAL Geometry and Simulation

The nominal (engineering specification) geometry of the ECAL is simulated in fine detail in the GEANT3/CMSIM model. Figure 14-8 shows a transverse section of the ECAL, including the end-cap preshower detector, as it is described in GEANT3/CMSIM.

The sizes and positions of ECAL barrel (|η| < 1.479) crystals are unchanged since the ECAL TDR [14-11]. Each half barrel consists of 18 super-modules each containing 20×85 crystals. The crystals are

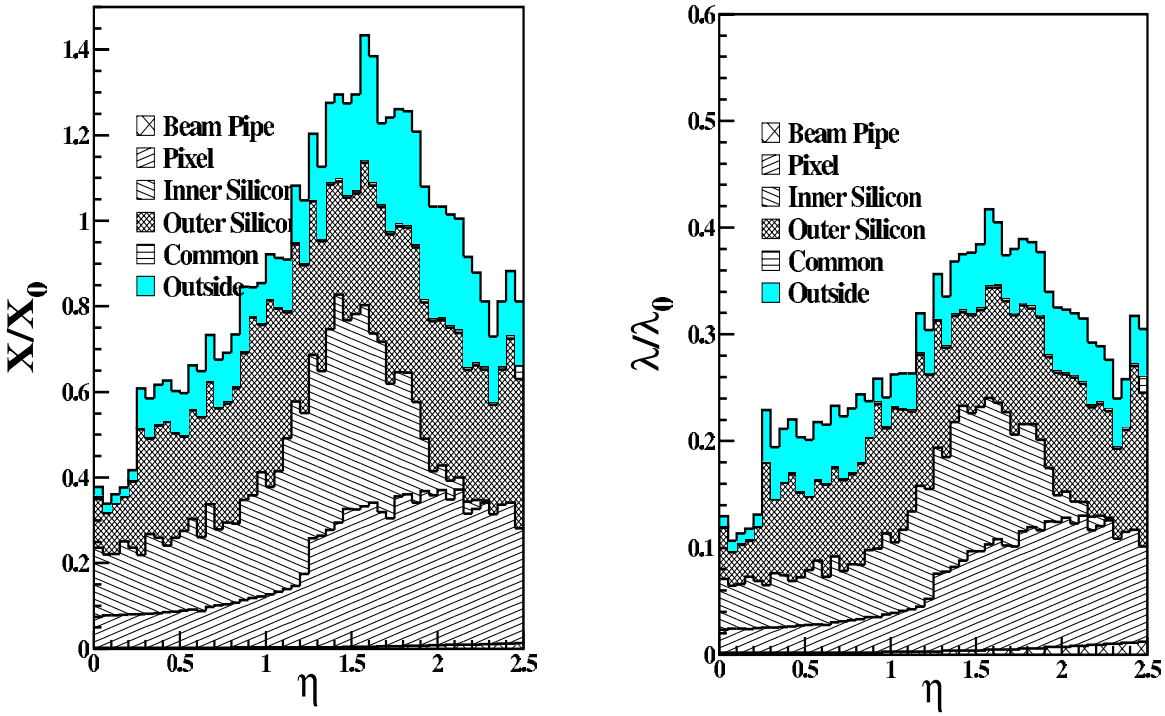


Figure 14-7 Material budget as a function of η for the different tracker subunits. On the left: material thickness in units of radiation length versus η. On the right: material thickness in units of interaction length versus η.

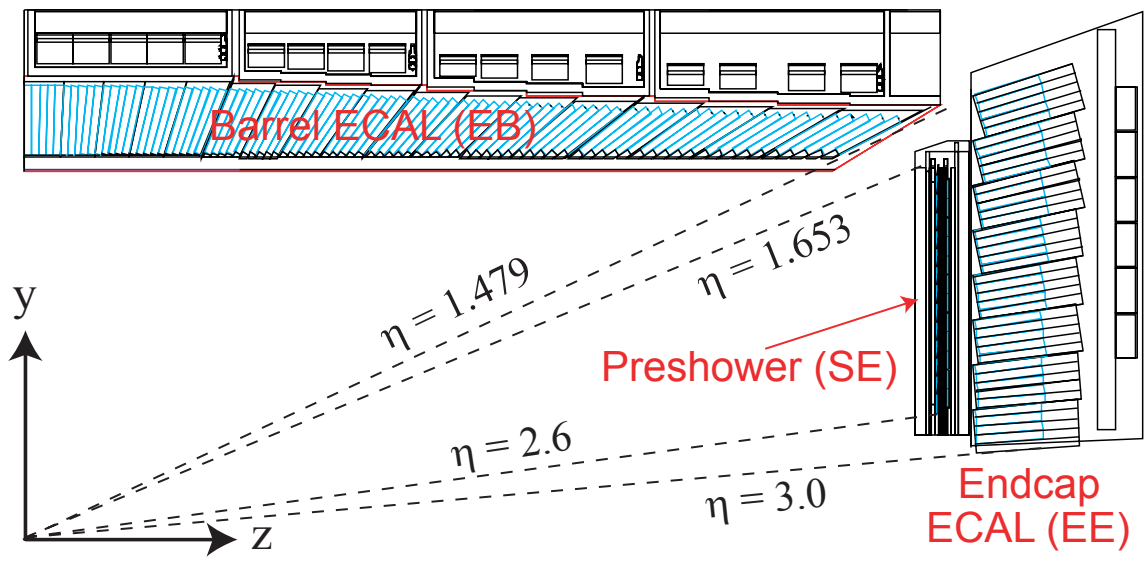


Figure 14-8 Transverse section of ECAL, as described in GEANT3/CMSIM (version CMS125).

tilted so that their axes make an angle of 3° with a line from the nominal vertex point, and each covers approximately 0.0174×0.0174 in η–ϕ. This is achieved using 17 pairs (left and right stereo images) of different crystal shapes. The crystals are 230 mm (25.8 X_0) in length. They are contained in a thin walled glass-fibre alveola structure. There is a light and thin structure in front of the crystals acting as a thermal shield and neutron moderator, which also carries the light fibres used for the laser-based monitoring system. All this is described and simulated in fine detail in the geometry model, together with the electronics,

the thermal regulation system and mechanical structures behind, as well as the mechanical structure separating the four modules which make up each super-module.

The end-cap consists of identically shaped crystals, slightly shorter (220 mm, 24.7 X_0) and somewhat larger in cross-section than the barrel crystals, grouped in mechanical units of 5×5 crystals (super-crystals) consisting of a carbon-fibre alveola structure. The crystals and super-crystals are arranged in a rectangular x-y grid, with the crystal axes off-pointing from the nominal vertex by angles between 2° and 5°. The end-cap mechanical design details, crystal sizes and the precise positioning of the super-crystals have been slightly revised since the ECAL TDR. In front of most of the end-cap ($|\eta| > 1.653$) there is a $3X_0$ preshower detector, consisting of lead radiators and two orthogonal planes of silicon strip detectors. As in the case of the barrel, the model of the geometry contains all these details together with a description of the mechanical structures and thermal regulation structures, and a less detailed modelling of the electronics behind them.

The active volumes of the ECAL, for which the deposited energy is recorded (i.e. the output of the GEANT3/CMSIM step in the simulation chain), are the crystals and the silicon strips. The arrival time of hits, as well as the energy deposited, is recorded. For the crystals the variation of the light collection efficiency along the length of the crystal is simulated by multiplying the energy, as it is deposited, by a nominal longitudinal light collection curve. This is a simple function of the distance from the crystal front-face, and is flat in the front half of the crystal, with a 5% linear rise over the rearmost 100 mm. Deviations from this nominal curve, as measured on production crystals, which deteriorate the energy resolution, are accounted for by a random smearing which is added later to the energy (see Section 14.3.1.4). Further discussion of longitudinal non-uniformity of light collection can be found in [14-12] and references listed therein.

14.2.3 HCAL Geometry and Simulation

The CMS hadronic calorimeter consists of four detectors, the barrel (HB), outer barrel (HO), end-cap (HE), and forward calorimeter (HF). The HB and the HE are inside the superconducting solenoid whereas the HO resides inside the return yoke of the magnet. The forward calorimeter is located beyond the return yoke of the magnet at a distance of 11.15 m from the center of CMS.

The HB covers the pseudorapidity region $|\eta| < 1.740$. The central barrel is divided into two half-sections each consisting of 18 identical wedges in ϕ. Each wedge consists of 17 sampling layers and these layers are ganged together as a single tower in the readout. There are cutouts at $\eta = 1.2$ for the routing of inner detector cables and after layer 16 for a patch panel. The absorber and scintillator plates are staggered in ϕ. The scintillator “megatiles” are made of scintillator wrapped in plastic. The first and last sampling layer have 8mm thick scintillator, the rest 4 mm. The first absorber layer is iron, and all subsequent ones brass of thickness 60 mm (first 9 layers) and 66 mm (the remaining 6 layers).

The HE covers the region $1.305 < |\eta| < 3.0$, overlapping with the HB in the region $1.305 < |\eta| < 1.740$. It is divided into a backward and a forward unit, each containing 18 wedges. The wedges have 19 layers of absorber and scintillator. The front layer of scintillator is 9 mm thick, subsequent ones are 4 mm thick. The absorber is in the form of 80 mm thick brass plates.

The HF covers the region $2.853 < |\eta| < 5.191$. It is located between 11.15 m and 12.80 m from the interaction region, and is made out of 18 iron wedges. The detector measures signals produced by Cerenkov radiation in quartz fibers. The forward calorimeter is longitudinally separated into two compartments, an electromagnetic section corresponding to 16 radiation lengths, and a hadronic section corresponding to approximately 9 interaction lengths.

The HO is located in the central region of the detector, $|\eta| < 1.305$, outside the solenoid in the five rings of the barrel return yoke. The central ring has two layers of 10 mm think scintillator separated by the tail-catcher iron and covers $|\eta| < 0.348$. The side rings contain one layer of 10 mm thick scintillator. The tower structure matches the HB.

The HB and HE calorimeters are subdivided into 72 ϕ sectors, for $|\eta| < 1.740$, and 36 for $1.740 < |\eta| < 3.0$. In this region, the energy in each physical tower is divided in half and assigned to 2 trigger towers so that the Level-1 Trigger will have a uniform $\eta-\phi$ segmentation through the HB/HE calorimeter. The HF has 36 sectors in ϕ and two readout channels for each η-ϕ segment (short and long fibers). For the Level-1 Trigger, the HF has 4 η divisions. The geometry is summarized in Table 14-2.

Table 14-2 Summary of HCAL Geometry. Note that in the range $1.740 < |\eta| < 3.0$ the energy in each physical tower is divided into 2 trigger towers, giving 72 ϕ divisions for the Level-1 Trigger.

η range	Detector names	number η divisions	number of φ divisions	number of readouts per η-φ division	tower size in η	Level-1 segmentation
0-1.218	HB,HO	14	72	2 (one in HB, one in HO)	0.087	same
1.218-1.305	HB/HO	1	72	4 (three in HB, one in HO)	0.087	same
1.305-1.392	HB/HE	1	72	4	0.087	same
1.392-1.740	HB,HE	4	72	2 (for fake EM)	0.087	same
1.740-2.65	HE	6	36	2 (for fake EM)	0.09,0.10, 0.113,0.12 9,.0.150,0. 178,0.150	same in η, 72 in ϕ
2.65-3.0	HE	3	36	3	0.12	same in η, 72 in ϕ
2.853-5.191	HF	13	36	2 (short and long fibers)	about 0.175 except for the first (0.111) and last (0.302)	4 η divisions and 18 φ divisions

Figure 14-9 shows the thickness of the calorimeter versus η. The calorimeter is more than 10 interaction lengths thick over $|\eta|<3.0$.

Simulated events were produced with the hadronic shower generator GCALOR. The energy cut-offs have been carefully tuned by comparing simulation results with test beam data [14-13]. The cutoffs are 100 keV for photons, and 1 MeV for electrons, neutral and charged hadrons. Saturation effects in the scintillator are taken into account by incorporating corrections due to Birk's law. Good agreement is found between our simulation and test beam data, including the non-linearity of the energy response, the energy dependence of the energy resolution, and the lateral and longitudinal shower profiles. The simulation results also show the "brightening effect" in the scintillator due to the magnetic field.

Shower generation in the HF was carried out using a shower library which contains several thousand single particle events (electrons and pion showers with energies up to 1 TeV). The libraries were created using a simulation with a detailed geometry that contained the location and material of every quartz fibre.

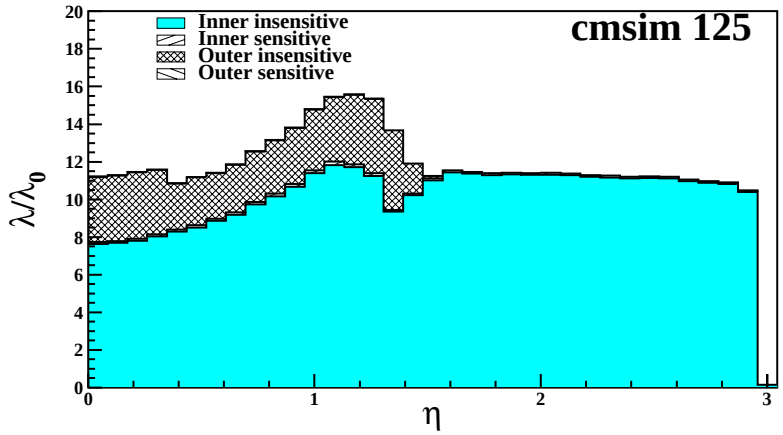


Figure 14-9 Material budget plot as a function of η through the last layer of the hadron calorimeter.

Cerenkov photon emission and transmission in the fibres were modelled in detail. The results obtained from the shower library closely match those from the detailed simulation [14-14].

14.2.4 Muon Chamber Geometry and Simulation

There are three muon systems in CMS: a barrel muon system composed of drift tubes (DT) in the central region (|η|<1.2), an end-cap muon system composed of cathode strip chambers (CSC) in the forward region (0.9<|η|<2.4), and resistive plate chambers (RPC) deployed throughout the central and forward regions (|η|<2.1). A longitudinal view of the three muon systems is shown in Figure 14-10.

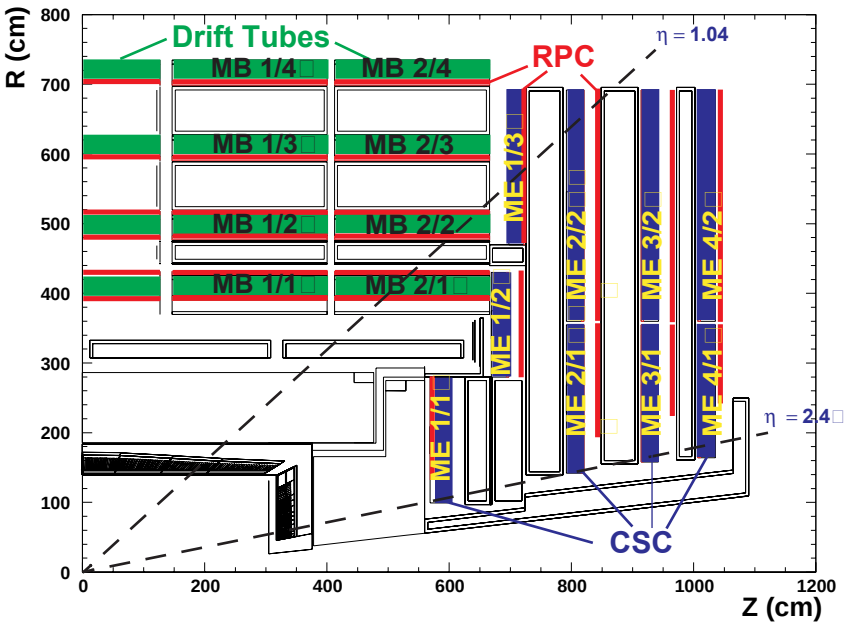


Figure 14-10 Cross-sectional view of one quarter of CMS showing the DT, CSC, and RPC muon systems.

The drift-tube system of CMS is segmented into 5 iron wheels along the beam axis, each 2.5 m in width, with chambers arranged in 4 stations at different radii (MB1–MB4) within each wheel. Each station is subdivided in azimuth into 12 sectors, with a single chamber mounted into each sector except for the top and bottom sectors of MB4, which have been split into two chambers. The total number of DT chambers is thus 250. A single DT chamber is composed of three superlayers of drift-tubes (two to measure the ϕ coordinate and one to measure z), where a superlayer is composed of four layers of drift-tubes. The outermost station, MB4, has only the 2 superlayers for the measurement of the ϕ coordinate. The transverse dimensions of a single drift cell, shown in Figure 14-11, measure 42 mm by 13 mm, and the wire length varies from 2 to 4 m.

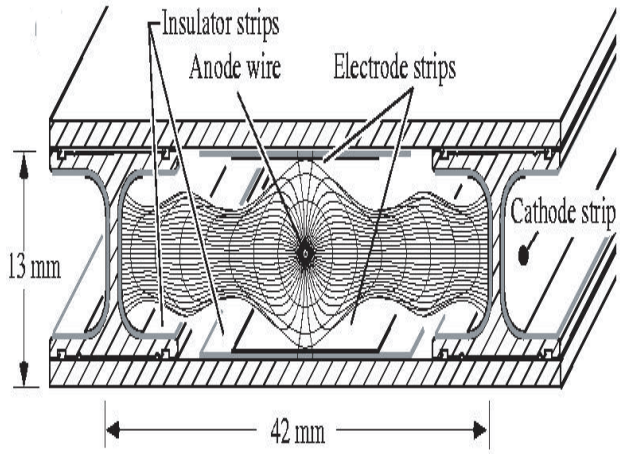


Figure 14-11 Transverse view of a drift-tube cell, with drift lines and isochrones for a typical voltage configuration of the electrodes.

The cathode strip chamber system of CMS contains four stations (ME1–ME4) of chambers mounted on three iron disks perpendicular to the beamline on each end of the apparatus (see Figure 14-10). The fourth CSC station, ME4, is staged in the current planning and will not be available for initial low luminosity running, but is included in the HLT simulations for which the results are described here. A single CSC station is composed of two rings of chambers (three for ME1), where each chamber has six active gas layers. A single layer has a gap thickness of 9.5 mm, cathode strips aligned radially with a pitch that varies from 7 to 16 mm, and anode wires aligned in the orthogonal direction with a pitch of 3.2 mm (but ganged into groups of 5–16 wires for readout). Each chamber is trapezoidal in shape with either a 10° or 20° angular coverage in azimuth (ϕ), depending on the ring and station, with the largest chambers measuring 3.4 m in length and 1.5 m in width. The complete system contains a total of 540 chambers. The chambers in the innermost ring of the first station (ME 1/1) are segmented into two regions in radius. The most forward of these two regions ($|\eta|>2.1$) is not expected to participate in the Level-1 Trigger.

The resistive plate chamber system in the barrel has six radial stations placed within the same 5 wheels as the DT system (two RPC chambers in both the MB1 and MB2 stations, and one chamber each for MB3 and MB4). In the end-caps, the RPC system shares the same four stations as the CSC system. The chambers are a double-gap design, with a common pick-up segmented into strips that are rectangular in the barrel and trapezoidal in the end-caps. The gap thickness is 2 mm. The total number of chambers is 612.

The simulated hit (a “SimHit”) of a charged particle crossing an active layer in each of the three muon systems contains the entry and exit points for each active gas layer. GEANT3 is not used to model the interaction of this primary particle in the gas, only to estimate its energy loss. The interaction in the gas it-

self is considered as part of the detector response step and is simulated in the ORCA package. The detector response simulation is described in a later section.

14.3 Detector Response Simulation and Reconstruction Software

The CMS reconstruction software, “ORCA” (Object-oriented Reconstruction for CMS Analysis) [14-1], uses modern technology and in particular object-oriented programming. It is implemented using the C++ programming language. The ORCA project covers currently not only the reconstruction tasks, but also includes code for simulating detector response and the Level-1 Trigger, as well as High-Level Trigger and analysis code.

ORCA has adopted a two-level decomposition – subsystems and packages – to match the different tasks it covers. Typical subsystems are Calorimetry, Tracker, Muon or Trigger matching the hardware components of CMS. Other subsystems provide common services for several subdetectors (CommonDet, CommonReco) or analysis tasks (Jets) or high-level reconstruction (TrackerReco, MuonReco, Vertex). Finally, subsystems for High-Level-Trigger selection and analysis are provided (ElectronPhoton, JetMetAnalysis, MuonAnalysis, bTauAnalysis). Figure 14-12 shows the subsystems and their dependencies. Each subsystem contains a set of packages fulfilling cleanly separated subtasks.

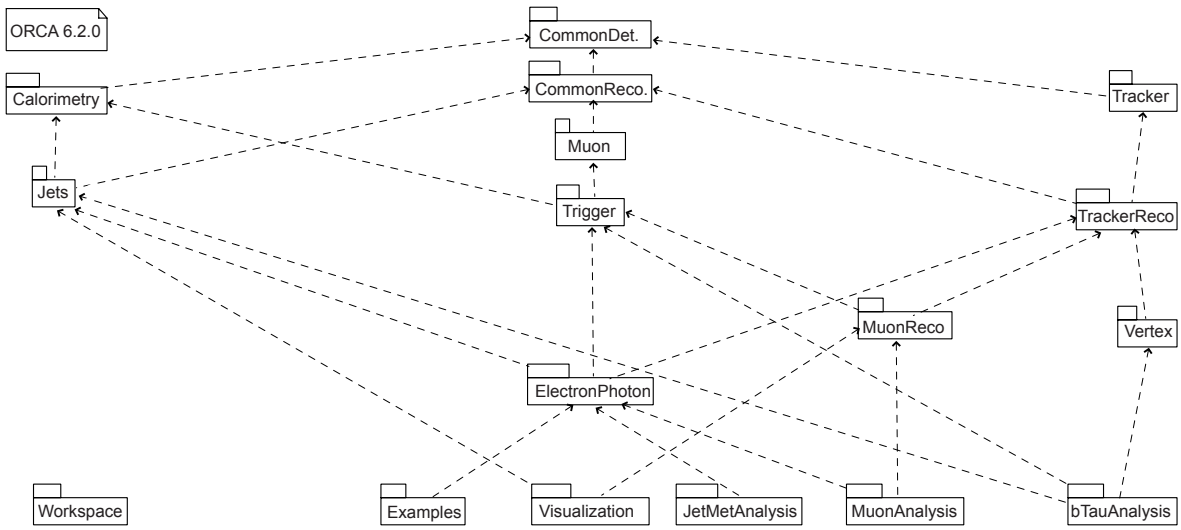


Figure 14-12 Subsystems in the ORCA reconstruction package and some of their inter-dependencies.

The reconstruction software is based on the COBRA (Coherent Object-oriented Base for simulation Reconstruction and Analysis) software framework [14-1]. COBRA provides basic services and utilities: physics type services (histogrammers, fitters, mathematical algorithms, geometry and physics calculation routines) and computer services (data access, inter-module communication, user interface, etc.).

COBRA also contains a set of classes that capture CMS-specific concepts like detector components and event features and implements a control policy that orchestrates the instances of those classes that handle flow control, module scheduling, input/output etc.

Two main concepts implemented in the COBRA framework to steer the event processing are “action on demand” and “implicit invocation”. These concepts imply that modules register themselves at creation time and are invoked when required. In this way only those modules that are needed are loaded and exe-

cuted. Implicit invocation uses additional mechanisms to construct the required modules, to customize them and to manage dependencies among them. Modules whose change of state depends on the occurrence of some external “event” register as “observers” to the “dispatcher” of such an “event”. Typical “events” are the reading of a new physics event from the detector, a new run or a new detector setup. On the arrival of a new “event” the dispatcher informs all registered observers. These can then update their state accordingly and eventually perform a specific action (like the reconstruction of some object, see below). To avoid useless computation, such an action can be deferred until a client asks an observer for a service (e.g. the delivery of some reconstructed tracks).

14.3.1 Simulation of Detector Response (DIGI Creation)

After combining a signal event with the selected pile-up events for a particular luminosity, the detector response is simulated taking into account the simulated hits of all events of a crossing. The time information of each hit is shifted according to the bunch crossing to which the corresponding event belongs.

The finite response time and time resolution of the CMS subdetectors defines for each subdetector a time window of sensitivity, ranging from a few tens of nanoseconds to several hundred nanoseconds. To simulate the full detector response and occupancy, simulated hits in the full time interval of sensitivity need to be taken into account.

The simulation of the detector response is different for each subdetector and is described in the following sections.

14.3.1.1 Pile-up Treatment

The LHC luminosity is such that a number of pp collisions occur each bunch crossing. In most of these collisions “minimum-bias” events are produced. For the simulation these “minimum-bias” or “pile-up” events are distinguished from hard scattering events that trigger the detector readout. The latter are referred to as “trigger” or “signal” events.

Because there is only 25 ns between consecutive bunch crossings, the CMS detector will integrate over pile-up events produced before and after the signal event. In total up to about 160 pile-up events need to be overlaid with a single signal event to properly simulate the detector response at high luminosity.

The number of bunch crossings to be piled-up is determined by the subdetector with the slowest response time. The values used are -5 to +3, i.e. five bunch crossings preceding the trigger bunch crossing, and 3 following it. This also takes into account most signals generated by soft particles looping in the detector.

For each crossing an individual signal event is combined with a random set of minimum-bias events taken from a pool of pile-up events. The size of the pool of pile-up events is chosen to be large enough to ensure a unique combination for each crossing. This “recycling” of pile-up events is needed due to the limited CPU resources available.

14.3.1.2 Pixel Detector Response

The entrance and exit points in the pixel sensitive volumes, together with the deposited energy, are recorded during the GEANT track propagation, providing input information for signal simulation. The track segment inside the sensor volume is divided into 10 μm long subsegments. Each subsegment is assigned a fraction of the energy using the GEANT3 routine GLANDZ [14-15]. This distribution takes into account Landau fluctuations in thin material layers.

The charge from each track subsegment is drifted to the detector surface and simultaneously diffused in the perpendicular plane. The diffusion is assumed to be Gaussian and is proportional to the square-root of the drift length. The diffusion constant is normalized to 7 μm for a 300 μm thick detector. The final charge distribution on the detector surface is calculated as a sum of two-dimensional Gaussian distributions. The magnitude of the Lorentz drift in 4 T magnetic field is defined by the drift length and the Lorentz angle, which depends on the strength of the magnetic field.

The resulting two dimensional charge distribution is mapped to the pixel geometry and the fraction of charge collected by each pixel is determined. The cut-off value for integration of the charge distribution is 5 σ. A list of hit pixels for all contributing tracks is formed. If a single pixel is hit by more than one track, the charge contributions are added together.

The noise is assumed to be Gaussian, centred at zero. All hit pixels have the noise contribution added to them. In addition, noisy pixels with charge amplitude larger than the pixel threshold are generated according to the tail of the Gaussian distribution. The default noise used in the simulation is σ=500 electrons.

To digitize the signal collected in each pixel, the charge is multiplied by a gain factor and converted into an integer number, so simulating ADC digitization. Signals which exceed the ADC range are assigned a saturation value which depends on the ADC maximum bit number. Both the gain and the number of ADC bits (typically 6) can be adjusted. Pixel zero-suppression is simulated by comparing the digitized signal to a threshold for each channel (Figure 14-13). The peak close to 35000 electrons corresponds to signal saturation and that at 1000 electrons is due to noise. The threshold is defined in units of noise and is typically set at 5 σ (2500 electrons).

The finite space available in the pixel readout chip requires the size of data buffers, and the complexity of the circuits, to be carefully optimized. Although the readout efficiency has been maximized, some data losses will always be present. Additional pixel inefficiencies are expected due to unbonded and noisy pixels. To simulate such inefficiencies, randomly chosen pixels are erased. In addition some pixel double-columns and some whole readout chips are cleared as well. The data losses are in general dependent on the CMS trigger rate, LHC luminosity and the module distance from the interaction point. They were parameterized according to results obtained from detailed simulations and included in the ORCA digitization code.

Figure 14-13 shows the simulated charge distribution for minimum-bias events and for single muon tracks (100 GeV). Part a) of the figure shows the total charge deposited in the pixel barrel by minimum-bias (solid line) and muon (dashed) tracks. The edge at 18000 electrons corresponds to a mip signal. Part b) shows the signal seen by a single barrel pixel for 125 μm and for 150 μm pixels. The average signal is about 12000 electrons, which corresponds to the S/N ratio assumed in this simulations of 24:1.

Figure 14-14 shows the average number of hit pixels in the pixel barrel layer at 7 cm as a function of rapidity for single muon tracks (100 GeV). The pixel cluster size in r-ϕ is determined by the Lorentz shift and is about 2 pixels for the whole rapidity range. The cluster size in the z coordinate is determined mostly by the track impact angle and varies with rapidity from 1 to 6. In this case all tracks are assumed to originate from the detector centre (no vertex smearing).

14.3.1.3 Silicon Microstrips Detector Response

The entry and exit points, as well as the energy deposited in the silicon (E_{ion}) by the incident particle, are provided by the GEANT3 package during track propagation in CMSIM.

The signal from real silicon detectors is processed by the APV25 readout chip [14-16] [14-17]. The finite shaping time of this amplifier is the primary cause of signal pile-up from out-of-time bunch cross-

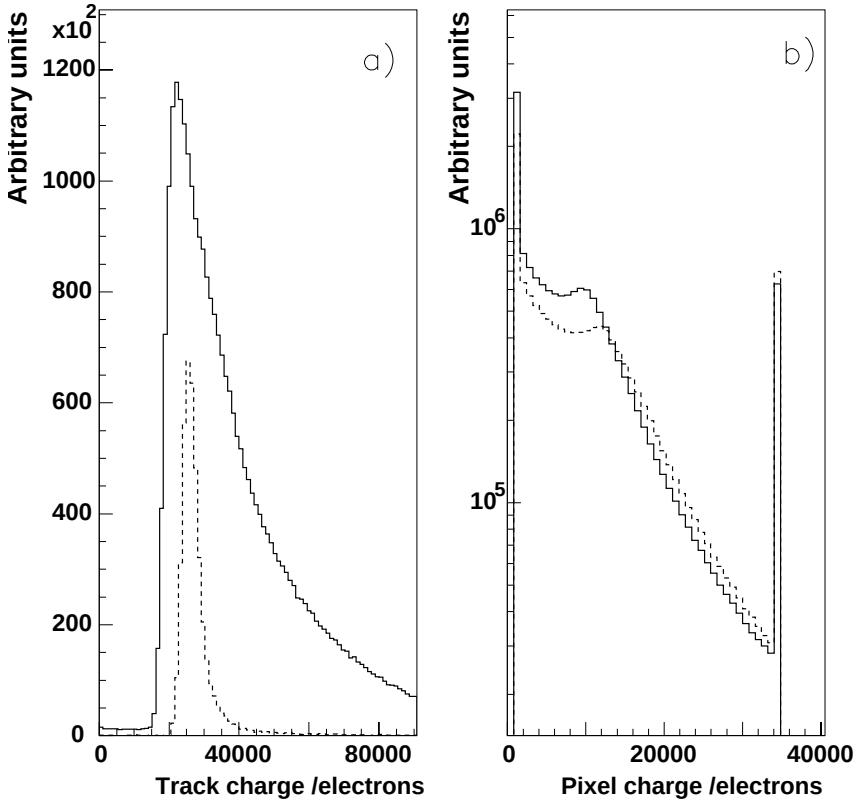


Figure 14-13 a) Simulated charge distribution deposited by minimum-bias tracks (solid line) in a 250 μm barrel detector at a radius of 7 cm. For comparison, charge from 100 GeV muon tracks at normal incidence is also shown (dashed line). b) Simulated charge distribution seen by a single barrel pixel 125 μm (solid line) and 150 μm (dashed line) large.

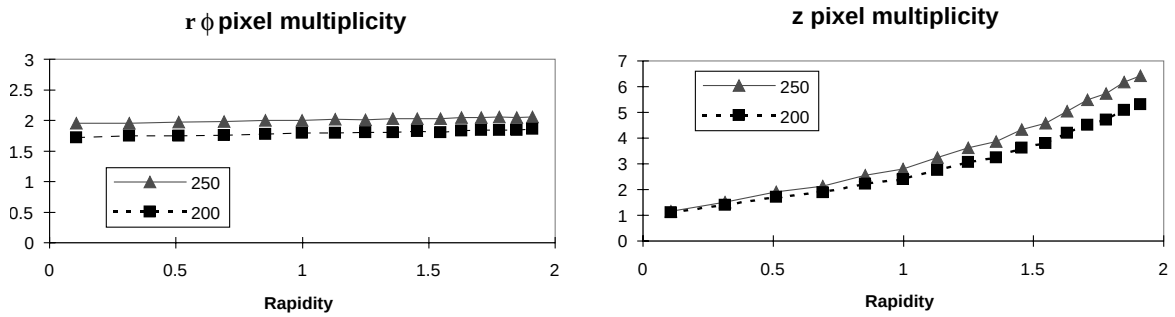


Figure 14-14 Average number of pixels hit in the barrel layer at 7 cm (pixel size 150 μm) versus rapidity for a single 100 GeV muon track. The solid line (triangles) is for a fully depleted detector (250 μm) and the dashed line (squares) is for a detector with partial depletion of 200 μm.

ings. The APV25 will usually operate in “deconvolution mode”, since this gives it the best time resolution. Figure 14-15 shows the APV time response, fitted with a Gaussian of width 12.1 ns [14-16][14-17]. The software simulates the dependence on the pile-up arrival time, by weighting the energy loss from

GEANT according to the time at which the hit was produced in the silicon relative to the nominal readout time.

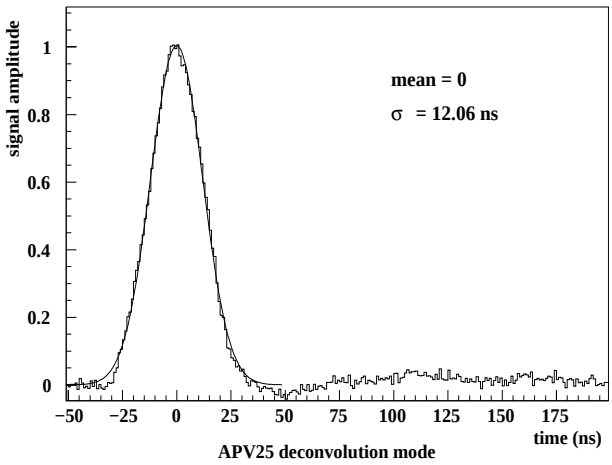


Figure 14-15 APV time response in deconvolution mode. The superimposed Gaussian fit shows the response time of the electronics.

Inside the silicon wafer, the track path is simulated by joining the entry and exit point of the GEANT hit by a straight line; the path is then subdivided into N_{seg} segments (currently 10 times the path-length traversed by the particle in units of strip pitch). Independent Landau fluctuations are assigned to each segment using the GEANT3 routine GLANDZ [14-15], and the normalization is fixed to the total energy deposited. For each segment a cluster of hole-electron pairs is created using a calibration factor of 2.77×10^2 pairs/keV: in a 300 μm thick silicon layer about 3×10^4 pairs are produced by a minimum ionizing particle (mip) at normal incidence.

The charges are drifted to the detector surface, taking into account the effect of the magnetic field. The drift direction is therefore given by the direction of the vector: $\vec{U} = \vec{E} + \mu_H \vec{E} \times \vec{B}$, where the Hall mobility, μ_H, which depends on hole mobility and environmental conditions (temperature, magnetic field, etc.), is modelled using experimental data. The Lorentz angle Θ_L is about 6 degrees/Tesla ($\tan\Theta_L=\mu_H B=0.12$), and varies slowly with environmental parameters and irradiation. Even after 10 years of irradiation at LHC conditions, the expected variation in the hole mobility is only ~2%.

Diffusion is taken into account by computing the expected spread of the charge on the detector surface. When the charges reach the surface, the strips within 3σ of the diffusion centre collect the signal. Each strip is assigned a fraction Q_i of charge, weighted with the distance from the expected impact point with Gaussian probability. The charge Q'_i induced on each strip is computed taking into account the cross-talk between neighbouring strips; only the cross-talk to the two nearest strips is considered, and the charge on the *i*-th strip is evaluated as

$$Q'_i = C_1 Q_{i-1} + C_0 Q_i + C_1 Q_{i+1}$$

Current test beam data suggest large values for the cross-talk. In the current simulation, the central strip gets 76% of the charge (C_0), while 12% is induced to each of the neighbouring strips (C_1).

The noise on silicon strip detectors signals is added using Gaussian statistics. On strips with a signal, a Gaussian noise of about 2000 electrons is added. On strips without signal, a Gaussian tail generator is used to speed up the computation: since the ADC signal is zero-suppressed by the FED, only strips with noise sufficient to pass the zero-suppression threshold need be simulated. The number of strips which ex-

ceed the threshold is calculated using Poisson statistics, and a noise exceeding the threshold is simulated only for this small subset of strips (about 5% with the standard threshold cut at 2σ).

An alternative and more realistic operating mode can optionally be enabled in the simulation, which accounts for effects due to the overall electronics chain. In this mode, the noise is simulated on each strip, together with common mode noise and pedestals as obtained from laboratory measurements. The FED common mode and zero-suppression algorithms are simulated at the APV (128 strips) level, as are possible algorithms which could be used to calibrate the pedestals and noise. This allows the study of algorithms to be implemented in the FED and in the FED crate controller. The simulation shows that the currently proposed common mode subtraction algorithm will function correctly at CMS, and initial results on possible pedestal and noise calibration algorithms are encouraging (Figure 14-16).

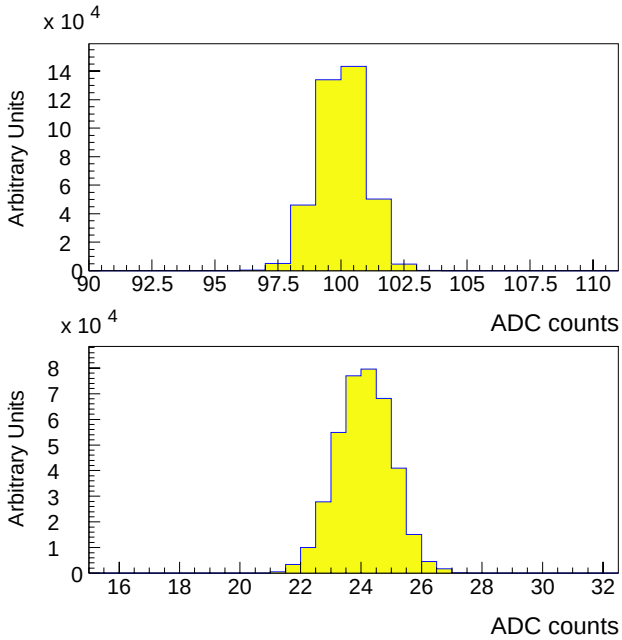


Figure 14-16 Reconstructed noise and pedestals in a TOB detector, estimated using a calibration algorithm on a sample of 300 minimum-bias events. The true values of the noise and pedestal, input to the simulation, are 100 and 24 respectively.

The FED implements a zero-suppression algorithm to limit the data rate from the tracker: a strip is accepted only if the significance of its signal exceeds 5σ, or if it exceeds 2σ and is part of a cluster. This algorithm has been implemented in the software simulation.

By taking into account all estimates reported in this section, a signal to noise ratio of 12 to 14 is obtained for tracks at normal incidence. The signal to noise ratio is defined here as the ratio of the most probable cluster charge to the noise of one strip. Figure 14-17 shows the S/N shape, with tails due to Landau fluctuations.

14.3.1.4 ECAL Detector Response

The ECAL detector response simulated in ORCA consists of:

- simulation of the digitization of hits by the readout electronics;

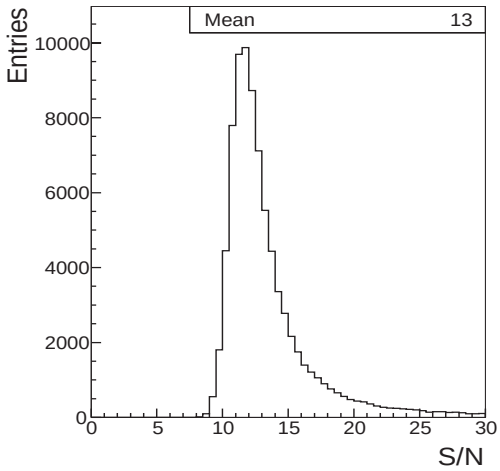


Figure 14-17 S/N ratio for silicon microstrip detectors. All energy losses are scaled to perpendicular impact.

- addition of Gaussian random smearing to the energy, to account for electronics noise, intercalibration uncertainty, crystal deviations from the nominal longitudinal light collection curve, and photostatistics.

Since the ECAL front-end electronics is specified such that digitization noise will make a negligible contribution to the errors on the measured energy, the main function of the digitization simulation for the HLT work is to provide a realistic modelling of out-of-time pile-up, i.e. pile-up from bunch crossings other than the one in which the Level-1 Trigger occurs. A function is used to describe the time development of the signals from the ECAL FPPA. When used together with the recorded time at which the energy was deposited, and sampled at 40 MHz, the observable time frames can be simulated. These time frames, consisting of 10 time samplings[1], correspond to the raw data which will be input to the HLT farm. The simulation of the preshower detector follows the same pattern, although the time development of the signals is faster, and the time frames consist of only 3 time samples.

In current practice, for reasons of technical simplicity and simulation speed, the Gaussian random smearings are added to the energy reconstructed from the time frames (see Section 14.3.2.2) rather than to the deposited energy before the simulation of the time frames. Table 14-3 lists the Gaussian smearings added. It is interesting to note that, since the ECAL is a homogeneous electromagnetic calorimeter, these smearing dominate the energy resolution for unconverted photons in the barrel. For example, making the simplified approximation that most of the energy is deposited in a single crystal, the smearing detailed below gives a resolution of about 0.8% for a 35 GeV photon reconstructed in 5×5 crystals, as compared to about 0.9% measured with full simulation.

Table 14-3 Gaussian smearing applied to the energy deposited in ECAL (for the preshower the value given is expressed in terms of reconstructed energy).

	Barrel	**End-cap**	**preshower**
Electronics noise	σ = 40 MeV	σ = 150 MeV	σ = 4 MeV
Photostatistics (E in GeV)	$\sigma/E = 1.8\%/\sqrt{E}$	$\sigma/E = 2.4\%/\sqrt{E}$	—
Intercalibration error and longitudinal non-uniformity	$\sigma/E = 0.5\%$	$\sigma/E = 0.5\%$	—

1. Although it is intended to record a full 10 time samples, the reconstruction used for the HLT studies used only 8.

14.3.1.5 HCAL Detector Response

The HB/HE/HO readout electronics consist of a hybrid photodiode and a QIE chip which integrates and digitizes the pulse at the bunch crossing frequency (every 25 ns), and is located in crates mounted near the back of the detector. The digitized data is then shipped to electronics that constructs the trigger primitives which are sent to the Level-1 regional calorimeter trigger. The HCAL trigger and readout electronics (HTR) holds the data for a Level-1 accept, and then sends it to the DAQ for further processing.

Our simulation of the electronics for the HB/HE/HO starts with the information provided from GEANT on the energies deposited in the scintillator and their timing. The energies are converted to number of photoelectrons using the conversion factor of 1 photoelectron per 0.68 MeV in the HB, per 0.45 MeV in HE, and per 0.25 MeV in the HO. Fluctuations of this average number are simulated assuming a Poisson distribution.

The shape of the signal at the input to the QIE is influenced by the shape, size, and time constant of the scintillator and wavelength shifter, the length of the clear fibers that bring the signal from the detector to the front-end electronics, and the signal shaping from the time constant of the HPD.

Figure 14-18 shows two possible shapes for the response. Test beam data is currently being taken to verify this shape and to check its variations as a function of rapidity (the scintillator size and fibre length vary with rapidity). Preliminary results from test beam data taken in 2002, i.e. after the simulation was completed, favour the shorter pulse shape. This shape is used in the estimate of the bandwidth requirements[1]. However, the simulation uses the longer of the two shapes. Because the longer shape stretches the pulse over more crossings, the total noise over the integrated pulse is larger. Also, the longer pulse shape will make the HCAL analysis more sensitive to pile-up from out-of-time crossings. The bunch crossing identification algorithm used in making the Level-1 Trigger primitives works better for the shorter pulse. Thus the longer pulse adopted for the simulation was a conservative choice. Note that the longer shape spreads the signal over approximately two bunch crossings. The integral of this pulse shape is set equal to the number of photoelectrons. Noise is then added, Gaussian in E, equal to 1.5 photoelectrons per time sample per read out depth segment, uncorrelated between time buckets (which corresponds to about 240 MeV after the corrections described below).

The pulses from the different energy depositions (both from the current crossing and from up to 5 previous and 3 subsequent crossings) are then added.

The QIE integrates and digitizes this pulse. Data are stored in a pseudo-floating point format, with 5 bits of mantissa, 2 bits of range, and two bits for capacitor identification.

The HF electronics uses a conventional photomultiplier tube instead of a hybrid photodiode, but nevertheless the same electronics as for the HB/HE/HO is used. The HF pulse shape is short enough to be entirely contained in one bunch crossing and is thus not affected by pile-up from previous or later bunch crossings. The magnitude of the noise used in the HF simulation is 0.125 photoelectrons, and the ADC count size is set at 0.43 photoelectrons.

14.3.1.6 Muon Detector Response

The responses of the TDCs form the output of the digitization step of the barrel Drift Tube system. Particular care is taken in simulating the behaviour of the drift cells as a function of the muon direction and im-

1. The shorter pulse length has less noise, and thus will give fewer towers over the zero-suppression threshold. It will also have fewer towers that are over the threshold because of energy from previous crossings.

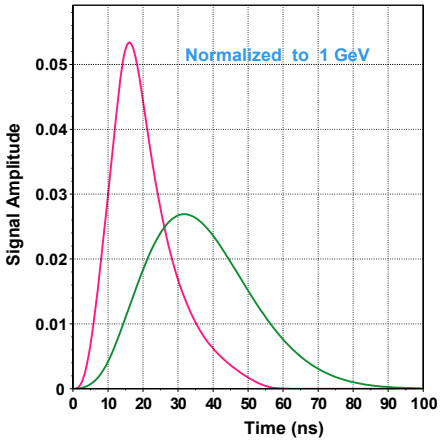


Figure 14-18 Signal shape at the input to the QIE for the HB, HE, and HO calorimeters. The two curves indicate the uncertainty in the shape at the time the simulation was made. Test beam data taken since that date favour the shorter shape. The longer shape was used in most simulations in this TDR.

pact position with respect to the sense wire, and of the residual magnetic field in the air gaps of the magnet iron yokes, where the chambers are located. The results of a study of the cell response based on a dedicated simulation with the GARFIELD package [14-18] were parameterized in terms of an "effective drift velocity" and included in the digitization simulation. Drift times obtained from this parameterization differ by at most a few nanoseconds from those obtained with the full simulation [14-19]. The latter was performed with the cell design described in the Muon Technical Design Report [14-20], and the results were linearly rescaled to the new cell size (width 4.2 cm instead of 4.0 cm) adopted for the final design of the barrel muon chambers. The simulated variation of the drift velocity with the magnetic field is consistent with that observed in test-beam data using a muon chamber prototype with the final cell design (*e.g.* 0–7% for 0–0.6 T for tracks inclined by 15° with respect to the direction normal to the chamber plane) [14-21]. The resulting drift time was smeared so as to obtain a 4 ns resolution, corresponding to an intrinsic cell resolution of about 220 μm, as measured in test-beam data. The TDC output signal for the hit reconstruction was obtained from this drift time by adding the muon time-of-flight from the collision vertex and the propagation time of the signal along the cell wire. The simulated distribution of drift times in a cell is shown in Fig. 14-19. The average muon time-of-flight from the collision vertex to a given chamber is assumed to have been subtracted at TDC level. In case of multiple hits in a cell due to delta rays and/or muon showering, the signal from the hit nearest the wire was retained.

The digitization step of the Cathode Strip Chamber system involves simulating the responses of the ADCs and discriminators connected to the strips and wires, respectively, and is described in detail in [14-22]. The GEANT thin-layer approximation is used to model energy loss in the gas. This uses the atomic structure of the gas molecules in the calculation of the collision cross section. Each ionizing collision along the path of the primary muon through the gas produces a free electron. Further delta electrons may be produced, and each electron is transported to the neighbouring anode wire taking into account the local electric and magnetic fields. A scale factor is applied to account for electron losses due to the attachment coefficients of the $Ar/CO_2/CF_4$ gas mixture. Gas multiplication occurs in the vicinity of the anode, with fluctuations in the avalanche size. The simulated number of drift electrons for 100 GeV muons is shown in Figure 14-20. The avalanche induces an image charge on the cathode planes with a spatial distribution taken from reference [14-23].

To create the analog signals seen by the CSC wire and strip electronics, parameterizations of the amplifier and shaper response are convoluted with the ion drift collection time. Note that the signal may contain contributions from drifting electrons due to background hits from other beam crossings. Cross-talk, both

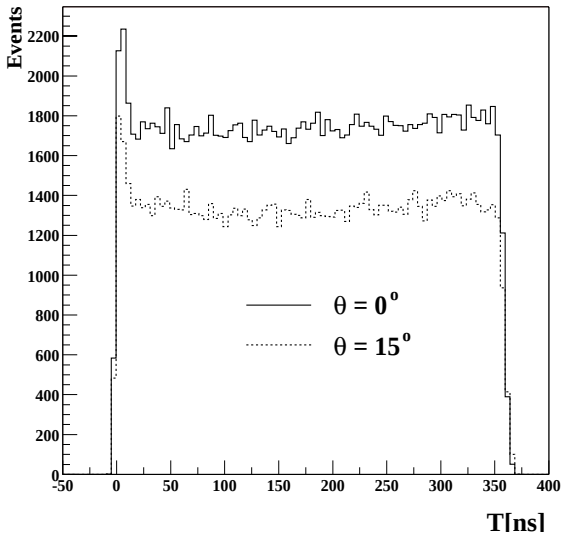


Figure 14-19 Distribution of the drift times in a DT cell for two angles of inclination with respect to the direction normal to the chamber: 0° and 15°.

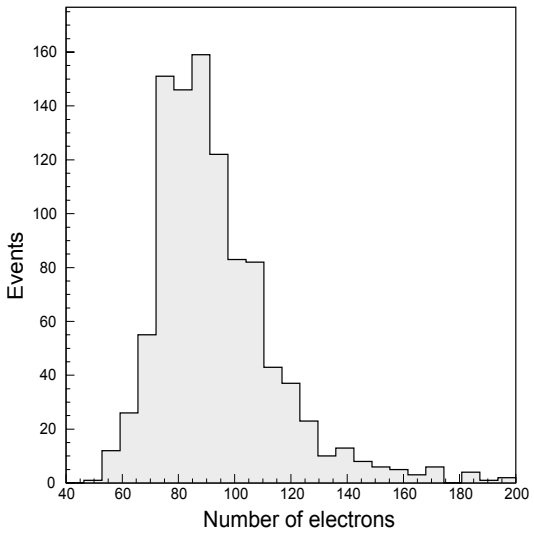


Figure 14-20 The number of simulated drift electrons produced by 100 GeV muons hitting an ME 2/1 chamber at normal incidence.

capacitive and resistive, is included in the strip signal. The capacitive component is proportional to the derivative of the neighbouring signal, and the resistive component is proportional to the signal itself. The level of the cross-talk is normalized to that observed in test beam data from the largest chambers (ME 234/2) and scaled proportionally to the length for the smaller chamber types. Each strip which satisfies the Local Charged Track (LCT) comparator logic causes the readout of a group of 16 strips in the simulation. Within such a group, noise is simulated on the empty strips that neighbour the signal strip, and remaining empty strips are suppressed. A readout dead time of 200 ns is assumed. This differs slightly from the actual DAQ electronics readout, where the presence of a LCT initiates the readout of a front-end board that covers a region 16 strips wide and 6 layers deep, but the effect is expected to be negligible. Finally, the storage of the strip signals in Switched Capacitor Arrays (SCA) is simulated. The signal shape is sampled and stored at 8 times, each 50 ns apart. Noise is added to each SCA sample, which is deter-

mined from test beam data so that bin-to-bin time correlations are accounted for. The SCA time samples are used for the CSC track segment reconstruction described below.

The RPC response is assumed to take place within 20 ns of the passage of a charged particle through the detector with a 3 ns Gaussian distributed jitter, which also accounts for the contribution from the front-end electronics and the cables to the link board. The 20 ns wide time gates were adjusted in order to accommodate triggering signals. The RPC cluster size is set to 1.5 strips, and the average hit efficiency is 95%. It should be noted that intrinsic chamber noise, to which RPC chambers are most susceptible, is not included in these simulations because of CPU time constraints.

14.3.2 Reconstruction

From the software point of view, reconstruction is modelled as a process of data reduction whose main client is the analysis. The reconstruction process is seen as a collection of independent reconstruction units, each one providing a set of corresponding reconstructed objects as output.

Reconstruction units may depend on

- real data provided by the DAQ system or simulated data from a simulation program
- objects produced by other reconstruction units
- environmental data like detector description, detector status, calibrations, alignment
- parameters to steer the reconstruction algorithms

The reconstruction process is carried out in two steps. First a local reconstruction is performed separately for the various subdetectors. The corresponding reconstruction units use as input the results of the digitization step, corresponding to what will be the digital response from the readout electronics. For simulated data the result of the digitization step (DIGIs) are used to produce “reconstructed hits” (RecHITs). The framework provides a mechanism to store and retrieve RecHits to/from databases. The access to RecHits is provided via DetectorUnit objects that can also encapsulate geometrical properties and calibrations. DetectorUnits also provide access to the raw data via ReadOutUnits. These model the readout electronics of the detector and have the responsibility to create DIGIs - from the online DAQ or during the digitization - or read the DIGIs from the database. They are “observers” of physics events. The relationships between the classes involved are shown in Figure 14-21.

The second step is global reconstruction which delivers “reconstructed objects” (RecObjs) suitable for further global reconstruction or for physics analysis.

Reconstruction is performed by RecUnits that register as observers to the framework and get notified about the arrival of new physics events. The RecObjs are produced in collections and iterators are provided to access them. The reconstruction algorithms are contained in Reconstructor objects that interact with the RecUnits. They obtain from the framework the handle to the “event” and contain the algorithm. Different Reconstructor/RecUnit combinations may produce the same RecObjs. This allows an easy exchange or comparison of different algorithms. The RecObjs produced by different Reconstructors are identified with ASCII strings. Figure 14-22 shows the relations of the classes involved.

It is required that reconstructed objects can be made persistent, and COBRA provides full support to store and retrieve RecObjs belonging to a given event. Different versions of the same RecObj collection are supported. A typical application of this is to enforce the re-reconstruction of RecObjs with the same Reconstructor but using different calibrations, alignments or parameters to the reconstruction algorithms.

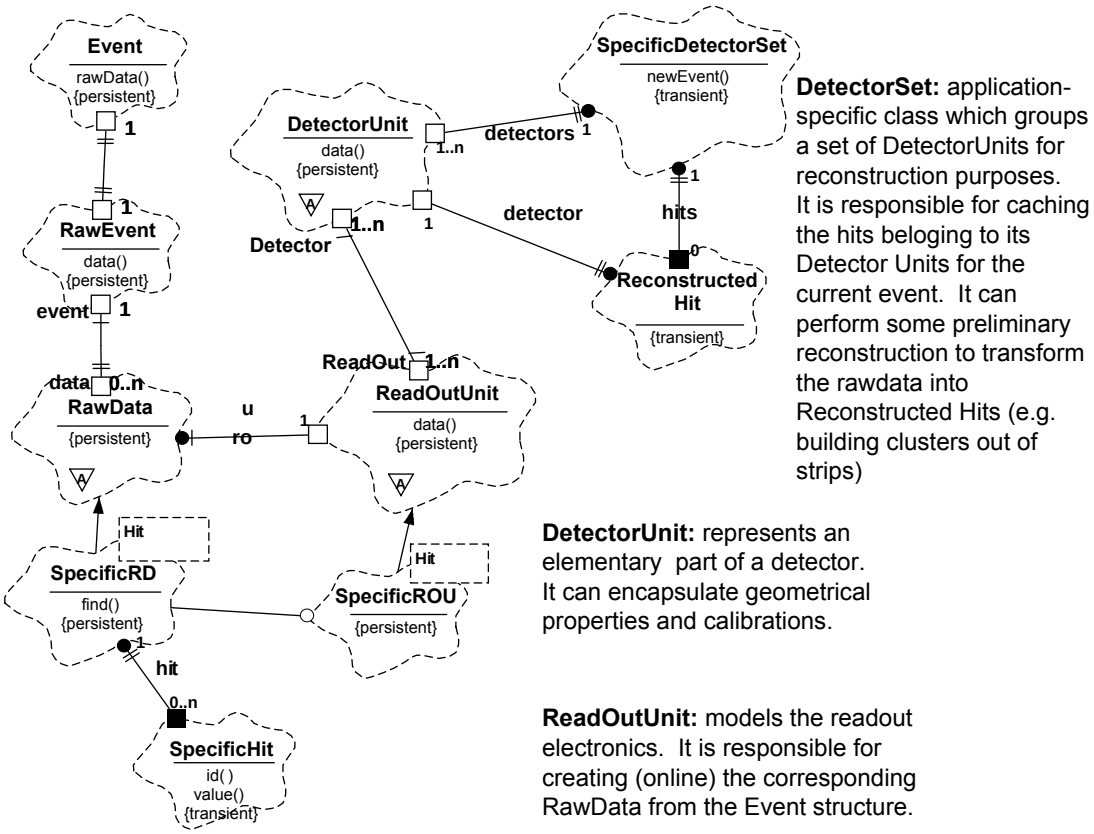


Figure 14-21 The main reconstruction classes and their relationships within ORCA.

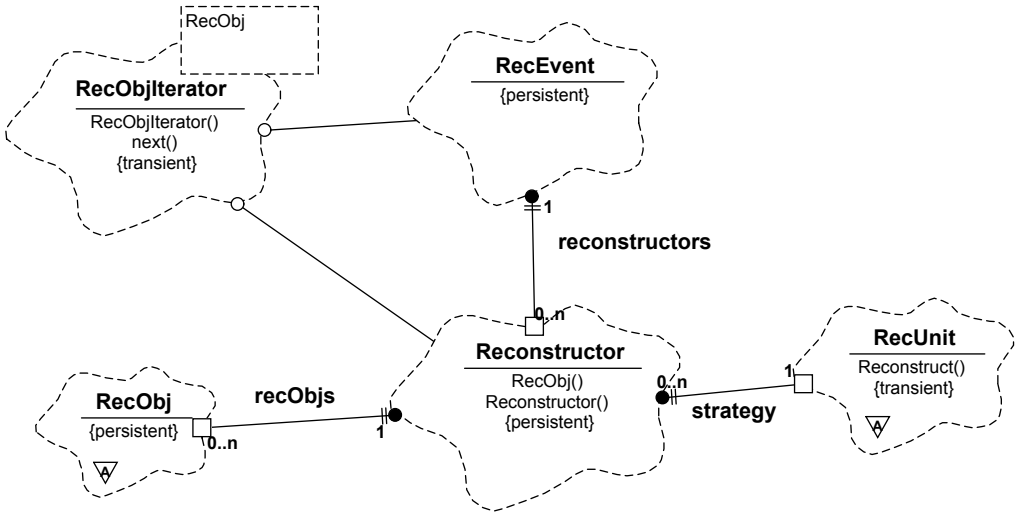


Figure 14-22 RecObjs, their classes and their relationships.

14.3.2.1 Tracker Data and Local Reconstruction

14.3.2.1.1 Tracker Readout

In the pixel system, each barrel module is read out by 16 readout chips. In the end-cap modules, the number of readout chips varies from 2 to 10. Each readout chip reads an array of 52×53 pixels. The readout is column oriented and therefore the chip is divided into 26 double columns 2×53 pixels each. During the readout, all pixels with collected charge above a preset threshold are read and stored in data buffers placed in column peripheries. Only pixels confirmed by the Level-1 Trigger are sent from the data buffers to the Front-End Drivers (FEDs) through analogue fibre links. A maximum of 36 links can be attached to a single FED. The whole readout system will consist of 38 FEDs (32 for the barrel and 6 for the end-caps). The FEDs will digitize the incoming data, build events, check for errors and transmit event packets to the DAQ.

In the microstrip system, signals produced by the silicon strip detectors are initially processed by the front-end APV chips [14-16], each of which provides analogue readout of 128 silicon strips. The APVs amplify the signals and store them in an analogue memory pipeline until the Level-1 Trigger decision is made. If this decision is positive, the data from multiplexed pairs of APVs are transmitted along analogue optical fibre links to the Front-End-Driver (FED) cards [14-24] in the counting room. Each FED can process data from up to 192 APVs. The FEDs digitize the data, optionally perform zero-suppression and then send them to the DAQ system.

14.3.2.1.2 Tracker Zero-suppression and Calibration

The zero-suppression and calibration procedures are different in the pixel and microstrip systems.

For the pixels, zero-suppression, i.e. the rejection of pixels without a significant signal, is performed by the readout chip as described above. The following calibration measurements will be performed:

- measurement of the gains and pedestals (using injected charges),
- measurement of noise and threshold (using injected charges),
- average pixel counting rate/occupancy.

The first two will be carried out about twice per day in the ‘‘off-beam’’ periods. Neither the central DAQ nor the global trigger system will be needed to perform these measurements. The last item is part of the data monitoring procedure executed by the local pixel control system during standard data taking. It does not involve any special procedures and no special triggers are needed. A single calibration sequence generates about 1.2 GB of data. In order to reduce the storage required, the full calibration data will be stored once per week, and subsequently only changes in the values will be stored.

In the microstrip system, during normal physics running (both in p-p and Pb-Pb collisions) the FEDs operate in zero-suppression mode, since this significantly reduces the data volume. In this mode, the FEDs first subtract the pedestal and common-mode offsets from the signal on each strip, and then they search for clusters. Only strips associated to clusters have their pulse height information transmitted to the DAQ. It is proposed to estimate the common-mode offsets in each event from the median of the 128 pedestal-subtracted pulse heights on each APV. The clustering will be done by accepting groups of two or more neighbouring strips, if they contain signals exceeding a threshold of a few times the noise, or isolated strips if they have a signal exceeding a higher threshold. The algorithms proposed for the common-mode subtraction and cluster finding have been evaluated using the ORCA Monte Carlo simulation. In p-p collisions, they are expected to give almost no loss in signal efficiency, whilst a small loss might be expected

in Pb-Pb collisions due to the very high occupancy. To accommodate any unpleasant surprises that actual LHC running conditions may show, the thresholds used for cluster finding are fully programmable. Furthermore, the FED algorithms are implemented in FPGA chips, which allows them (with some design work) to be modified should this prove necessary.

The estimates of the pedestal and noise values for each strip, required by the FED for performing zero-suppression, will be made by the FED VME crate controller. This will be done using a few thousand events of raw data (i.e., not zero-suppressed) taken between fills. The choice of calibration algorithm is still under study. However, preliminary estimates indicate that this calibration should take no more than a few minutes. The calibration constants will be monitored during physics fills by sampling the raw data of a small fraction of events over VME.

14.3.2.1.3 Tracker Data Format and Data Rates

The data rates from the silicon pixel and microstrip trackers have been estimated for both high (10^{34} $cm^{-2}s^{-1}$) and low (2×10^{33} $cm^{-2}s^{-1}$) luminosity pp collisions, assuming a Level-1 Trigger rate of 100 kHz, and also for Pb-Pb collisions, assuming a Level-1 Trigger rate of 7.6 kHz. The data rate is proportional to the pixel (strip) occupancy in events passing the Level-1 Trigger, where the occupancy is defined as the fraction of pixels (strips) accepted by the zero-suppression algorithm of the pixel readout chip (silicon-strip FED). This occupancy has been evaluated using ORCA.

In high luminosity p-p collisions, the tracker occupancy is dominated by the 17.5 superimposed minimum-bias events at each beam crossing. The uncertainty on the occupancy is estimated to be at least ±35% and is dominated by the uncertainty (±30%) on the minimum-bias cross section. However, it also includes contributions of ±12% from the uncertainty in the charged particle multiplicity and transverse momentum spectrum, and of ±12% due to the contribution of the ‘signal’ event which caused the Level-1 Trigger to fire. (Monte Carlo studies of these signal events remain preliminary). At low luminosity, there are five times fewer superimposed minimum-bias events per bunch crossing, so the signal event makes a significant contribution to the occupancy. This increases the relative uncertainty on the low luminosity occupancy to at least ±60%. These figures, described in detail in reference [14-25], are directly translated to uncertainties on the data rates. Additional, subdetector-specific uncertainties are considered below.

In the pixel system, each FED will send to the DAQ a data packet starting with the standard packet header (two 32 bit words) and ending with the standard packet trailer (one 32 bit word). The header includes a bit field identifying uniquely the FED. Each pixel will occupy a 32 bit word. The bit assignment is:

- 6 bit link address within the FED;
- 5 bit ROC address within the link;
- 5 bit double-column address within the ROC;
- 8 bit pixel address within the double-column;
- 8 bit analog signal.

Based on this assignment the FED data volume in bytes is calculated as 4·(3+number-of-pixels).

For the high (low) luminosity pp collisions, the pixel data volume for the barrel FEDs will be around 2.0 (0.6) kB per event. For a 100 kHz trigger rate this corresponds to a maximum data rate of 200 (60) MB/s. For the end-caps the data volume per FED will be smaller, about 1.8 (0.55) kB per event, corresponding to 180 (55) MB/s.

For Pb-Pb collisions the pixel occupancy is about ~14 times larger compared to that in pp collisions at high luminosity. The resulting data volume per barrel FED is about 20 kB if the first barrel layer is used and 12 kB without it (note that in [14-26] the first barrel layer was not considered). The maximum trigger rate given by the 200 MB/s limit will be 10 kHz (17 kHz) for the case with (without) the first layer.

For the microstrip tracker, detailed studies of the occupancies and data rates are presented in [14-25]. In summary, the occupancy in high-luminosity pp collisions rises from 0.6% in the outermost barrel layer to 2.8% in the innermost one. It has little dependence on the z coordinate, except in the outermost disks of the end-cap, where it increases by a factor of two, as a result of soft neutrons emitted from the ECAL end-cap. The uncertainty on the occupancy is dominated by the physics effects mentioned above. Additional contributions arising from uncertainties on the cross-talk between neighbouring strips and time resolution of the electronics have been evaluated and are not significant. However, one potentially large source of systematic uncertainty has yet to be evaluated and therefore represents a possible risk — the occupancy in the outermost disks of the end-cap is dominated by soft neutron emission from the ECAL end-cap. When the neutrons are captured, they yield soft photons, which create signals in the silicon. If this results in a large rate increase one could envisage doubling the number of FEDs in the affected region of the tracker to deal with it.

In the case of Pb-Pb collisions, the occupancy is about eight times higher than in high luminosity p-p collisions, but the corresponding strip tracker data volume is such that the desired Level-1 Trigger rate of 7.6 kHz can be reached [14-25]. Even in the pessimistic scenario, where the true occupancy turns out to be much higher, a minimum Level-1 Trigger rate of 3.4 kHz is guaranteed. At this trigger rate the DAQ can read out all of the tracker strips.

The data rate per FED depends on the tracker region which is read out. A preliminary allocation of strip tracker detector modules to FEDs can be found in reference [14-25] and has been used to estimate the strip tracker FED data rates. The data rate has a small dependence on the data format. In zero-suppression mode, the strip tracker data produced by each FED for each event consists of a 112 B header followed, for each pair of APVs, by a list of the reconstructed clusters they contain. For each cluster the following information is given: the strip address of the first strip in the cluster, the cluster width in strips, and the pulse height on each strip in the cluster. This format [14-24] has been assumed in estimating the data rates.

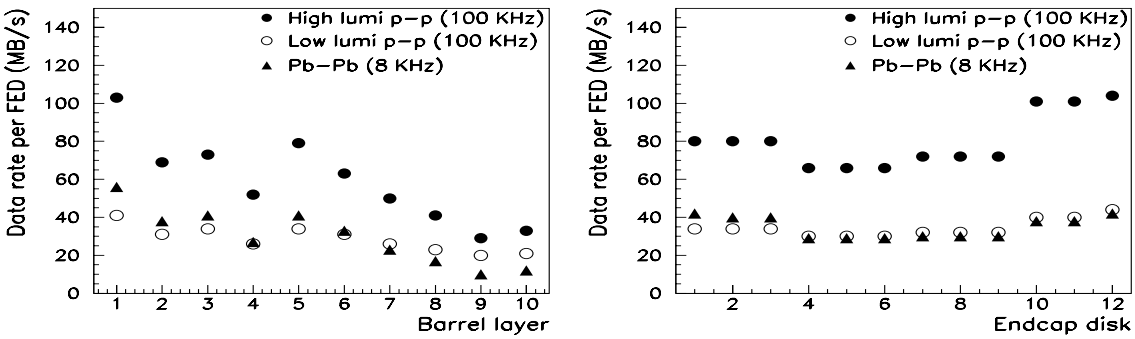


Figure 14-23 Data rate per FED in the strip tracker barrel (left) and end-caps (right).

Figure 14-23 shows the estimated data rates per FED in the barrel and end-cap of the strip tracker, respectively. They take their highest value in high luminosity p-p collisions, where they rise to 104 MB/s in the outermost disk of the end-cap. This rate is comfortably below the 200 MB/s limit that the DAQ can accept. However, the strip tracker requires a total of 440 FEDs, whereas only about 256 of the DAQ switch inputs are available to it. It is therefore proposed to merge the data from pairs of FEDs reading the less occupied parts of the tracker, using the DAQ Front-end Readout Links (FRL). A proposed scheme for doing

this [14-25] reduces the number of switch inputs required to 272, with the maximum data rate per FRL being 138 MB/s.

The data rate is expected to fluctuate from event to event. In p-p collisions, the relative size of these fluctuations is, to a fair approximation, given by the relative fluctuation due to Poisson statistics in the expected number of superimposed minimum-bias events per bunch crossing, i.e. 17.5±4.2 at high luminosity and 3.5±1.9 at low luminosity.

14.3.2.1.4 Tracker Reconstruction

The momentum reconstruction in CMS uses of a 4 T magnetic field provided by a super-conducting solenoid magnet [14-27]. This high magnetic field affects event topologies, by confining low-P_T charged tracks to helical trajectories with small radius. Together with the steeply falling P_T spectrum of charged particles in minimum-bias events, this results in the track density decreasing rapidly with increasing radius. Moreover, interesting physics events are characterized by tracks with P_T exceeding 1 GeV; it is therefore natural to attempt the reconstruction only of the tracks with P_T higher than a given threshold in the GeV range. Momentum reconstruction within the rapidity range of $|\eta|<1.6$ benefits from the full momentum analysing power.

14.3.2.1.5 Cluster Reconstruction

The cluster reconstruction algorithm for the pixel detector starts from a cluster seed, defined as a pixel with a signal to noise ratio S/N > 6. It then adds pixels adjacent to the cluster, if they have S/N > 5, continuing this process until no more adjacent pixels are found. Diagonally adjacent pixels are considered adjacent. Finally the cluster is retained if its total charge has S/N > 10.1. The same algorithm is applied to the barrel and forward pixel detectors.

The position of selected clusters is estimated independently in both dimensions. The position assignment is based on the charge widths W, which are the projections onto the two directions of the area where the charge is collected on the detector surface. In the transverse direction, both the track inclination and the Lorentz shift contribute to the charge sharing. For clusters of two pixels the hit position is assigned as

$$x_{cluster} = x_{center} + \frac{Q_2 - Q_1}{2(Q_1 + Q_2)} \cdot W$$

where x_{center} is the geometrical centre of the cluster and Q_1 and Q_2 are the charges deposited in the first and last pixel hit respectively. This formula can be generalised for clusters larger than two pixels [14-28].

A precise charge width calculation is performed if the impact angles of the particle on the detector are known from the partial track reconstruction, otherwise the track impact angles are estimated from the polar and azimuthal angles of the hit modules with all tracks assumed to come from the nominal interaction point.

The hit resolution depends also on the cluster size and the track impact angle. The pixel cluster size in the transverse direction is mainly determined by the Lorentz shift and is usually 2 over the whole rapidity range (in 82% of cases for high transverse momentum muon tracks). The cluster size in the longitudinal coordinate is determined by the track impact angle and ranges from 1 to 6 pixels depending on the rapidity. The error on the reconstructed position is estimated from the spatial displacement between simulated and reconstructed hits (residuals) and error parameterization is performed as a function of the cluster size and the rapidity. An additional and more precise error parameterization is performed when the track im-

pact angles are available from the partial or complete track reconstruction. A detailed description of the error and position assignment of pixel hits can be found in [14-28].

The reconstruction inefficiency is defined as the fraction of simulated hits which do not have any associated reconstructed hit. For the pixel detectors this is about 0.5%. The fraction of reconstructed hits which are not associated to any simulated hit (ghost hits) is less than 0.01%. Both the reconstruction efficiency and the ghost rate quoted here do not take into account readout inefficiencies.

Figure 14-24 shows the residuals and the pulls of the barrel detector hits. Typical pixel resolutions are around 10 μm and the pulls are close to 1.

In the microstrip subsystem, the cluster reconstruction algorithm searches for strips with charge significance satisfying $S^{(i)}/\overline{N}>3$, where $\overline{N}$ is the average strip noise and $S^{(i)}$ the strip signal, both expressed in ADC counts. Once at least one such strip is found, all adjacent strips with $S^{(j)}/\overline{N}>2$ are added to the cluster. When no adjacent strips above this threshold remain, the cluster charge $S^{cluster}$ is computed as the sum of the pulse heights over the accepted strips. If the condition $S^{cluster}/N^{cluster}> 5$ is satisfied, the cluster is retained. The same algorithm is applied to both barrel and forward microstrip detectors. For isolated strips, which do not have adjacent strips over threshold, a stronger cut of $S^{(i)}/\overline{N}>5$ is imposed.

If the cluster size is less than 4, the cluster finder computes the cluster coordinate from the average position of all the strips assigned to the cluster, weighted by their pulse heights (this is the “centroid algorithm”). For wider clusters, Landau fluctuations spoil the resolution of this method, and only the two strips at the edges of the cluster are used. The expected error on the measured position is estimated by looking at the strip coverage by the simulated hits, using a simple parameterization. For double-sided (twin) detectors the stereo coordinates are calculated as well as their covariance term with the precise coordinate.

The reconstruction inefficiency is defined as the fraction of simulated hits, which do not result in reconstructed hits. This can be due to negative noise fluctuations, or to Landau fluctuations. In all the silicon microstrip detectors, it is estimated to be less than 0.5%.

The fraction of ghost reconstructed hits, which are not associated to any simulated hit, but are simply due to noise fluctuations, is low because of the tight cut applied on the cluster significance. The fraction is similar for barrel and end-cap detectors, and is estimated to be less than 0.1%.

Figure 14-24 shows, as an example, the residuals in silicon microstrip barrel detectors. The residual is defined as the distance, in μm, between the simulated and the reconstructed hit positions. The resolutions depend on whether the detectors have 512 or 768 strips, and on the detector shape and pitch. The widths of the pulls are always close to 1.

Several effects are not yet correctly simulated in the present software. For example, the cluster inefficiency just reported is only “algorithmic”, and does not take into account the number of dead strips which can be accepted when testing the modules (up to 2%). In the presence of dead strips, the cluster finder should be improved to handle holes in clusters and to correctly calculate a clusters total signal and noise.

14.3.2.2 ECAL Data and Local Reconstruction

14.3.2.2.1 ECAL Crystal Readout

The shape of the signals coming out of the front-end amplifiers attached to the crystal photodetectors (APDs or VPTs) is shown in Figure 14-25. This signal is sampled and digitized at 40MHz. After the first

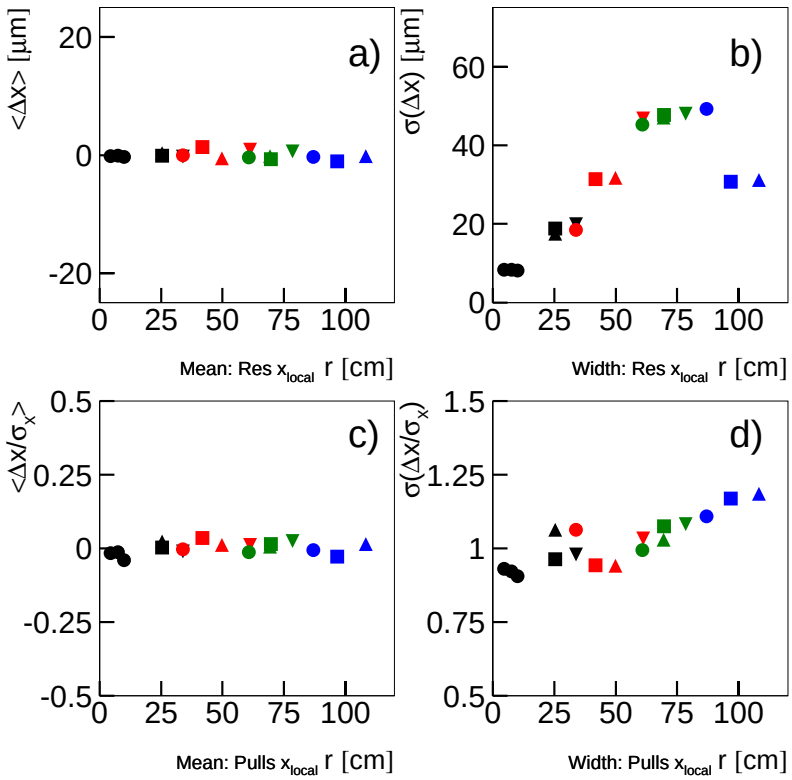


Figure 14-24 Hit residuals for silicon detector in the barrel tracker. Each point represents a different detector type. (a) Mean value of the residuals as a function of layer radius. The three points at r<12 cm are the pixel barrel detectors, the points at higher radius are microstrips (b) RMS of the residual distribution. (c) and (d) are the same as (a) and (b) for the pulls.

preamplifier stage there are four parallel buffer and sample-and-hold units each with a different gain. The output of the appropriate gain unit is selected, each 25ns, to be passed to the sampling ADC. The result is a floating-point digital representation comprising 14-bits: 12-bits from the ADC and 2-bits specifying the gain range. The trigger primitives are generated on the detector, and sent off detector on synchronous optical links to the Trigger Concentrator Cards, from where they pass to the Level-1 Trigger. The data is also digitally buffered, on the detector, awaiting the Level-1 Trigger decision. On receipt of a Level-1 accept, 10 time-samples for each channel are sent off the detector on optical links, one link for each 5×5 unit of crystals, to the Data Concentrator Cards. The 5×5 crystal units correspond to trigger towers in the barrel. In the end-cap they correspond to the constructional units (super-crystals).

14.3.2.2.2 Selective Readout Processor

A Selective Readout Processor makes decisions on the readout state to be assigned to each 5×5 crystal unit. Since information for the whole ECAL is present the decision can be based on information such as, for example, the transverse energy in a neighbouring unit. The Selective Readout Processor makes the decision based on two threshold bits sent to it by the Trigger Concentrator Cards, which generate the bits from the trigger primitives. The readout state is transmitted to the Data Concentrator Cards in two bits for each 5x5 channel unit. These bits define the *readout state* for each 5×5 unit, specifying how it should be read out. Suggested readout states (which have been utilised in simulation studies mentioned below) are:

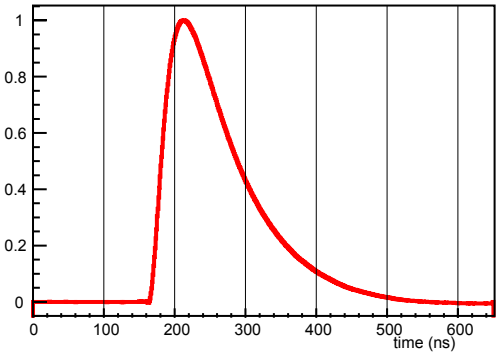


Figure 14-25 Pulse shape from preamplifiers attached to photodetectors for ECAL crystal readout.

00: Header; Trailer;

01: Header; Data from individual crystals data passing zero-suppression; Trailer;

10: Header; Data from all individual crystals; Trailer;

11: reserved for test, debug;

All trigger primitives data are always read out.

The total data volume from the ECAL, including the various overheads, headers and trailers etc., but not including the preshower detector data, is given by:

$$\text{ECAL data volume} = 37.75\text{KB} + (N \times 24)\ \text{B}$$

where N is total number of individual crystals read out.

14.3.2.2.3 Selective Readout Algorithms

A simple way to reduce the data volume is to discard data from channels where the recorded energy is below some defined threshold — zero-suppression. The data volume for the ECAL is to be limited to ~ 100KB. This requires that only about 5% of the channels are read out. To achieve this using zero-suppression alone, particularly at high luminosity, when occupancy is dominated by pile-up activity, requires zero-suppression thresholds (barrel ~100MeV, end-cap ~400MeV) which result in a noticeable (few percent) non-linearity in the energy scale of reconstructed showers — since for showers of higher energy more channels go above threshold. This complication can be avoided using the functionality provided by the Selective Readout Processor, either by reading out all crystals in the vicinity of showers of interest for precision measurement, or by discarding a sufficient number of channels in areas of little interest so that the remaining crystals can be read out with a very low zero-suppression threshold.

Two basic algorithms, following the two strategies just described, have been investigated:

1. The data reduction is achieved by zero-suppression only. The selective readout facilities are used to control the application of the zero-suppression. Crystals in 5×5 units having a high transverse energy, or neighbouring such units, are read out without zero-suppression. All the individual crystals in a 5×5 unit are read out without zero-suppression if either the unit has a transverse energy

greater than a first (low) threshold, or a neighbouring unit has an transverse energy above a second (higher) threshold.

2. Some data reduction is obtained by applying zero-suppression at a fairly modest threshold to all channels. Further data reduction is obtained by not reading out any individual crystals in some 5×5 units. The individual crystals in a 5×5 unit are read out if either the unit has a transverse energy greater than a first (low) threshold, or a neighbouring unit has an transverse energy above a second (higher) threshold.

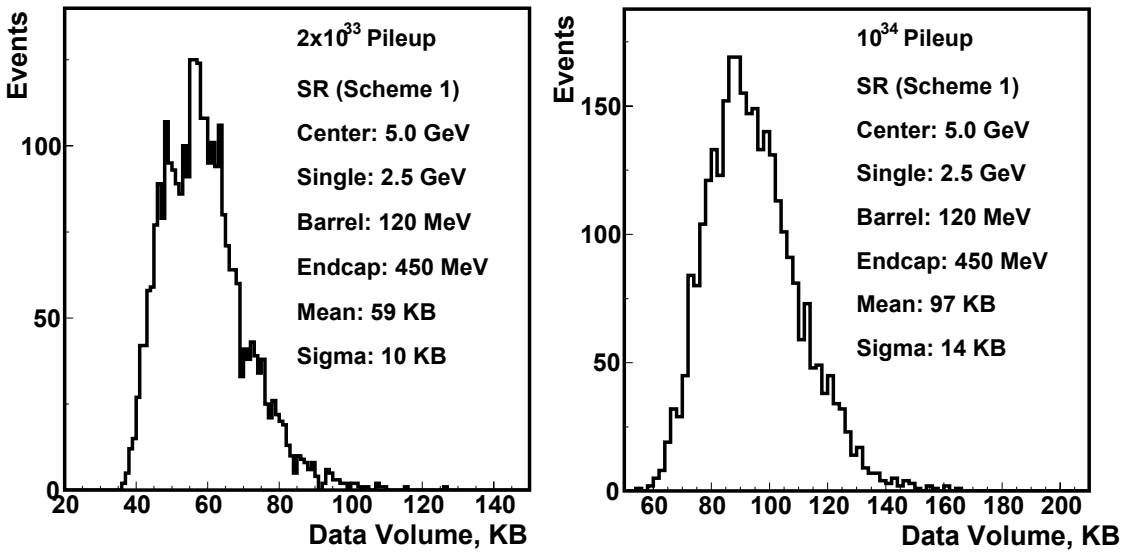


Figure 14-26 Distribution of ECAL data volume for jet events at a) $2x10^{33}$ $cm^{-2}s^{-1}$, and b) 10^{34} $cm^{-2}s^{-1}$.

Using these algorithms the ECAL data volume can be reduced to the 100 kB target with negligible impact on physics. Figure 14-26 shows the distribution of ECAL data volume for jet events when an algorithm of the first type listed above is used. The simulation was made using fully simulated high-E_T jet events with pile-up corresponding to a) 2×10^{33} $cm^{-2}s^{-1}$, and b) 10^{34} $cm^{-2}s^{-1}$. At high luminosity, for energies greater than about twice the electronics noise width, the occupancy is completely dominated by pile-up. The simulation was done with a model of the readout, where, in the end-caps, the units are trigger towers rather than super-crystals. The parameters of the algorithm used in this example are such that all channels belonging to trigger towers having $E_T > 2.5$ GeV are read out, as well as all channels in trigger towers neighbouring towers with $E_T > 5$ GeV. The remaining towers are readout with a zero-suppression threshold of 120 MeV in the barrel and 450 MeV in the end-cap.

The algorithms described give some freedom and flexibility to tune the ECAL data volume. Figure 14-27 shows how the mean ECAL data volume for high-E_T jet events at 10^{34} $cm^{-2}s^{-1}$ varies as a function of the zero-suppression threshold for three different pairs of values for the thresholds at which all crystals are read out from a tower (‘Single’) or from all neighbouring towers (‘Centre’).

14.3.2.2.4 Preshower Data

The CMS ECAL also includes the preshower detectors, on the end-caps, comprising 4304×32 silicon strips. Time samples, at 40 MHz, are stored in an analogue buffer on the detector. On receipt of a Level-1 Trigger three time samples per strip are digitized at 20 MHz, using the same 12-bit ADC as for the ECAL crystals, and transferred, by optical links, off the detector to the upper level readout system. A total of

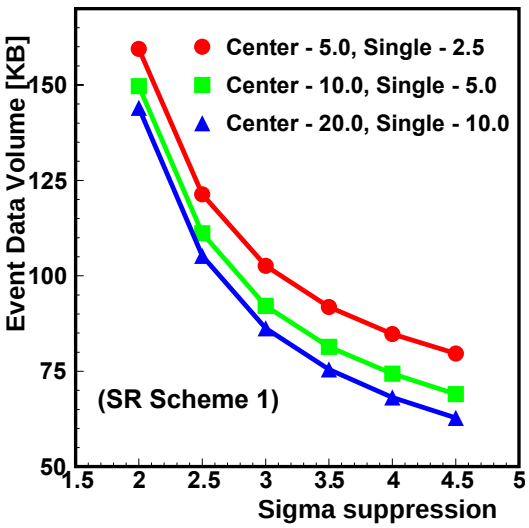


Figure 14-27 Mean ECAL data volume for high E_T jet events at 10^{34} $cm^{-2}s^{-1}$ as a function of the zero-suppression threshold. The three curves correspond to different tower thresholds for removal of the zero-suppression (see text).

4304×148 bytes are transferred. In the upper level readout system the data is zero-suppressed at a level much greater than the electronics noise width — $3\sigma_{noise}$ is suggested (this is about equal to 60% of the signal given by a minimum ionizing particle). Average occupancy at high luminosity is about 2%. Thus a little more than 10 kB of data will be finally read out from the preshower detector.

14.3.2.2.5 Reconstruction of the Energy from the Time Frames

The first step in reconstructing the energy deposited in a crystal from the time frames of ten time samples is the expansion of the floating point representation by multiplying the 12-bit ‘mantissa’ by the appropriate gain specified by the 2-bit ‘exponent’. At this point, also, account must be taken of the difference in pedestal between the ranges. The energy deposited in the crystal is then reconstructed by a scalar product between a vector of time samples and a vector of weights, to give a single value proportional to the deposited energy. The reconstruction currently used with simulated data uses only 8 of the time samples — 3 before the pulse, and 5 in the pulse. The precise details of the algorithm to be used and the methods of extracting the optimum weights are currently under study, and their fine tuning will require test beam data taken with the final front-end electronics.

The time of the signal, as well as the energy, can be reconstructed from the time frames. This can probably best be done using another vector of weights, optimized for extracting the pulse time rather than its magnitude. In the present simulation work a simple χ^2 is constructed to measure how well the observed pulse conforms to the hypothesis that the pulse was generated by energy deposited in the bunch crossing of the trigger event. Currently the only use that is made of this χ^2 is to suppress out-of-time clusters when making ECAL isolation cuts.

The reconstruction of the preshower signal is both simpler (there is a single gain range and only three time samples) and less critical, but it is likely to use similar techniques.

14.3.2.3 HCAL Data and Local Reconstruction

14.3.2.3.1 HCAL Readout

The data output by the QIE are in a pseudo-floating point format, with 5 bits of mantissa, 2 bits of range, and two bits for capacitor identification. The pedestal is set in the sixth ADC bin, and is assumed to have been measured to high precision during the abort gap.

The QIE data is processed to make trigger primitives for Level-1 and held, to be sent to the DAQ on receipt of a Level-1 accept. To produce the Level-1 Trigger primitives, the time samples are first converted from energy to transverse energy. If there is more than one depth segment (excluding the HO), as is the case for all towers beyond tower 15, the depth segments are summed. In the HF, the 2 readouts (long and short fibers) are summed, and the HF towers are given a coarser segmentation by ganging together 2 in ϕ by 3 in η of the full readout segmentation. The signal shape from the HB/HE/HO is longer than the time between bunch crossings, and several time samples are used to estimate the energy.

The algorithm used to extract the transverse energy from the time samples in the HB/HE/HO is simple. The amplitudes from two consecutive beam crossings are summed and the pedestal subtracted. The amplitude is then multiplied by a calibration factor (1.11) which includes a correction for the fraction of the signal outside the two time buckets. In the simulation we assume 0.68 MeV/pe in the HB, 0.45 MeV/pe in the HE, and 0.25 MeV/pe in the HO, with sampling fractions of 220 in the HB, and 147 in the HE. These multiplicative corrections will be implemented using a programmable look-up table in hardware.

The pattern of energy deposits in the different time samples is checked to verify its association with the triggering crossing. If the signal calculated from the time samples as described above, assuming the energy comes from the current crossing, is bigger than that calculated assuming the energy comes from the previous or subsequent crossing, the energy is kept. Otherwise, it is zeroed. The trigger primitives are then sent, in a compressed format with a LSB of 1 GeV, to the Level-1 regional calorimeter trigger.

The algorithm in the HF is very simple. The energy in one time sample is converted to GeV after pedestal subtraction. Since the pulse is much shorter than one beam crossing, there is no problem with out-of-time energy.

14.3.2.3.2 HCAL Zero-suppression and Data Rates

The HCAL readout contains a header and trailer, trigger primitive data, and precision readout (DAQ path) data. The header/trailer information consists of approximately 160 service bits per Data Concentrator Card (DCC). There are 32 DCC's. An additional few words of service bits (a few $\times$16 bits) is also needed. A total of 4320 channels of trigger primitive data, 9 bits per channel, are read out. The full-precision raw data for the DAQ path is checked to see if it is above a zero-suppression threshold. Currently, this threshold is assumed to be 500 MeV in energy in the HB/HE/HF and 150 MeV in the HO. If the energy is above threshold, ten time samples of QIE data are read out, otherwise a single time sample is read out.

There are 7560 channels in the HB/HE/HF and 2160 in the HO. When address bits and other service information are included there are 16 bits per time sample. Table 14-4 shows an example of a possible data format.

The HCAL data volume per readout is therefore given by

$$N_{DCC}H + L + D + O_{HBEF}N_{HBEF}(P + NR) + O_{HO}N_{HO}(P + A + NR)$$

Table 14-4 Tentative HCAL Data Format.

word type	Byte 3	Byte 2	Byte 1	Byte 0
HEADER	EV #23-16	EV # 15-8	Zeros	EV #7-0
EXT HEADER 2	No. Trig Towers	HTR card #	Spare + BX#11-8	BX#7-0
EXT HEADER 3	Spare	Spare	Trigger type	ORBIT #
Trig Towers 24-21	Tower 24	Tower 23	Tower 22	Tower 21
Trig Towers 20-17				
Trig Towers 16-13				
Trig Towers 12-9				
Trig Towers 8-5				
Trig Towers 4-1	Tower 4	Tower 3	Tower 2	Tower 1
Trigger Muon bits	Muon bits 24-1			
DAQ Mask	Mask for each of the 24 Channels of DAQ Data			
DAQ Data	Data		Data	
DAQ Data, cont	Data		Data	
FE Link Errors				
CapID Errors				
Other Errors				
TRAILER	EV # 7-0	zeros		

where N_{DCC} is the number of DCC's, H is the header and trailer information (=2×16+160=192 bits), L is the trigger primitive data (9×4320=38880 bits), D is the 7 bits of QIE data per HB/HE/HF channel (8×7560=60480) which is read out regardless of whether the channel is over threshold or not, P is the number of bits used for the pedestal value (8), R is the number of bits used for the ADC and service words (16), N is the number of time samples (10), N_{HBEF} is the number of HB/HE/HF readout channels (7560), N_{HO} is the number of HO readout channels (2160), O_{HBEF} is the probability for an HB/HE/HF to be over the zero-suppression threshold and O_{HO} is the same probability for the HO.

Figure 14-28 shows the occupancy as a function of the zero-suppression threshold in $E_T/(\Delta\eta\times\Delta\phi)$ normalized to the size of a central HB tower, in minimum-bias events at high luminosity for the shorter of the two possible HCAL signal shapes. It is currently planned to run with a readout threshold of E_T=500 MeV/(0.087×0.087), which gives an occupancy of about 10% for the HB/HE/HF, roughly independent of η. The HO will have a lower threshold, around 150 MeV, and has an occupancy there of about 5%. The total HCAL data volume at high luminosity will be

$$32 \times 192 + 38880 + 60480 + (0.1)(7560)(8 + 10 \times 16) + (0.05)(2160)(8 + 10 \times 16) \; = \; 245\text{kb}$$

Since the Level-1 Trigger operates at up to 100 kHz, and the total data rate is 245 kb per Level-1 Trigger, the total required readout throughput is 24 Gb/s. This data rate is consistent with the hardware bandwidth: each DCC sustains 200 MB/s, i.e. the 32 DCC's sustain 6400 MB/s or 57 Gb/s.

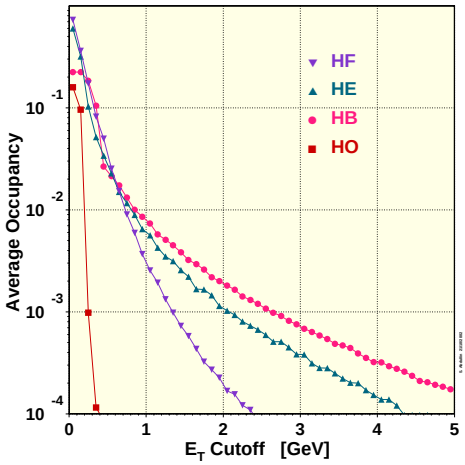


Figure 14-28 Occupancy in the HB, HE, HO, and HF Hadron calorimeters versus cut on E_T / ($\Delta\eta\Delta\phi$), normalized to the size of a central HB tower.

14.3.2.4 Reconstruction of the Energy from the Time Frames

The time samples are sent to the DAQ after a Level-1 accept. In the HLT the same algorithm is used to convert time samples to energy as was used to construct the Level-1 Trigger primitives, except that the conversion is done in energy, not transverse energy, and the DAQ data has the full segmentation of the calorimeter. Because in the HB/HE/HO two time samples are used in the calculation, the effective electronic noise is about 240 MeV. When this is combined with the shot noise due to the finite number of photoelectrons, the total noise is about 330 MeV.

14.3.2.5 Muon Data and Local Reconstruction

14.3.2.5.1 Drift-Tube Readout

The entire drift-tube system contains nearly 200,000 electronic channels. The front-end electronics of the barrel muon system amplifies and discriminates the pulses coming from the drift tube wires. The resulting signals in a single 30° azimuthal sector in one wheel of one station are collected by 6 to 9 Readout Board (ROBs), each of which contains four 32-channel TDC circuits. A TDC channel provides an output whenever there is a hit on the corresponding wire within a trigger window that can be programmed to accommodate the maximum drift time (~400 ns). The dynamic range of each TDC is 21-bits, and the generated data occupy 4 bytes. The output of all ROBs are collected by 60 Readout Server Masters (ROSMs), one per sector, which in turn transmit data via optical links to five Front-End Drivers.

In order to estimate the average event size at design luminosity, it is assumed that the rate of tracks crossing the drift-tubes from muons and punch-through does not exceed 1 Hz/cm^2, and that the rate of uncorrelated hits from neutron-induced electrons does not exceed 10 Hz/cm^2. In this case, the estimated average event size is 9 kB, including the header and trailer overhead from a preliminary data format. For a luminosity of 10^{34} cm^{-2}s^{-1} and a Level-1 rate of 100 kHz, the overall bandwidth coming from the drift-tube system is approximately 1000 MB/s.

14.3.2.5.2 Cathode Strip Chamber Readout

The entire cathode strip chamber system contains about 500,000 electronic channels. The pulses from the cathode strips, after amplification and shaping, are sampled every 50 ns and stored in a switched capacitor array (SCA) pipeline, as illustrated by Figure 14-29. Part of the signal is also discriminated and sent into a comparator network for the Level-1 Trigger. The capacitor cells corresponding to a CSC pulse are reserved for possible digitization if a Local Charged Track (LCT) trigger primitive is generated from the comparator signals arising in the same (or neighbouring) cathode front-end card, otherwise the cells are returned to the ring buffer by the SCA controller circuitry. The capacitor cells are reserved in groups of 8, with at least the first two cells of a group of 8 sampling the pedestal before the CSC pulse. The maximum of the CSC pulse typically lies between the fourth and fifth time samples if only one group of 8 cells is reserved. If a Level-1 accept is eventually generated, and an LCT trigger occurred within one bunch crossing of it, then the capacitor cells that were blocked off are digitized by 12-bit ADCs. All 96 channels (16 strips by 6 layers) of a cathode front-end board are digitized for typically 8 or 16 time samples. This coupling of the CSC data acquisition to the LCT trigger, which effectively selects a region of interest, is necessary in order to reduce the overall bandwidth from the system.

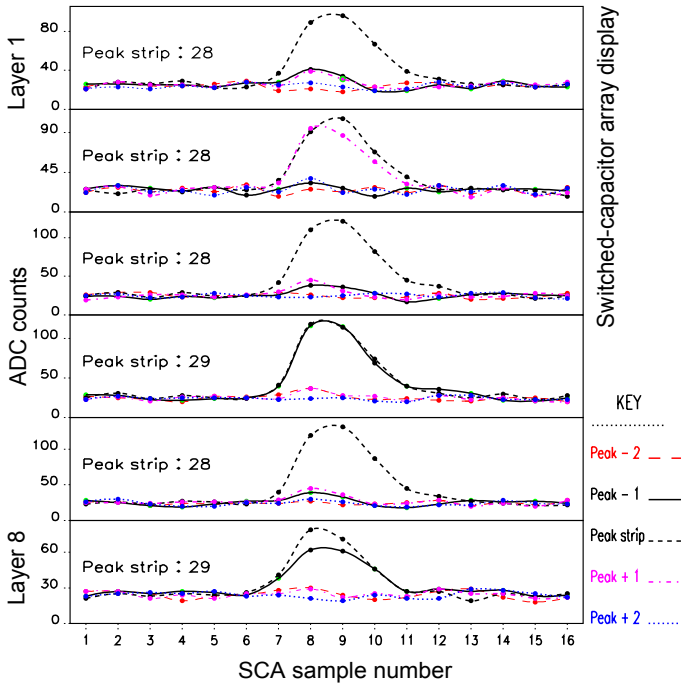


Figure 14-29 Display of the switched capacitor array data for a muon that crosses all 6 layers of a ME 234/2 cathode strip chamber.

The amount of data collected by a CSC Data Acquisition Motherboard (DMB) for a single cathode strip chamber (assuming that an average of 1.5 cathode front-end boards are read out) is 3 kB (5.4 kB) for 8 (16) digitized time samples, including comparator and discriminated wire data as well as header and trailer words. Eight time samples are the minimum required for pulse reconstruction in the presence of pile-up, but 16 time samples are expected to be read out initially during low luminosity operation. The output from 15 DMBs, corresponding to a 20° azimuthal slice of the end-cap, are sent via optical links to a single CSC Front-End Driver (FED), which buffers and transmits the data over the SLINK-64 protocol. Several Data Concentrator Cards (DCCs) combine the output from the 36 FEDs.

A single muon that traverses four CSC stations in the end-cap generates 12 kB (22 kB) for 8 (16) time samples, assuming no pile-up from other muons or punch-through. However, the average event size depends on the occupancy of the chambers as well as the fraction of triggers that contain muons. From simulation, and assuming that half of all triggered events contain a muon, it is expected that the average CSC event size is 10 kB. At low luminosity, this assumes that 16 time samples are read out, but to maintain the same event size at high luminosity requires a reduction to 8 time samples as well as a tighter LCT requirement (a coincidence of the anode and cathode LCT, rather than just cathode). At a luminosity of 2×10^{33} $cm^{-2}s^{-1}$, the overall CSC bandwidth is approximately 500 MB/s with an Level-1 rate of 50 kHz, and at the design luminosity of 10^{34} $cm^{-2}s^{-1}$, it is approximately 1000 MB/s for a Level-1 rate of 100 kHz.

14.3.2.5.3 Resistive Plate Chamber Readout

The Resistive Plate Chamber muon system contains nearly 200,000 electronic channels. The pulses coming from the RPC strips are amplified and discriminated by an 8-channel ASIC on the front-end boards. All single strip data used as input to the RPC Pattern Comparator logic used by the Level-1 Trigger are read by the readout system. An “LMUX” zero-suppression scheme used by the link subsystem, described in [14-35], guarantees that only non-empty channels are read. The data collected by the readout system are delivered to standard Readout Dual-Port Memory (RDPM) modules. The occupancy of the RPC system is dominated by noise, and leads to an average event size of 600 bytes.

14.3.2.6 Track Segment Reconstruction

14.3.2.6.1 Track Segment Reconstruction in the Drift Tube System

The reconstructed position of a muon hit in a drift cell is determined from the corresponding TDC measurement and the effective drift velocity (also used in the creation of the hits in the simulation, parameterized as a function of the track direction and the magnetic field), which must be known to an accuracy of about 1% in order to obtain a position resolution of about 250 μm. The reconstruction is performed in two steps: an average drift velocity is used for a first determination of the reconstructed hit position and direction. Next, these parameters are used to determine the correct effective drift velocity to refine the hit reconstruction. The final residuals of the simulated reconstructed hits in the r-ϕ bending plane and in the r-z non-bending plane have widths of 285 μm and 340 μm, respectively, for muons with $10<P_T<100$ GeV/c uniformly distributed throughout the barrel detector (and thus averaging over all inclination angles, cell positions, and magnetic fields). It should be noted that in the r-ϕ plane tracks originating from prompt muons have angles up to 15° with respect to the direction normal to the chamber plane, while in the r-z plane, the inclination angle extends to about 50°, depending on η.

Track segments are created by performing simple least-squares fits of straight lines in two-dimensions through the reconstructed hits in the superlayers of a chamber set. This is done separately in r-ϕ and r-z, but the 2-D segments are then associated into 3-D segments. Up to 12 points per track segment are included, corresponding to the maximum number of the chamber measurement layers in a station. Hit sharing between segments is not allowed, and left-right ambiguities are resolved on the basis of a χ^2 criterion, with the additional constraint that the track originates from the interaction region for the r-z orientation. The position and direction of each track segment (after the reconstructed hit refinement) are input to the Kalman filter algorithm for the track reconstruction used by the HLT. The residuals of the track segment position in the r-ϕ plane are considerably improved with respect to those obtained from a single drift-tube layer, as shown in Figure 14-30a, which shows the overall simulated residual distribution from chambers in all DT wheels and stations. The central width of the distribution is 109 μm. The corresponding residuals of the track segment direction angle in the r-ϕ plane (with up to 8 reconstructed hits and a 26 cm lever

arm) are shown in Figure 14-30b. The direction angle is defined with respect to the direction normal to the chamber plane (z). This distribution has a central width of 0.9 mrad. In the r-z (non-bending) plane, the direction angle residuals vary from 9–13 mrad for $|\eta|<0.9$. For higher η values, corresponding to tracks in the outer wheels of the MB1 and MB2 stations (see Fig. 14-10), the resolution in the non-bending plane worsens to 35 mrad.

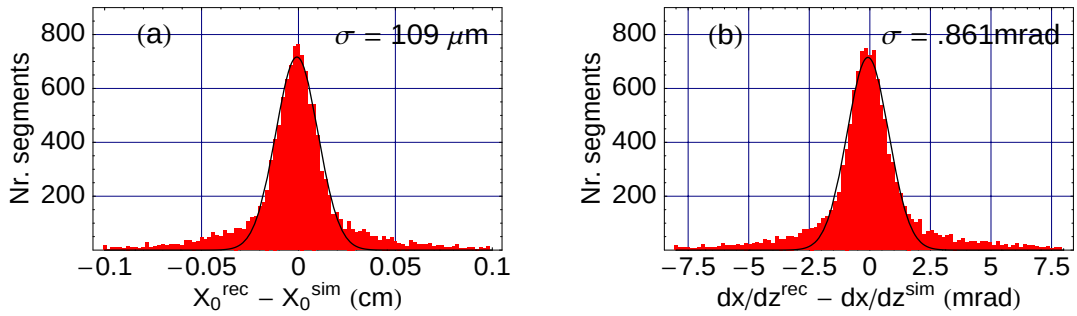


Figure 14-30 Residuals in the r-ϕ plane of the DT track segment position (a), and the DT track segment direction angle (b), defined with respect to the direction normal to the chamber.

The corresponding position and direction errors are also determined for each track segment in order to be used by the Kalman filter algorithm. In all cases, Gaussian fits to the central part of this pull distribution have zero mean and approximately unit width. However, long tails do exist, partially due to difficult regions at high η where a strong magnetic fringe field is present.

14.3.2.6.2 Cluster Finding and Track Segment Reconstruction in the Cathode Strip Chambers

The charge from a chamber hit is distributed across 3 to 5 cathode strips. In order to precisely reconstruct the position of the hit, one must accurately determine the pulse height on each strip from the SCA information and perform a fit to the expected spatial shape of the charge distribution. The measurement of the charge of each strip is taken from the fifth SCA sample, which is constructed to be taken at the peak of the time evolution of the signal that triggered the digitization. Baseline shifts are accounted for by subtracting from each pulse the average of the first two SCA samples, taken before the signal arrived. No correction is made for signal pile-up.

Strip clusters are reconstructed starting from the strip with the maximum charge, and including the two neighbouring strips. The so-called “Gatti function” [14-23] is used to fit the spatial distribution of the charge over the 3-strip cluster (and is also used in the creation of the strip charge in the simulation). With this reconstruction, the fitted width of the central part of the residual distribution is about 240 μm in the simulation of CSC layers from ME2, ME3, and ME4, as shown in Figure 14-31a. (It is about 120 μm for ME 1/1.) The pull distribution of the reconstructed strip positions for the same chambers is shown in Figure 14-31b.

Reconstructed hits are created at each intersection of a cathode strip cluster with each wire group that has enough charge to be digitized. The bunch crossing assignment between the anode and cathode clusters must agree to within two bunch crossings. No attempt is made to cluster neighbouring wire groups or to suppress ghost combinations.

Track segments are created by performing simple χ^2 fits of straight lines through the reconstructed hits in 3-D. Due to the small lever arm of the hits within a chamber, a fast straight line fit neglecting the small

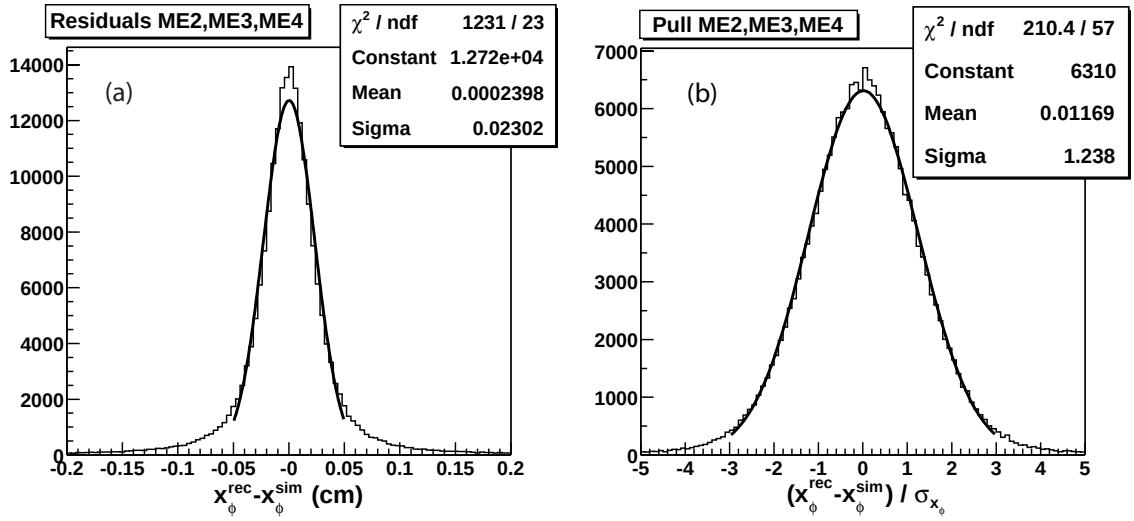


Figure 14-31 Simulated residuals (a) and pull distribution (b) of the reconstructed strip positions in the CSC layers from ME2, ME3 and ME4.

curvature effect of the residual magnetic field inside the chambers is accurate enough for the Kalman fit that will be performed using the segments. The procedure starts with the leftmost hit in the first layer of a chamber and the rightmost hit of the sixth layer. One hit is added per layer, and the line fit is redone after the addition. A hit is replaced if a better match is found. Hits must match within 2.5 mm in ϕ, and $\chi^2<100$ is required. A track segment must contain at least 4 of the 6 possible hits. Overall, the algorithm is 98% efficient for 50 GeV muons, with the inefficiency mainly arising from showers. The direction resolution of the 3-D track segment varies from 7 to 11 mrad in ϕ and from 50 to 120 mrad in θ at 50 GeV, depending on the CSC chamber type.

14.3.2.6.3 Cluster Reconstruction in the Resistive Plate Chambers

The clustering procedure applied to the strip signals from the RPC chambers consists of grouping all adjacent strips above a threshold. Once all groups are formed, the reconstructed position of each cluster is taken to be the centre-of-gravity of the total area covered by the cluster of strips. Thus, in the barrel, it is simply the centre of the rectangular area of strips; in the end-cap, it is the centre of a trapezoidal-shaped region. The error on the reconstructed position along each axis is calculated assuming a uniform hit probability across the cluster area; in other words, it is $1/\sqrt{12}$ times the length of the corresponding side.

14.4 Global Reconstruction

14.4.1 Track Reconstruction

Due to the complexity of the CMS tracker, with its tens of thousands of detector modules, tens of millions of channels, and tens of thousands of hits per bunch crossing, it is not obvious that there is a single optimal track reconstruction algorithm in all circumstances. It is more likely that there should be several specific algorithms, each of them optimized for a specific task within the complex physics at CMS.

Therefore a flexible framework for developing and evaluating track reconstruction algorithms is used [14-29].

The CMS track reconstruction framework is based on the Kalman Filter formalism [14-30], but not restricted to it. More advanced algorithms, like the Deterministic Annealing Filter or the Multi Track Filter have been successfully implemented as well [14-29].

The mathematical complexity of track reconstruction and fitting limits the number of developers. Usually the algebra involved is localized in a few places and is largely independent of the various types of track reconstruction algorithms. Therefore the development of track reconstruction algorithms can be made easier and more accessible if the algebra can be separated from the logic of the algorithm. In the case of the Kalman filter formalism, the basic operations are propagation of a trajectory state to some surface, and updating of a trajectory state with information from a measurement. They are encapsulated in two basic components, the Propagator and the Updator.

These components, in collaboration with the detector geometry components, allow for simple and flexible implementation of the different stages of the track reconstruction described below.

14.4.1.1 Influence of Tracker Material on Track Reconstruction

Track reconstruction is based upon the equations of motion of a charged particle in a magnetic field. In the standard Kalman filter [14-30] they determine the evolution of the state vector and its covariance matrix. This simple model has to be modified in the presence of matter: deterministic effects are included in the track model, stochastic processes result in additional contributions to the covariance matrix (“process noise” in the terminology of dynamic systems).

In order to speed up the reconstruction, the material of the tracker is described in a simplified way:

- all material is assumed to be concentrated on thin surfaces. In the current version these surfaces coincide with the active elements;
- the material properties of each detector layer are described by two numbers: the thickness in units of radiation length, and the thickness multiplied by the mean ratio of atomic number to atomic mass.

Two kinds of effects are taken into account:

- energy loss (for electrons due to Bremsstrahlung, for all other particles due to ionization), and
- multiple scattering.

Ionization energy loss is described according to the Bethe-Bloch formula without density correction [14-31]. The mean excitation energy used is that of silicon. The average and variance of the fractional energy loss of electrons due to radiation are computed based on the p.d.f. given by Bethe and Heitler [14-32].

The effect of multiple scattering is calculated using the approximation for the Gaussian core given by Highland [14-31]. As explained above, the effects due to the presence of material only enter the propagation stage of the Kalman filter. The track propagation is thus divided into two parts: a purely geometrical extrapolation and an update of track parameters and their covariance matrix at the destination surface. The mean energy loss <dE> results in a change of the predicted momentum, while the spread of the energy loss and the deflection due to multiple scattering modify the covariance matrix. Corrections to the path length due to the angle of incidence are also taken into account and applied both in the forward and, reverting the sign of <dE>, in the backward filter.

The simplified treatment of the material is valid only inside the tracker volume. Propagation to the Muon system requires detailed knowledge of the passive material. Currently the propagations outside of the tracker volume are performed with the GEANE package [14-33], which uses the full simulation geometry. For convenience and conformance the GEANE routines are also encapsulated in a Propagator.

14.4.1.2 Track Reconstruction Phases

Track reconstruction is decomposed in four phases:

- Trajectory seed generation;
- Trajectory building - seeded pattern recognition;
- Trajectory cleaning - resolution of ambiguities;
- Trajectory smoothing - final fit.

These components are described in detail in the following sections.

14.4.1.2.1 Seed Generator

Seed generation, which provides initial trajectory candidates can be internal to the tracking detector (inner tracker or muon system), or external, using input from other detectors (calorimeters). The seed must provide initial trajectory parameters and errors. To constrain all five parameters of the trajectory one needs at least two two-dimensional position measurements and a beam spot constraint, or a calorimetric measurement of position and energy and a beam spot constraint.

Consequently, internal seeds consist of pairs of RecHits and use a beam spot constraint and a minimal transverse momentum constraint to provide initial trajectory parameters. The beam spot constraint may introduce a bias at this stage, which is removed during the final fit.

The “beam spot”, or track origin region, used in the seed generator, is chosen according to context. For example, to reconstruct primary muons when the event vertex is not known the origin can be a cylinder with radius of 50 μm and length of 30 cm. If the primary vertex is known, e.g. from the already reconstructed muon, and the goal is b-jet tagging, the origin could be a cylinder 1 mm in radius and 2 mm long. The choice of the origin size can have dramatic effect on the reconstruction efficiency, fake rate, and CPU time. If the origin is too small the track reconstruction efficiency will suffer; if the origin size is bigger than the physics requirements, additional CPU time will be wasted to reconstruct ghost tracks that will have to be removed with additional track selection criteria.

In a similar way, the choice of the minimal transverse momentum has a strong influence on the number of seeds, and even more on the overall CPU time, since it takes longer to reconstruct a soft track than a hard track. The minimal transverse momentum is defined by physics requirements. It is typically 0.9 GeV for b-tagging, and can be as high as 20 GeV for trigger muon reconstruction.

The two hits that constitute an internal tracker seed are chosen to come from two different layers, the so called seeding layers. The best reconstruction performance, both in efficiency, accuracy and CPU time, is achieved when the pixel layers are used for seeding. There are several reasons for this:

- the channel occupancy and the noise of the pixel detectors is much lower than for the strip detectors;
- the quality of the 2D measurement is much better for the pixel detectors: there are no combinatorial ambiguities and both coordinates are measured with ~20 μm accuracy;

- the interaction probability for π in the tracker is so high that more than 20% of the 1 GeV pions do not reach the outermost layers. The reconstruction efficiency is therefore lower if the outer seeding layers are used, even if one allows many combinations;
- the number of seeds is smaller for pixel layers (for a given origin size and minimal transverse momentum);
- for some applications, like b-tagging or secondary vertex reconstruction, the tracks which do not have two pixel hits are not very useful and would not be considered anyway.

The pixel layers are therefore essential for fast and accurate track reconstruction. In case of staging of some of them, the innermost double-sided silicon strip layers are used.

For the baseline pixel detector (three barrel and two end-cap layers on each side), all pairs of layers are used to provide a high degree of redundancy and efficiency.

14.4.1.2.2 Trajectory Builder

In the trajectory builder the seed is transformed into a (possibly empty) set of trajectory candidates. The trajectory builder acts on only one seed at a time. To reduce the complexity of pattern recognition in a volume filled with 20,000 sensors, the trajectory building is done in terms of detector layers. There are only 41 layers in the inner tracker (13 barrel and 14 end-cap on each side), and the sequence of layers that a track can follow is simple (compared to the sequence of individual sensors).

The trajectory builder is based on the Kalman filter formalism, and is decomposed in the following way:

- layer navigation provides a list of reachable layers from the current layer in a given direction. Only the immediately reachable layers are provided (i.e. those that can be reached without crossing any other layers). The list is empty when the outer edge of the tracker is reached. The navigation can be internal to one detector (inner tracker or muon system), or can be combined between the inner tracker and the muon system;
- each reachable DetLayer provides measurements (reconstructed hits) compatible with a trajectory candidate. Since the compatibility test requires knowledge of the predicted trajectory parameters and errors at the measurement surface of the RecHit, these are also computed at this stage (using a Propagator);
- each compatible measurement is combined with the corresponding predicted trajectory state (using an Updator). The updated state is then used as the starting state for continuing to the following layers. At this stage the number of already existing trajectory candidates is multiplied by the number of compatible measurements, which can lead to “combinatorial explosion”. This is avoided by limiting the total number of trajectory candidates allowed to proceed to the next layer: only the best N candidates (in terms of χ^2) are retained.

The Trajectory Builder uses these components sequentially to follow all possible continuations of the seed, with some logic to limit the number of candidates. The logic is highly configurable, allowing a smooth transition from very fast to very efficient pattern recognition.

14.4.1.2.3 Trajectory Cleaner

The trajectory candidates resulting from a single seed are usually (but not always) mutually exclusive, in the sense that they have most of their hits in common. The trajectory cleaner resolves the ambiguities in a

very simple way, by finding subsets of candidates that share more than a given percentage of hits (typically 50%), and retaining only the best (in terms of χ^2 and number of hits) candidate for each set.

Trajectory cleaning is also invoked on all surviving candidates from all seeds, since trajectories from two different seeds can also be mutually exclusive.

14.4.1.2.4 Trajectory Smoother

After trajectory building and cleaning, the parameters of each surviving trajectory are fully known at the last measurement (typically at the outer layers of the tracker), but are very poorly determined at the origin. A “backward fit” of the same hits as in the forward fit, but starting from outside, gives optimal knowledge of the parameters at origin, but loses the knowledge at the outer layers. In the Kalman formalism, the procedure of combining the forward and backward fits in a statistically correct way (without double counting) is called “smoothing”. After smoothing the track parameters at every measurement surface along the track fully include all other measurements, and the χ^2 and probability of the track are more reliable than after a forward or a backward fit.

At this stage any possible bias introduced by the beam spot constraint of the seed can be removed by repeating the forward fit starting from the first measurement.

The track parameters at the interaction point, at the calorimeter surface, or at any other surface are computed by extrapolation (propagation) from the nearest measurement on the track.

14.4.1.3 Track Reconstruction Performance

The track reconstruction efficiency when using either the Combinatorial Trajectory Builder or the Forward Kalman Filter has been estimated for samples of single muons with P_T = 1, 10 and 100 GeV/c (see Figure 14-32). A track is deemed to be successfully reconstructed if it shares more than 50% of the hits of a simulated track. Reconstructed tracks are required to have at least 8 hits. Two efficiencies are defined.

The “algorithmic efficiency” (Figure 14-32, left plot) is the efficiency of reconstructing correctly tracks which have at least eight hits in the tracker, of which at least two are in the pixel detector. This measures directly the performance of the track reconstruction algorithm and is essentially the efficiency of the trajectory builder, since the seed generator is fully efficient. The “global efficiency” (Figure 14-32, right plot) is the reconstruction efficiency for all tracks. In addition to the efficiency of the algorithm, it includes the acceptance, hit efficiency and any other factor influencing reconstruction. It will mainly differ from the algorithmic efficiency in the forward region, with the loss of coverage of the disks, especially in the pixel system.

As the algorithmic efficiency shows, the trajectory builder is fully efficient in the full pseudo-rapidity range, with a drop beyond $|\eta|$=2.4 due to the lack of coverage of the end-cap disks. For the global efficiency, the drop of efficiency in the region at η~0 is due to the gaps between the two half barrels of the pixel at z=0. As the gaps for the three layers are aligned, this will cause some tracks not to have the two required pixel hits. At high pseudorapidity, the drop of efficiency is mainly due to the lack of coverage of the two pairs of forward/backward pixel disks.

Five parameters are chosen to describe a track: d_0, z_0, ϕ, cot θ and the transverse momentum P_T. The track parameters are defined at the point of closest approach of the track to the beam axis (this point is called the impact point); d_0 and z_0 hence measure the coordinate of the impact point in the transverse and longitudinal plane ($d_0 = y_0 * \cos \phi - x_0 * \sin \phi$, where x_0 and y_0 are the transverse coordinates of the impact point). ϕ is the azimuthal angle of the momentum vector of the track at the impact point and θ the polar

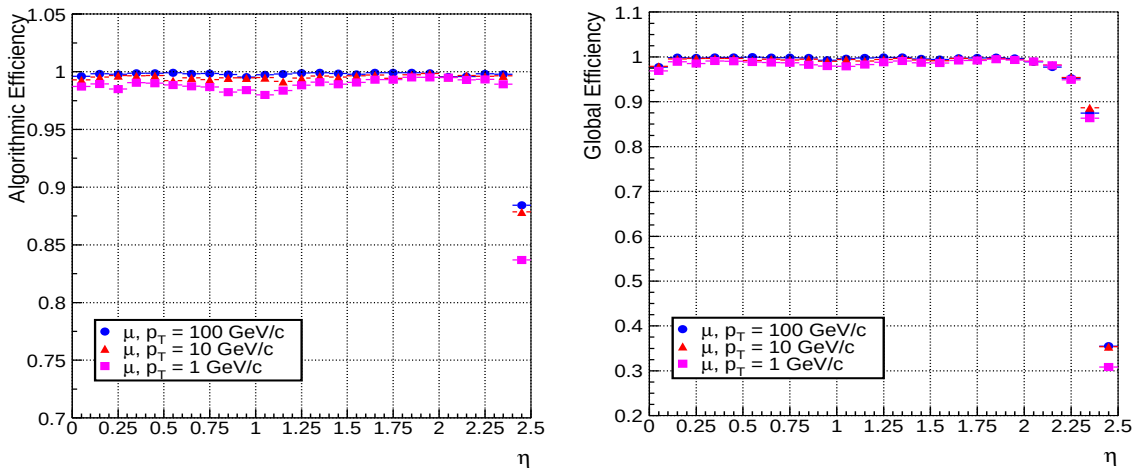


Figure 14-32 Algorithmic (left) and global (right) track reconstruction efficiency for single muons.

angle. Figure 14-33 show the resolution of the five track parameters for samples of single muons with P_T of 1, 10 and 100 GeV/c using the Combinatorial Trajectory Builder and the Forward Kalman Filter. The resolution of the transverse momentum is shown in Figure 14-33 a). At high momentum (100 GeV/c), the resolution is around 1-2% up to a pseudo-rapidity of $|\eta|=1.75$, for higher values of $|\eta|$ the lever arm of the measurement is reduced. The degradation around $|\eta|=1.0$ is due to the gap between the barrel and the end-cap disks and the degradation beyond $|\eta|=1.1$ is due to the lower hit resolution of the last hits of the track measured in the TEC ring 7 with respect to the hit resolution in the TOB layers 5 and 6. At a transverse momentum of 100 GeV/c, the material in the tracker accounts for between 20 and 30% of the transverse momentum resolution; at lower momenta, the resolution is dominated by multiple scattering and its distribution reflects the amount of material traversed by the track.

The resolutions on the transverse and longitudinal impact parameters d_0 and z_0 are shown in Figure 14-33 d) and e). At high momentum, the d_0 resolution is fairly constant and is dominated by the hit resolution of the first hit in the pixel detector. At lower momenta, the d_0 resolution is progressively degraded by multiple scattering, until the latter becomes dominant. The z_0 resolution of high momentum tracks is also dominated by the hit resolution of the first pixel hit, with multiple scattering dominating at low momentum. The improvement of the z_0 resolution up to a pseudo-rapidity of $|\eta|=0.5$ can be attributed to the fact that in the barrel, as the angle with which the tracks cross the pixel layers increases the clusters become wider, improving the pixel-hit resolution.

14.4.2 Vertex Reconstruction

For the High-Level Trigger, three uses of vertex reconstruction have been identified.

- Reconstruction of the primary vertex of the signal event. Knowing the primary vertex, initial track segments built with the hits from the pixel detector can be filtered, rejecting those not pointing to the primary vertex. The track seeds pointing to the primary vertex can be used, for τ or electron identification and tests of particle isolation, with track reconstruction pursued from these seeds only, saving computation time;
- Detection of a secondary vertex inside a τ- or b-jet candidate;

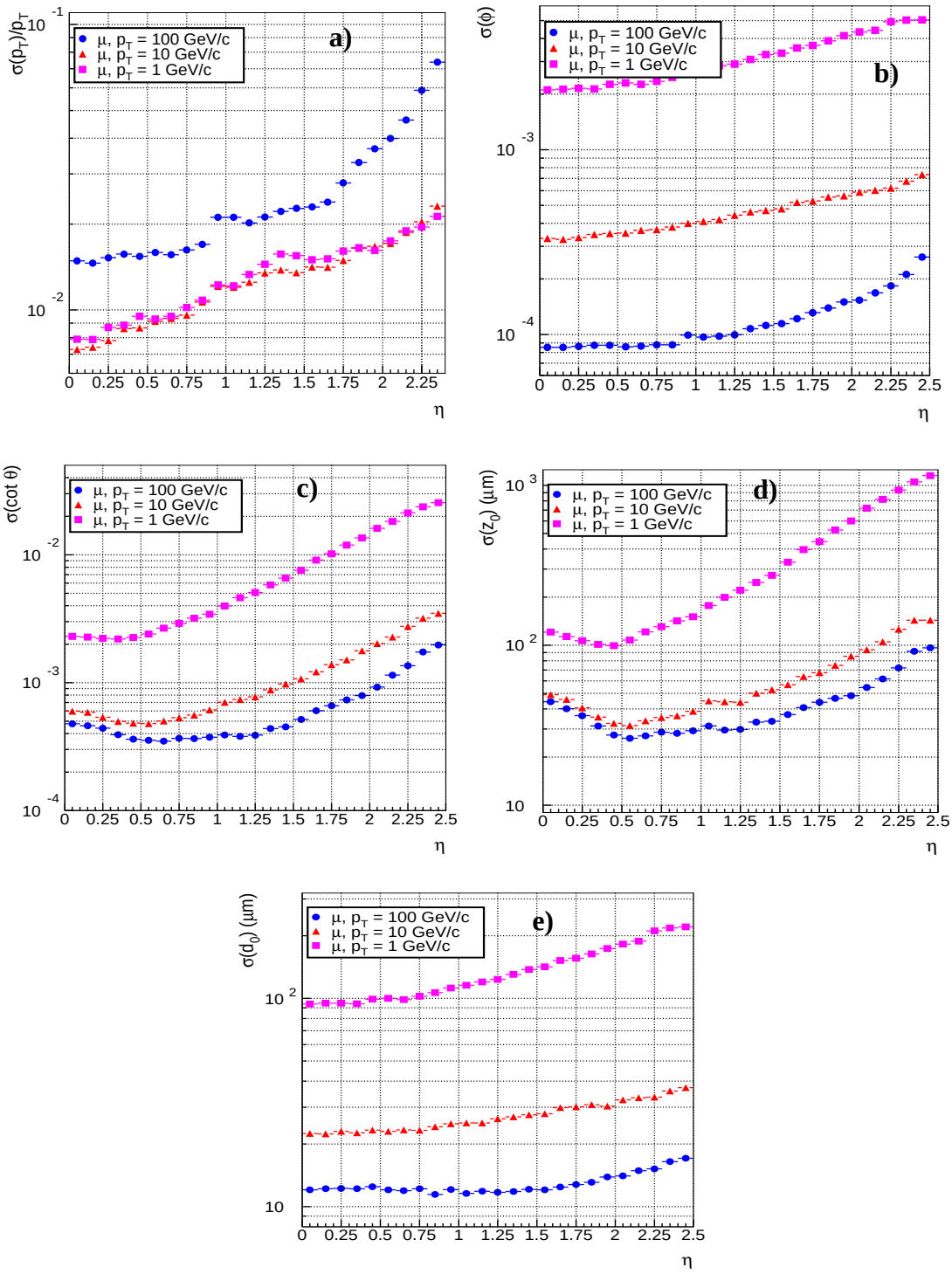


Figure 14-33 Resolution of the five track parameters for single muons with transverse momentum of 1, 10 and 100 GeV/c using the Combinatorial Trajectory Builder and the Forward Kalman Filter.

- Study of the event kinematics at the vertex. In B-physics, selection of an exclusive decay channel requires computation of invariant masses, etc. This implies knowledge of the B decay vertex position and of the track parameters at the vertex.

These require fast vertex finding and fitting algorithms. The ORCA vertex reconstruction package provides such algorithms for both HLT and offline reconstruction. It also provides a software framework for algorithm development and testing. This section summarizes the algorithms, stressing the aspects relevant to HLT, and their use in HLT benchmark studies.

14.4.2.1 Vertex Fitting

The vertex fitting algorithm used for HLT analyses is described in [14-34]. It is based on a straight line approximation of the tracks in the vicinity of the vertex, each track being replaced by the straight line tangential to it at a certain linearization point. For tracks with $p_T > 0.5$ GeV in the 4 T magnetic field, the linearization error amounts to 1-2 μm, providing that the linearization point is less than 2 mm from the true vertex position. In this approximation the 3D vertex position which minimizes the fit χ^2 has an explicit expression involving only 3x3 matrix algebra. The computation is thus very fast. To guarantee precision and robustness at low p_T, the vertex fit is repeated as long as the distance between the fitted vertex position and the linearization point is larger than 2 mm. The linearization point used at iteration j is the vertex position fitted at iteration (j-1).

Figure 14-34 shows the distribution of the χ^2 probability $P(\chi^2(n) > \chi^2_{fit})$ of the vertex fit for a) tracks with

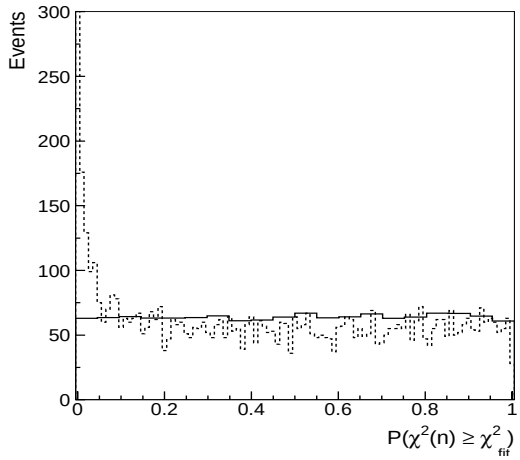


Figure 14-34 χ^2 probability $P(\chi^2(n) > \chi^2_{fit})$ of the vertex fit for tracks with perfectly Gaussian track parameters (full line) and tracks with parameters affected by 3% non-Gaussian tails (dashed line).

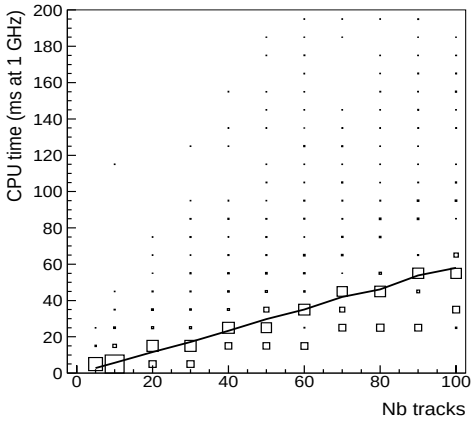


Figure 14-35 CPU Time distribution as a function of the number of tracks. A 1 Ghz Pentium-III CPU is used.

perfectly Gaussian track parameters (full line), and b) tracks affected by 3% non-Gaussian tails reconstructed with ORCA (dashed line). Here n denotes the number of degrees of freedom of the vertex fit. The flatness of the distribution in the first case shows the statistical consistency of the algorithm.

Figure 14-35 shows the distribution of the computation time on a 1 GHz CPU as a function of the number of tracks entering the vertex fit. The full line shows the average time <t>. The increase is linear as expected. The average time is very small even at the largest multiplicities expected (55 ms for 100 tracks). The fraction of fits requiring a computation time larger than 2*<t> is below 1%.

The resolution on the vertex position is dominated by the precision of the track impact parameter measurement, and the straight line approximation gives negligible contribution. The vertex fitting algorithm

provides an accurate estimation of the vertex position uncertainty. This is shown by the distributions of the pulls of the vertex coordinates in Figure 14-36, for tracks with perfectly Gaussian parameters (full line) and for ORCA tracks in 100 GeV u-jets (dashed line).

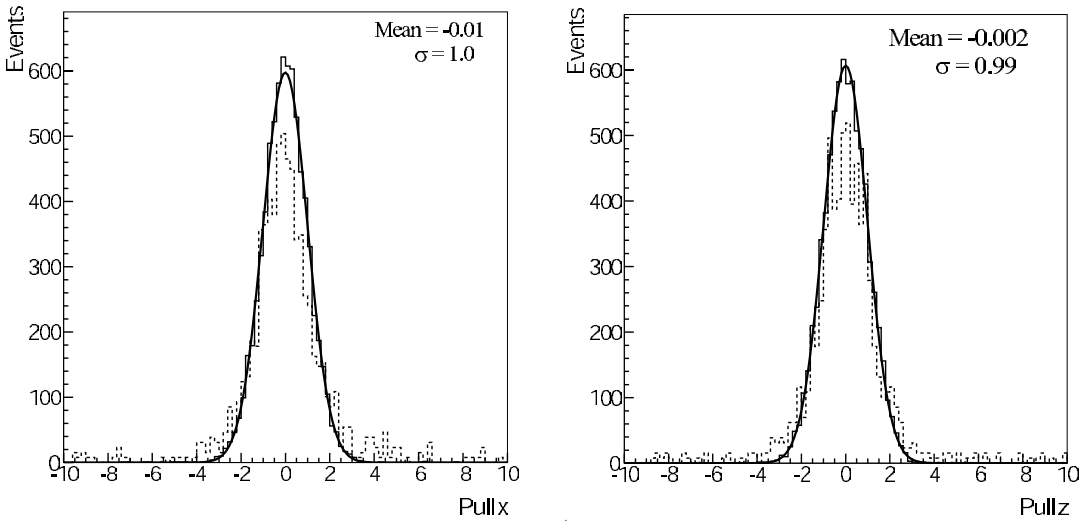


Figure 14-36 Pulls of the vertex coordinates for tracks with perfectly Gaussian parameters (full line) and for ORCA tracks in 100 GeV u-jets (dashed line).

The flight path value and significance is used in numerous algorithms, for example in τ-tagging, using the linearized vertex fitter to fit the three-prong tau decay vertex.

14.4.2.2 Vertex Finding

14.4.2.2.1 Definition of Vertex Finding Efficiency and Fake Rate

The efficiency value is very sensitive to the definition of a “reconstructible” vertex, and to the requirements on the reconstructed vertex quality. In what follows a simulated vertex is considered to be reconstructible if at least 2 of its tracks have been reconstructed. A reconstructible vertex is defined as found if it is associated to a reconstructed vertex made out of tracks, the majority (>55%) of which originate from the correct simulated vertex. The fraction of tracks which originate from the correct simulated vertex is called the purity of the reconstructed vertex. This definition of efficiency is adequate for comparing vertex finding algorithms, since it is sensitive to small differences in performance. It leads to low efficiency in b-jets since tracks from the decay chain are mixed. Such merged vertices are still useful for b-tagging, as shown later in the text. The fake rate is defined as the number of reconstructed vertices per event, which have no associated simulated vertex.

14.4.2.2.2 Pixel Primary Vertex Finding

With 3 pixel hits per track one can approximately reconstruct tracks and find primary vertices (PVs) using the pixel detector only. At low luminosity, when only two pixel barrel layers are available, the transverse beam constraint can be used. A summary of the algorithms follows.

Pixel hits are matched in r-ϕ and r-z to establish track candidates. Using these tracks, a list of primary vertices is formed with z values where at least three tracks cross the z axis. Tracks which do not point to any

vertex candidate are erased. The algorithm results in a list of primary vertices and lists of tracks attached to each vertex. The hit matching cuts have been optimized to reconstruct tracks above 1 GeV/c in P_T. For all event types which have been simulated the track finding efficiency is above 90% with the number of ghosts being between 1% and 10%.

The PV candidates found during the track finding stage are re-analyzed. Only PVs with at least 3 valid tracks are kept and the position of each vertex is estimated as the mean value of the z impact parameters of all tracks assigned to it. In addition to the main ‘‘signal’’ PV, on the average 6 to 8 more PVs per event are found at high luminosity. The signal PV is usually found with a high accuracy of better than 50 μm. This is shown in Figure 14-37 where the difference in the z position of the reconstructed vertex and the Monte Carlo vertex is plotted. The efficiency of this algorithm is high; in the HLT event samples the signal PV is always found with an efficiency of better than 95%.

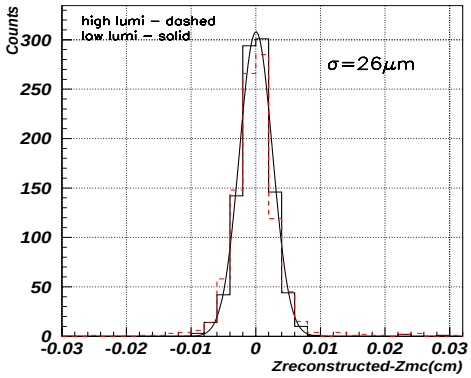


Figure 14-37 The difference (in cm) between the z position of the Monte Carlo primary vertex and the reconstructed primary vertex. Only pixel hits are used in the vertex reconstruction.

14.4.2.3 Secondary Vertex Finding: Principal Vertex Finder

The recursive secondary vertex finding algorithm finds the principal vertex among a set of tracks: it fits all tracks to a common vertex, separates the incompatible tracks, stores the cleaned-up vertex if its χ^2 is small enough, and searches for additional vertices in the set of tracks discarded in the previous iteration. It only has two parameters to tune: the cut on the probability of compatibility of a track to a vertex, and the cut on the vertex fit χ^2. It is thus simple to optimize, and even more so since meaningful cut values are guided by statistics (the default cuts at 5% in probability are optimal in all physics cases tested so far). However since the algorithm has to test the compatibility of each track to the fitted vertex at each iteration, the CPU time is expected to rise as N^2, where N is the initial number of tracks.

Figure 14-38 shows the efficiency to find a reconstructible secondary vertex inside a b-jet, as a function of the jet E_T and η. The average efficiency is ~30%. The inefficiency is due to the following reasons:

- often there are less than two tracks which are significantly displaced from the primary vertex and which can make a secondary vertex. This is visible in Figure 14-39 showing the efficiency versus the impact parameter significance of the track with the 2^{nd} largest impact parameter. This effect accounts for 20% of the inefficiency;

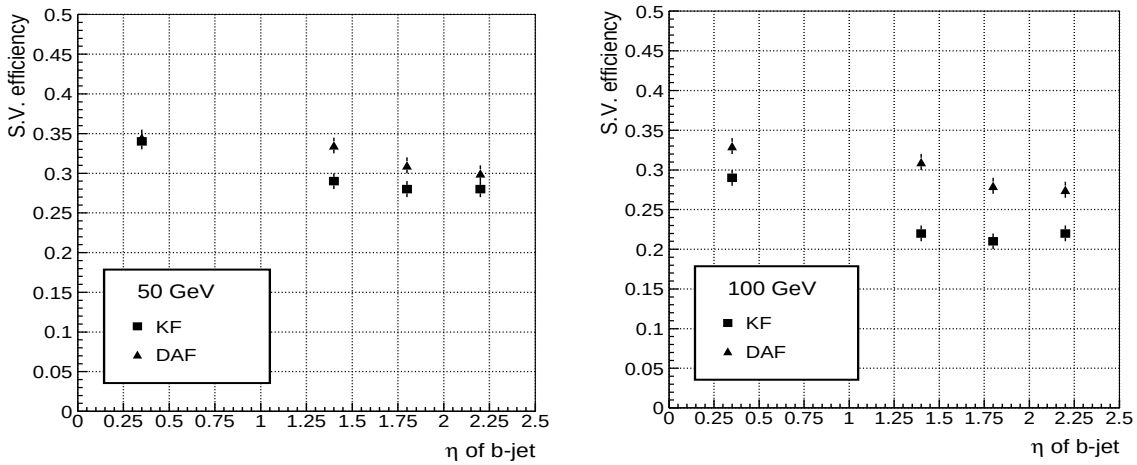


Figure 14-38 Efficiency to find a reconstructible secondary vertex inside a *b*-jet, as a function of the jet E_T and η, with the Kalman Filter track reconstruction (KF) and with an algorithm providing a better estimation of the track parameter errors (Deterministic Annealing Filter, DAF).

- vertices from the b-decay chain get mixed and lead to reconstructed vertices of low purity. The purity cut accounts for another 20% of inefficiency, as shown in Figure 14-39 when relaxing the purity cut down to 39%.

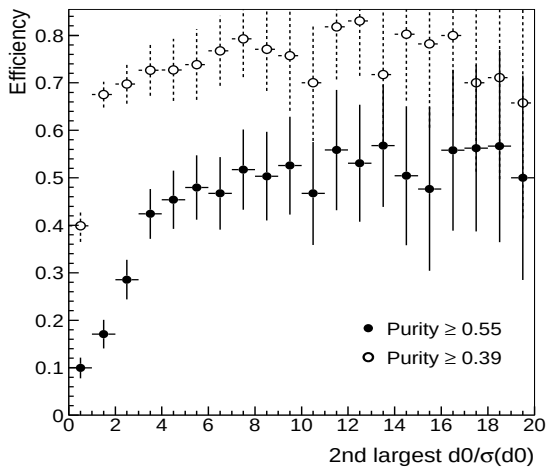


Figure 14-39 Secondary vertex efficiency versus impact parameter significance of the track with the 2nd largest impact parameter.

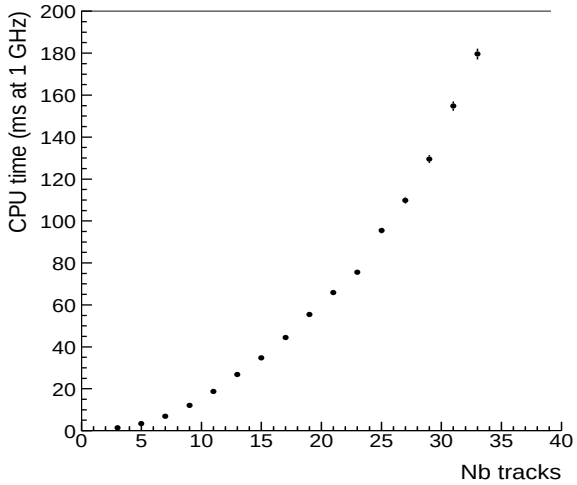


Figure 14-40 CPU time for vertex reconstruction using the principal vertex finder, as a function of the number of tracks on input. A 1 GHz Intel Pentium-III CPU is used.

The fake rate of secondary vertices of this algorithm is very low. In $u\bar{u}$ events with 100 GeV u-jets, it amounts to five fake vertices per 1000 events.

Figure 14-40 shows the CPU time spent using the principal vertex finder, normalized to a 1 GHz CPU, as a function of the number of tracks on input. The computation time increases as N^2 as expected. The CPU time remains below 20 ms for multiplicities below 10. This multiplicity is characteristic of 100 GeV *b*-jets, inside a cone of $\Delta R = 0.4$ around the jet axis, containing 95% of the tracks from the *b*-decay chain.

The CPU performance means that even inclusive secondary vertex finding should be affordable at HLT, e.g. for b-tagging purposes.

14.4.3 High-Level Trigger (HLT) Tracking

Track reconstruction for the HLT is special in two respects:

- it is typically guided by external information from a Level-1 Trigger or a High-Level Trigger candidate. This external input defines one or more regions for track reconstruction;
- it has to address specific questions, such as verifying that the transverse momentum of a Level-2 muon candidate is above a threshold, whether a jet is a *b*-jet or whether an electron is isolated. In most cases the answer can be given without fully reconstructing all tracks in the region in question.

These two aspects lead naturally to the concepts of regional, partial and conditional track reconstruction. Most importantly, the CPU time available for a trigger decision must be kept to a minimum. Therefore HLT track reconstruction concentrates on reducing the number of seeds to a low level, and in minimising the number of operations per seed. The first goal is achieved with a regional seed generator whereas the second goal is achieved with partial, or conditional, trajectory building.

14.4.3.1 Tracking Region and Regional Seeding

A tracking region is 5-dimensional since a helix is 5-dimensional. These dimensions, and therefore the corresponding constraints which can be put on the region, are conveniently represented as:

- 2 positions: the transverse and longitudinal impact parameter with respect to the interaction region;
- 2 directions: the pseudorapidity and the azimuthal angle;
- the transverse momentum, P_T.

All five constraints can be used to limit the number of seeds. In fact, the standard seed generator already uses the constraints on the size of the origin and on the minimal transverse momentum (Section 14.4.1.2.1). In addition, the regional seed generator uses the angular constraints to restrict the hits on the seeding layers. The resulting seeds are compatible with the tracking region.

For each seeding layer a measurement estimator matching the size of the tracking region is constructed, and the hits from the layer are requested using this estimator. The DetLayer uses the estimator first to find the compatible detectors, and then to filter the hits from the compatible detectors. The detectors which are not compatible with the tracking region are never accessed for hits, and their digis (raw data) are never accessed. Since the loading of the detector raw data happens on demand, this automatically ensures that only the minimal necessary amount of raw data will be accessed and clustered.

The reduction factor in number of seeds depends on the size of the region, which in turn depends on the physics. In some cases, like trigger muon reconstruction, the constraints on transverse impact parameter, momentum, and angles are sufficiently strong to reduce the number of seeds below 100, even without any knowledge of the z position of the primary vertex of the signal event. In other cases, like b-tagging of a jet, the number of seeds in the jet cone is too big for direct reconstruction (of the order of 2000 at high luminosity for a cone size of interest). In such cases knowledge of the z position of the signal primary vertex is required to reduce further the number of seeds.

The pixel vertex reconstruction, described in Section 14.4.2.2.2, provides primary vertex candidates without prior track reconstruction, and within the CPU time available at HLT, up to the full Level-1 rate. The

elimination of all seeds not compatible with primary vertex candidates results in an acceptable number of seeds within an acceptable CPU time.

14.4.3.2 Partial (Conditional) Track Reconstruction

The trajectory-building stage is affected significantly by the momentum constraint which provides a useful speed-up since trajectory candidates with P_T below the cut are abandoned immediately. The other constraints are not very helpful: if the seed is within the tracking region, the resulting track is very likely to be inside thne regionas well. The speed of trajectory building can be increased, at the cost of reconstruction efficiency, by tuning the various parameters that control the number of combinations and the acceptable number of inefficient layers. However, for algorithmic efficiencies above 97% the savings in CPU are not substantial.

The idea of partial reconstruction goes beyond the tuning of algorithmic parameters. Starting from a HLT object specification it addresses the minimum reconstruction needed to provide the information needed by the requirement. A simple example helps illustrate the process: if an electron or a muon is already reconstructed and passes the trigger criteria, and the next step is to check if it is isolated, the following steps must be performed:

- the seeds in the isolation cone must be constructed. The z impact parameter of the trigger lepton can be used to define the signal vertex and thus reduce the number of seeds;
- trajectory-building must proceed for all seeds until enough tracks are reconstructed to fail the isolation criteria. Typically, as soon as one or two tracks are found, the lepton can be classified as non-isolated, rendering the reconstruction of any remaining seeds unnecessary.

If the rejection factor of this trigger is high, i.e. most leptons are not isolated, and if the average multiplicity in the isolation cone is high, the gain in CPU time can be substantial.

Conditional reconstruction works particularly well in restricting the trajectory-building process. Reconstructed tracks are used only if they satisfy some selection criteria. If it is possible to recognise at an early stage of trajectory building, that a track candidate fails the selection, then the reconstruction of this candidate can be stopped, and the CPU time saved. Again, an example can illustrate this.

Almost every set of physics-object selection criteria has a requirement on the minimum P_T of the object. The P_T is in turn known after each update step in the Kalman filter. For a real track, adding subsequent hits cannot modify the momentum significantly at any point along the track. Using a Guassian approximation, the change in track parameters should be smaller than a number of standard deviations on the momentum measurement. It is therefore safe to stop the reconstruction of any trajectory candidate if its current P_T drops e.g. 5 σ below the cut value, even if there are only three hits on the track. This is called a stopping condition for trajectory building.

As for the seed generator, the minimal P_T condition is always present in the trajectory builder since there is always a value below which reconstruction is not sensible. This reduces significantly the CPU time (this is the only use of the Tracking Region in the trajectory builder).

This is the essence of conditional trajectory building: tracks are reconstructed until some condition is met. A generic interface allows the specification of an unlimited number of stopping conditions. Some conditions are expensive to verify numerically. For example, for *b*-tagging it is important to reconstruct a track until the accuracy of the impact parameter is sufficient, or the impact parameter is significant. However, in an “inside-out” Kalman filter (as is the case with pixel seeds) the track parameters are not known at the interaction point until the smoothing stage. It would be too expensive computationally to smooth each

candidate after each update with a new hit, only to check if the precision or significance on the impact parameter is reached.

A study of the accuracy on track parameters as a function of the number of hits in the track, shown in Figure 14-41, reveals that asymptotic accuracy on the impact parameter is reached after only 5 or 6 hits. This is true for a large range in momentum and in pseudorapidity, covering the physically interesting region. Therefore, the most commonly used stopping condition for HLT tracking is to limit the number of hits in a track to 5 or 6, provided the ghost track rate is acceptable. Figure 14-42 also shows the error on the track 2D impact parameter as a function of the number of hits along the track. Using tracks with 5 hits the precision is comparable to the full reconstruction. A reduced number of hits along the track, however, increases the fraction of ghost tracks.

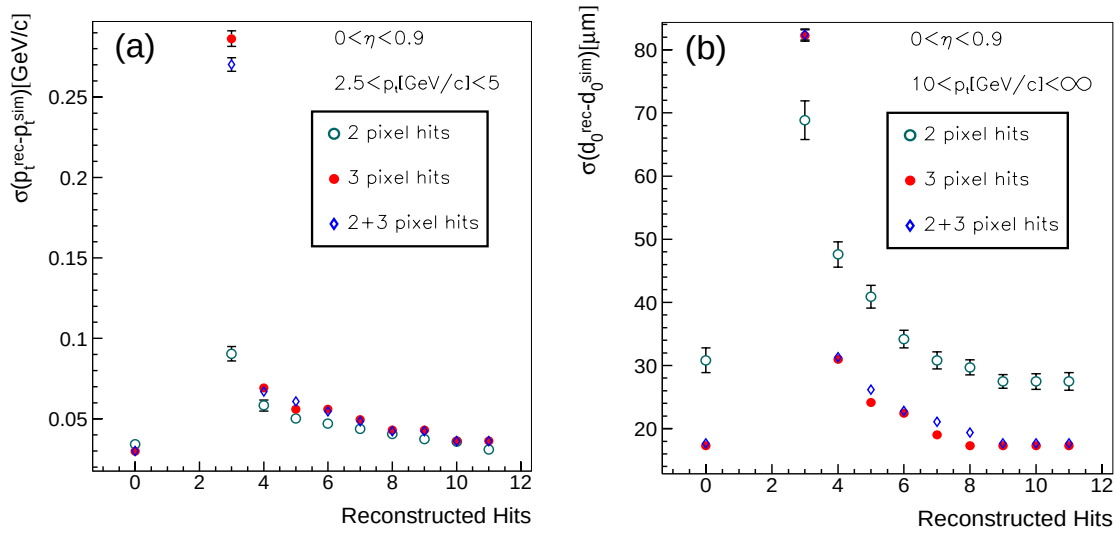


Figure 14-41 The resolution on a) P_T and b) impact parameter for partial track reconstruction, compared with full track reconstruction, as a function of the number of smoothing steps in different P_T and for the barrel region.The leftmost point at "0" reconstructed hits shows the full tracker performance.

Figure 14-42 compares the efficiency for track reconstruction with the fraction of ghost tracks as a function of the number of hits along the track, for different η ranges. The efficiency for track reconstruction decreases slightly with increasing the number of hits, mainly due to nuclear interactions in the detector. The fake rate decreases to below the 1% for tracks with at least six hits. In the case of staged pixel detector the fake rate increases considerably for the same number of hits.

14.5 References

14-1 V. Innocente, L. Silvestris and D. Stickland, on behalf of the CMS collaboration, "CMS Software Architecture. Software framework, services and persistency in High-Level trigger, reconstruction and analysis", Comput. Phys. Commun., 140 (2001) 31; Coherent Object Base for Reconstruction, Analysis and simulation (COBRA), see http://cobra.web.cern.ch/cobra/.

14-2 T. Sjostrand et al., "High Energy Physics Event Generation with PYTHIA 6.1", hep-ph/0010017, LU TP 00-30; also S. Mrenna, "SPYTHIA, A Supersymmetric Extension of PYTHIA 5.7", Comput. Phys. Commun. 101 (1997) 232.

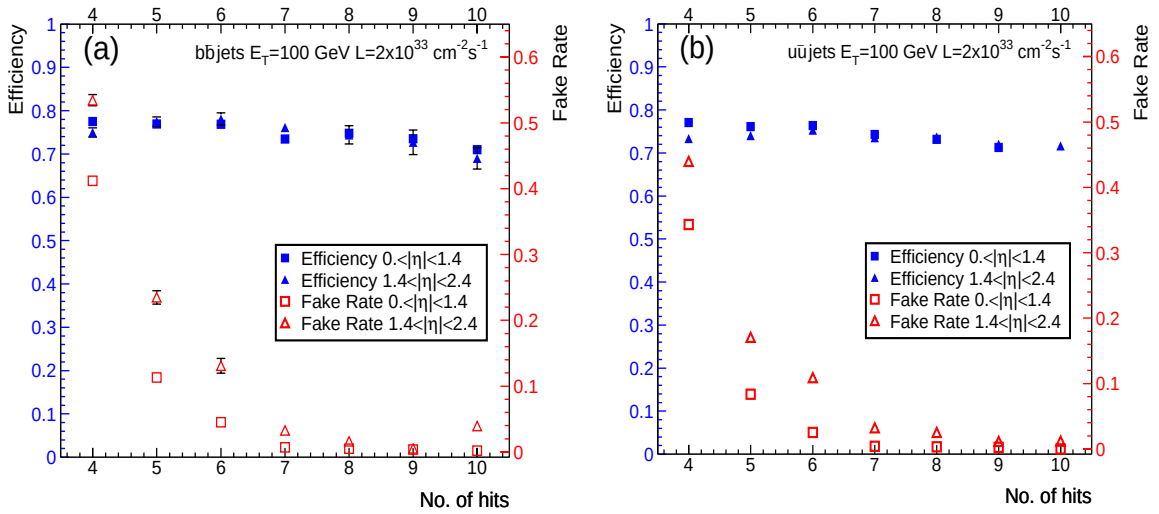


Figure 14-42 Efficiency (left axis) and ghost rate (right axis) for track reconstruction as a function of the number of hits along the track for 100 GeV jets. The left plot is for *b* tracks and the right plot for *u*-jet tracks.

14-3 F. Paige and S. Protopopescu, in Supercollider Physics, p.41, ed. D. Soper (World Scientific, 1986); also H. Baer et al., in Proceedings of the Workshop on Physics at Current Accelerators and Supercolliders, ed. J. Hewett, A. White and D. Zeppenfeld, (Argonne National Laboratory, 1993).

14-4 N. Amapane et al., “Monte Carlo Simulation of Inclusive Single- and Di-Muon Samples”, CMS Note 2002/041.

14-5 H. Sakulin, “Methodology for the Simulation of Single-Muon and Di-Muon Trigger Rates at the CMS Experiment in the Presence of Pile–Up”, CMS Note 2002/042.

14-6 D. Acosta, M. Stoutimore, and S.M. Wang, “Simulated Performance of the CSC Track-Finder”, CMS Note 2001/033.

14-7 M. Huhtinen, “Optimization of the CMS Forward Shielding”, CMS Note 2000/068.

14-8 G. Bruno and M. Konecki, “Simulation of the Baseline RPC Trigger System for CMS: Efficiency and Output Rates in Single Muon Topology “, CMS Note 2001/012.

14-9 CMS Simulation Package CMSIM — Users' Guide and Reference Manual, http://cmsdoc.cern.ch/cmsim/cmsim.html

14-10 GEANT3, version 3.21/13 (release 15111999), Detector Description and Simulation Tool, CERN program library long writeup W5013.

14-11 CMS Coll.,“The Electromagnetic Calorimeter Project, Technical Design Report”, CERN/LHCC 97-33, CMS TDR 4, 15 December 1997.

14-12 1999 Status Report on the RD-12 Project, CERN/LHCC 2000-002, 3 January 2000.

14-13 V. V. Abramov et al., "Studies of the response of the prototype CMS hadron calorimeter, including magnetic field effects, to pion, electron, and muon beams", Nucl. Instr. and Meth. A 457 (2001) 75.

14-14 D. Litvintsev, “Detailed Monte Carlo program for Quartz Very Forward Calorimeter for CMS Detector”, CMS Technical Note 1995/098.

14-15 GLANDZ, see http://wwwinfo.cern.ch/asdoc/geantold/GEANTMAIN.html

14-16 RAL Micro-electronics Design Group, “APV25 User Guide”, http://www.te.rl.ac.uk/med/projects/High_Energy_Physics/CMS/APV25-S1/pdf/User_Guide_2.0.pdf

14-17 M. Raymond et al., “ The CMS Tracker APV25 0.25um CMOS Readout Chip”, in Proceedings of the 6th Workshop on Electronics for LHC Experiments, Cracow, Poland, 11-15 September 2000, pp. 130-134.

14-18 R.Veenhof, CERN program library long writeup W5050, GARFIELD.

14-19 A. Gresele and T.Rovelli, “Parametrization of B Field Effects in DTBx”, CMS Note 1999/064.

14-20 CMS Coll.,“The Muon Project, Technical Design Report”, CERN/LHCC 97-32, CMS TDR 3, 15 December 1997.

14-21 M.Aguilar-Benitez at al., “Construction and Test of the final CMS Barrel Drift Tube Muon Chamber Prototype”, Nucl. Instr. and Meth. A480 (2002) 658;
M. Cerrada et al., “Results from the Analysis of the Test Beam Data taken with the Barrel Muon DT Prototype Q4”, CMS Note 2001/041.

14-22 R. Wilkinson and P.T. Cox, “Simulating the Endcap Muon CSC System in ORCA”, CMS Note 2001/013.

14-23 E. Gatti et al., “Analysis of the position resolution in centroid measurements in MWPC”, Nucl. Instr. and Meth. 188 (1981) 327.

14-24 K.W. Bell et al., “User Requirements Document for the Final FED of the CMS Silicon Strip Tracker”, CMS Note 2001/043.

14-25 O. Kodolova, I. Tomalin and P. Yepes, “Expected Data Rates from the Silicon Strip Tracker”, CMS Note-2002/047.

14-26 B. Baur et al. ,”Heavy Ion Physics Programme in CMS”, CMS Note 2000/060.

14-27 .CMS Coll.,“The Magnet Project, Technical Design Report”, CERN/LHCC 97-10, CMS TDR 1, 2 May 1997.

14-28 S. Cucciarelli, D. Kotlinski and T. Todorov, “Position Determination of Pixel Hits”, CMS Note 2002/049.

14-29 M. Winkler, PhD Thesis, 2002/015.

14-30 R. Fruhwirth, Nucl. Instr. and Meth. A262 (1987) 444.

14-31 K.Hagiwara et al., Phys.Rev. D66 (2002) 010001, and references therein.

14-32 H.A. Bethe and W .Heitler, Proc.R.Soc.London A146 (1934) 83.

14-33 V. Innocente, M. Maire, and E. Nagy, CERN program library long writeup W5013-E, GEANE.

14-34 V. Karimaki, “Effective vertex fitting”, CMS Note 1997/051.

14-35 CMS Coll.,“The Trigger and Data Acquisition project, Volume I, The Level-1 Trigger, Technical Design Report”, CERN/LHCC 2000-038, CMS TDR 6.1, 15 December 2000.

15 Physics Object Selection and HLT Performance

This chapter discusses the reconstruction and selection of physics objects in the High-Level Trigger. These objects constitute candidates for electrons, muons, neutrinos (as missing energy), and hadronic jets, as well as jets tagged as originating from tau leptons or *b*-quarks. For each object, the efficiency and purity of the reconstruction and selection are estimated. In the end, a "trigger table", i.e. a set of kinematic requirements on each object, leading to an output rate to storage consistent with the current capability of $O(10^2)$ Hz, is produced.

The chapter is organized as follows: an overview of the physics requirements and the general guidelines used in the development of the HLT algorithms can be found in Section 15.1. The following sections describe the reconstruction algorithms, the selection and the CPU requirements for the identification of each physics object. Section 15.7 describes the samples and procedures needed for the calibration of the CMS subdetectors, as well as for the monitoring of the performance of the HLT. The chapter concludes with a summary of all thresholds for selecting physics objects for storage, the overall CPU requirements of the HLT and the efficiency for some key physics channels in Section 15.8.

15.1 Overview of Physics Reconstruction and Selection

The key aspect of the High-Level Trigger which runs in the Filter Farm is the real-time nature of the selection. This imposes significant constraints on the resources that the algorithms can use and on the reliability of these algorithms. Events rejected by the HLT, with the exception of very small samples retained for monitoring of the HLT performance, are lost forever. The correct functioning of the HLT is therefore a key issue for the CMS physics program.

The next section discusses the physics requirements of the online selection while the strategy used in implementing the software is described next. The allocation of bandwidth to different physics objects at Level-1 is the subject of the last section.

15.1.1 Physics Requirements

The main requirements from the physics point of view are:

- foremost, the requirement that the selection fulfill the needs of the CMS physics program; the efficiency for the physics objects must be as high as possible;
- the selection must be as inclusive as possible. The LHC represents a new energy frontier, and unexpected new phenomena may appear. The selection requirements must not reject events of potential use in such exotic searches;
- all thresholds and other requirements applied by the selection should be such that detailed knowledge of calibration constants and other run conditions is not necessary in real-time;
- the final selection of events should include data samples for the calculation of all trigger and reconstruction efficiencies offline;
- the rate of events accepted by the HLT should be within limits, i.e. approximately $O(10^2)$ Hz;
- all algorithms and code should be monitored continually for their correct functioning;
- the events selected by the HLT should be tagged to indicate the reasons for their selection, to aid the offline reconstruction;

- the HLT should include all major improvements in the offline reconstruction, and thus it should be as close as possible to the standard offline reconstruction code.

Above all, the HLT selection must be reliable and robust. Every event that will be used by CMS for the very rich physics program it expects to carry out must be accepted by the High-Level Trigger. The HLT is thus a crucial element for the success of the CMS experimental program.

15.1.2 Selection Strategy and Reconstruction on Demand

As discussed in Chapter 1, "Overview", the CMS DAQ/HLT complex processes all events accepted by the Level-1 Trigger in a single processor farm. There is therefore no “Level-2” or “Level-3” trigger in CMS, but a single entity, the “High-Level Trigger”, or HLT. Nevertheless, as in a traditional multi-level system, the selection of events can be optimized by rejecting events as quickly as possible. For this reason the basic reconstruction strategy is to reconstruct those parts of each physics object that can be used for selection while minimizing the overall CPU usage.

As an example, as described in Section 15.2, "Electron/Photon Identification", reconstruction of an electron includes the reconstruction of a cluster in the electromagnetic calorimeter, the matching of hits in the pixel detector and the subsequent reconstruction of a full charged particle track in the tracker. At the end of each step a set of selection criteria results in the rejection of a significant fraction of the events accepted by the previous step. The rate of events that need to be processed through the remaining algorithms is decreased thus reducing the required CPU. Reconstruction and selection are therefore closely intertwined in the online environment of the filter farm. For an optimal system the HLT should reconstruct the minimum amount of detector information needed to apply a set of selection criteria that reject background events while keeping the desired physics events for further processing.

15.1.2.1 Trigger Levels — Definitions

The reconstruction and selection in the HLT takes place in steps which correspond roughly to what would have been distinct trigger systems, the Level-2 and Level-3 trigger systems. It is thus convenient to use the terminology, and to refer to a “Level-2 trigger” or a “Level-3 step” to describe the selection algorithms and criteria of the HLT. Clearly, in the case of the CMS HLT there is no sharp division between these trigger steps, other than the order in which they are applied.

In what follows, the convention used is that “Level-2” triggers, algorithms and requirements refer to the first selection step in the HLT process. Typically, a Level-2 trigger, which has the maximum rate of events input to it, uses information from only the calorimeter and muon detectors. In contrast, “Level-3” refers to selection that includes the reconstruction of full tracks in the tracker. Traditionally, because of the high number of channels, the complex pattern recognition and higher combinatorics, track reconstruction is a process that demands large amounts of CPU time. Carrying the analogy even further, in what follows there are references to “Level-2.5” triggers, which refer to algorithms that use partial tracker information, e.g. pixel hits for a fast confirmation of the electron candidate. The numbering, “2.5”, attempts to indicate the intermediate nature of the selection, as one that occurs between the selection that is based solely on the calorimeter information, and the selection that is based on the full CMS detector information.

As described in the previous chapter, reconstruction of charged particle tracks in the CMS detector has turned out to be a task that demands less CPU time than is normally expected, and for this reason some of the selection algorithms may change in the future. As an example, if full charged-track reconstruction ever becomes faster than the fast pixel reconstruction in the electron selection, the two steps will be interchanged in the HLT. This flexibility is the major advantage of executing the full selection in a single proc-

essor farm: the system can make optimal use of the most recent advances not only in computing but also in the algorithms and software developed in the experiment.

15.1.2.2 Partial Event Reconstruction

To minimize the CPU required by the HLT, a key feature of the algorithms is to reconstruct the information in the CMS detector only partially. In many cases the decision on whether an event should be accepted by the HLT involves the reconstruction of quantities in only a limited region of the detector. As an example, for an event accepted by the Level-1 Trigger in the inclusive muon stream, only the parts of the muon chambers pointed to by the Level-1 Trigger information, and the corresponding road in the tracker, need be considered for the validation of the muon.

The idea of partial event reconstruction has been embedded in the CMS reconstruction code from the very beginning. In the case of the HLT, the reconstruction of physics objects is driven by the corresponding candidates identified by the Level-1 Trigger. Except in a few cases, which are indicated, all reconstruction described in this chapter starts from the Level-1 Trigger information. This approach leads to significant CPU savings, however it also leads to rejecting events that contain “volunteer” objects, i.e. objects that did not pass the Level-1 Trigger. This disadvantage is mitigated by the fact that it is, in general, very difficult to understand the properties of such objects in later offline analyses. It is clearly always feasible, albeit with an increased CPU cost, to look for additional objects in each event if a particular physics analysis raises such a requirement.

15.1.3 Level-1 Trigger Settings and Rates

The performance of the Level-1 Trigger system is summarized in Appendix F, "Summary of Level-1 Trigger" while the full details of the system can be found in the Trigger TDR [15-1]. There have been some minor changes in rates and efficiencies with respect to that document. The changes reflect the new Monte Carlo input, including an updated version of the PYTHIA Monte Carlo (to version 6) and much increased statistics in the Monte Carlo samples generated, as well as a more detailed and tuned detector simulation. The main thrust of the conclusions of reference [15-1] remains unchanged: the overall performance of the system satisfies the requirements of the CMS physics program.

Another change with respect to the Trigger TDR is the definition of the low luminosity run conditions: “low” luminosity in this document refers to an instantaneous luminosity of 2×10^{33} $cm^{-2}s^{-1}$, while “high” luminosity refers to the nominal LHC design value of 10^{34} $cm^{-2}s^{-1}$. As discussed in Chapter 17, the current plans of CMS call for a phased installation of the Data Acquisition system so that at start-up the DAQ can handle an event rate of up to 50 kHz and at the full design luminosity it can handle a rate of 100 kHz.

The allocation of the Level-1 Trigger bandwidth must be optimized in order to ensure the widest possible physics reach for the experiment, while including all technical triggers intended for calibration and monitoring of the detector. The procedure used for this optimization starts with a determination of the maximum Level-1 Trigger rate that can be allocated, which is 16 kHz at low luminosity and 33 kHz at high luminosity. These maximum rates arise from dividing total DAQ bandwidth, i.e. 50 kHz (100kHz) at low (high) luminosity, by a safety factor of three. This safety factor is used to account for all uncertainties in the simulation of the basic physics processes, the CMS detector, and the beam conditions.

The second step is a first allocation of this Level-1 bandwidth across the different objects. An equal allocation across the four categories of “objects”, taken as (a) electrons/photons, (b) muons, (c) tau-jets and (d) jets plus combined channels, is made. For example, at low luminosity 4 kHz is pre-allocated to each of these triggers.

The third step is to determine thresholds for the single and double-object triggers within this first Level-1 rate allocation. As an example, the 8 kHz of electron and photon triggers at high luminosity has to be divided among single and double electrons/photons. For this purpose the isorate contours on the plane of the E_T cut[1] on the single objects vs the E_T cut on the double objects are examined, as in Figure 15-1. An operating point in this single-double threshold plane must be selected for a given a Level-1 Trigger allocation. To determine the optimal operating point, the efficiency for triggering on selected physics channels, via the single and the double-object trigger, is considered. In the case of the electron/photon trigger the efficiency for $W \rightarrow e\nu$ decays vs the efficiency for $Z \rightarrow ee$ decays can be used, as in Figure 15-1 (right). The optimal point can be selected as a trade-off between the two efficiencies, e.g. adopting the point at which both efficiencies drop at the same rate, as shown in the same figure.

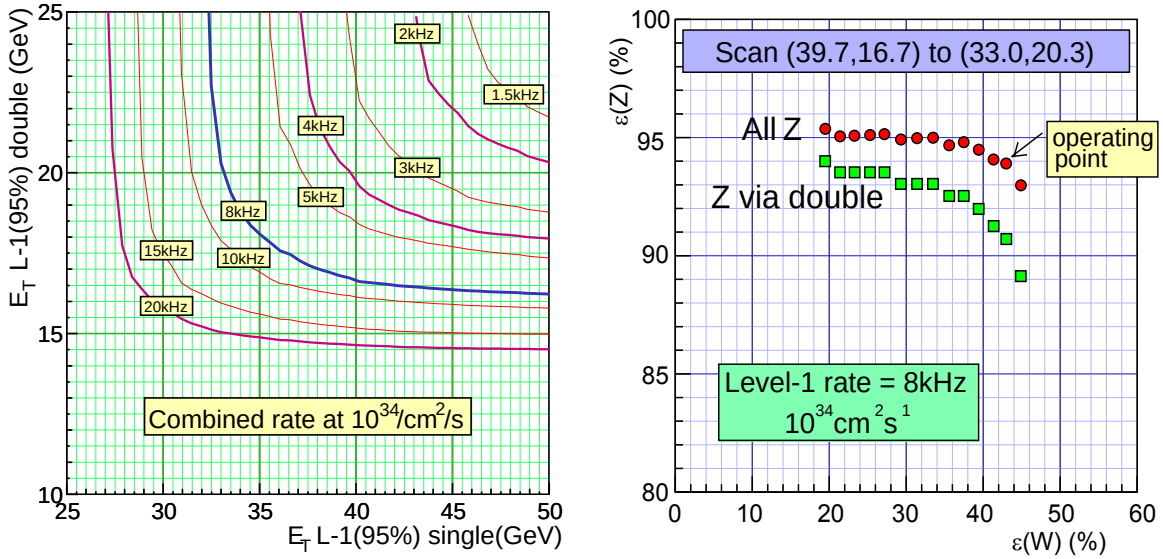


Figure 15-1 Contours of equal rate on the plane of E_T thresholds for single and double-electron/photon triggers (left), and efficiency for *W* and *Z* electronic decays as a function of the same thresholds.

There is no similar optimization of the allocation for jet triggers since there is no reason to include a 2-jet trigger requirement[2].

The last step in this procedure is to evaluate the optimal efficiencies for each of the four object categories and to then re-iterate on the entire procedure with slightly different Level-1 Trigger allocations.

15.1.3.1 Level-1 Trigger Table

With the constraints described previously, the Level-1 Trigger table that is used in the present TDR for further analysis of the HLT algorithms and performance is listed in Table 15-1. The thresholds quoted correspond to the E_T or P_T value at which the efficiency of the trigger is 95% of its maximum value. There is no entry for a muon+jet trigger at low luminosity because, as explained in Section 15.5.9, there is

1. The energy scale of the Level-1 electromagnetic trigger has been calibrated by deriving a linear relationship between the uncalibrated number from the trigger output and the electron transverse momentum at which the efficiency for the trigger reaches 95% of its plateau value. All thresholds given for the electromagnetic trigger are, in fact, 95% efficiency points.
2. All QCD events have a second jet which, though by definition at a lower E_T than the leading jet, is always present in the event. This is very different from di-electron and di-muon signatures.

little to be gained given the low inclusive muon threshold. The corresponding trigger table for high luminosity is listed in Table 15-2. In both cases, a 1 kHz bandwidth is allocated to minimum-bias events which will be used for calibration and monitoring purposes.

Table 15-1 Level-1 Trigger table at low luminosity. Thresholds correspond to values with 95% efficiency.

Trigger	Threshold (GeV or GeV/c)	Rate (kHz)	Cumulative Rate (kHz)
Inclusive isolated electron/photon	29	3.3	3.3
Di-electrons/di-photons	17	1.3	4.3
Inclusive isolated muon	14	2.7	7.0
Di-muons	3	0.9	7.9
Single tau-jet trigger	86	2.2	10.1
Two tau-jets	59	1.0	10.9
1-jet, 3-jets, 4-jets	177, 86, 70	3.0	12.5
Jet * E_T^{miss}	88 * 46	2.3	14.3
Electron * Jet	21 * 45	0.8	15.1
Minimum-bias (calibration)		0.9	16.0
TOTAL			**16.0**

Table 15-2 Level-1 Trigger table at high luminosity. Thresholds listed correspond to values with 95% efficiency.

Trigger	Threshold (GeV or GeV/c)	Rate (kHz)	Cumulative Rate (kHz)
Inclusive isolated electron/photon	34	6.5	6.5
Di-electrons/di-photons	19	3.3	9.4
Inclusive isolated muon	20	6.2	15.6
Di-muons	5	1.7	17.3
Single tau-jet trigger	101	5.3	22.6
Two tau-jets	67	3.6	25.0
1-jet, 3-jets, 4-jets	250, 110, 95	3.0	26.7
Jet * E_T^{miss}	113 * 70	4.5	30.4
Electron * Jet	25 * 52	1.3	31.7
Muon * Jet	15 * 40	0.8	32.5
Minimum-bias (calibration)		1.0	33.5
TOTAL			**33.5**

15.2 Electron/Photon Identification

The selection of electrons and photons proceeds in three steps. The first step, “Level-2.0”, uses the Calorimeter information alone. The next step, “Level-2.5”, demands hits in the pixel detectors consistent with an electron candidate. This matching of an ECAL super-cluster (defined in paragraph 15.2.1) to hits in the pixel detector divides the electromagnetic triggers into two streams: electron candidates (single and double), and, above significantly higher thresholds, photon candidates. In the final step, “Level-3”, the selection of electrons uses full track reconstruction, seeded from the pixel hits obtained by the matching step. The selection of photons can use isolation cuts and rejection of π^0s based on lateral shower shape and the reconstruction of converted photons.

For the calorimeter reconstruction — clustering as well as energy and position measurement — the emphasis is predominantly on the reconstruction of electrons, rather than photons, because the target transverse momentum cuts and thresholds for triggering on electrons are much lower than those for photons. The first challenge for ECAL clustering is to include all energy radiated by electrons in the tracker material. Photons, including those that have converted in the tracker material, are adequately reconstructed by the electron algorithms for threshold cuts to be applied. For the final reconstruction of photons with high precision it is envisaged that unconverted photons will be reconstructed using energy sums of fixed arrays of crystals (probably 5×5 crystals), and that tracker information will be used to assist in the clustering for converted photons. These algorithms to refine photon energy resolution are not needed for the HLT selection, and have not yet been fully developed, although much of the necessary work has been started in studies of π^0 rejection.

15.2.1 Calorimeter Reconstruction: Clustering

The first step in the reconstruction of an electron in the High-Level Trigger is the clustering of the energy deposits in the electromagnetic calorimeter (ECAL) and the estimation of the electron’s energy and position from this information. In the barrel section this involves the energy deposited in the lead tungstate crystals alone, in the endcap energy is also deposited in the ~$3X_0$ thick preshower detector.

Electrons radiate in the material between the interaction point and the ECAL. The bending of the electron in the 4T magnetic field results in a spray of energy reaching the ECAL. The spreading of this spray is, to good approximation, only in the φ-direction. The electron energy can be collected by making a cluster of clusters along a φ road. This cluster of clusters is called a “super-cluster”.

15.2.1.1 The Island Algorithm

The island algorithm starts by a search for seeds which are defined as crystals with an energy above a certain threshold. Starting from the seed position, adjacent crystals are examined, scanning first in φ and then in η. Along each scan line crystals are added to the cluster until a rise in energy or crystal that has not been read out is encountered.

In much the same way as energy is clustered at the level of calorimeter cells, non-overlapping clusters can in turn be clustered into calorimetric “super-clusters”. The procedure is seeded by searching for the most energetic cluster and then collecting all the other nearby clusters in a very narrow η-window, and much wider φ-window.

15.2.1.2 The Hybrid Algorithm

For single showers, such as those produced by unconverted photons, or those produced by electrons in test beam conditions, energy sums of fixed arrays of crystals give better energy resolution performance than energy sums of crystals collected dynamically according to a cluster or "bump" finding algorithm. The Hybrid algorithm uses the $\eta-\varphi$ geometry of the barrel crystals to exploit the knowledge of the lateral shower shape in the η direction (taking a fixed bar of three or five crystals in η), while searching dynamically for separated (bremsstrahlung) energy in the φ direction.

The Hybrid algorithm is designed to reconstruct relatively high energy electrons in the barrel (so far we have used it for electrons with P_T>10 GeV/c). By contrast, when looking for small deposits of energy in individual clusters, for example when making a calorimetric isolation cut, the basic clusters of the Island algorithm are more appropriate objects to work with. Further details about the clustering algorithms can be found in reference [15-2].

15.2.2 Endcap Reconstruction with the Preshower

Much of the endcap is covered by a preshower device with two planes of silicon strip readout. The energy deposited in the preshower detector (which is about $3X_0$ thick) needs to be added to the crystal clusters. The constants relating the energy deposited in the two planes of silicon to the crystal energy are explained in reference [15-3].

The energy in the crystals is clustered using the Island algorithm and the clusters are associated to form super-clusters. A preshower cluster is constructed in each plane, in front of each crystal cluster of the super-cluster. The search area in the preshower is centred on the point determined by extrapolating the crystal cluster position to the preshower plane in the direction of the nominal vertex position.

15.2.3 Energy and Position Measurement

15.2.3.1 Position Measurement Using Log-weighting Technique

A simple position measurement of the shower can be obtained by calculating the energy-weighted mean position of the crystals in the cluster. To obtain a precise position measurement two issues need to be considered in more detail.

Firstly, the meaning of crystal position needs to be defined. The crystals in the CMS ECAL are quasi-projective, and do not exactly point to the nominal interaction vertex. So the lateral position (η,φ) of the crystal axis depends on depth as illustrated in Figure 15-2. A depth t_{max} thus needs to be defined. This depth is something like the longitudinal centre of gravity of the shower, and its optimal mean value varies logarithmically with the shower energy. There is also a dependence on particle type — electron showers have a maximum about one radiation length less deep than photon showers. In the position measurement used for both Island and Hybrid super-clusters the depth is measured from the front face of the crystals along the direction from the nominal vertex position to the approximate shower position calculated using the arithmetic energy weighted mean of the shower front face centres.

To account for the energy dependence a parametrization is used: $A(B + \log(E))$. The determination of the parameters is described in reference [15-2].

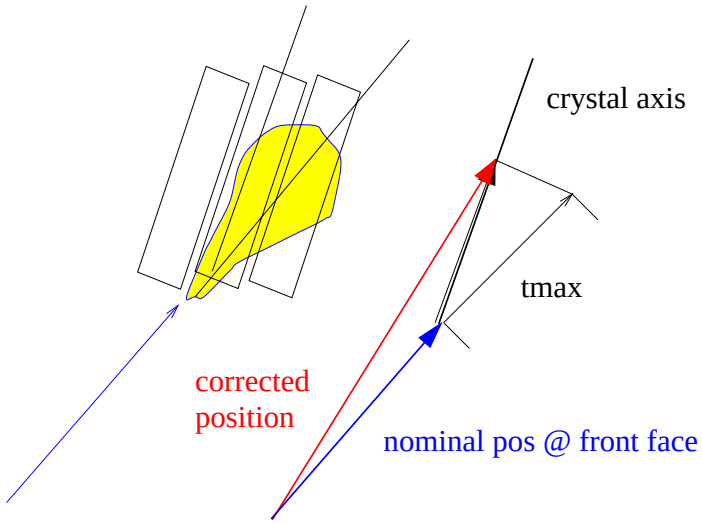


Figure 15-2 Illustration of crystal off-pointing.

The second issue that requires more detailed treatment is related to the lateral shower shape. Since the energy density does not fall away linearly with distance from the shower axis, but rather exponentially, a simple energy weighted mean of crystal energies is distorted and the measured position is biased towards the centre of the crystal containing the largest energy deposit.

In earlier studies this bias was removed by applying a position correction (the “S-shape correction” described in reference [15-4]). A simpler algorithm which yields almost as good precision is used here. The weighted mean is calculated using the logarithm of the crystal energy:

$$x = \frac{\sum x_i \bullet W_i}{\sum W_i}$$

where x_i is the position of crystal *i*, and W_i is the log weight of the crystal — the log of the fraction of the cluster energy contained in the crystal, calculated with the formula:

$$W_i = W_0 + \log\left(\frac{E_i}{\sum E_j}\right)$$

where the weight is constrained to be positive, or is otherwise set to zero. W_0 then controls the smallest fractional energy that a crystal can have and still contribute to the position measurement. More details can be found in reference [15-2].

So far what has been described refers to measurement of the position of a single cluster. The position of a super-cluster is calculated by making the energy-weighted mean of the positions of its component clusters. An important point at issue here is that the energy-weighted mean impact position of an electron and the photons it has radiated in its passage through the tracker material corresponds to the position a non-radiating electron would have impacted. It is this position which we wish to measure to enable the correct reconstruction of the original electron from the vertex. So, although it is unimportant for the energy measurement whether the crystal energy deposits from the electron and its radiated photons are all collected into a single cluster or into separate clusters, correctly clustering the energy deposits improves the φ measurement of an electron. If the electron and a bremsstrahlung photon fall into the same cluster then the weighting of their contributions used is the log weighting appropriate to lateral shower spread, and not the linear weighting appropriate between the electron and a photon it radiated.

15.2.3.2 Energy Measurement and Corrections

The measurement of energy in the crystals is obtained by simple addition of the deposits measured in the crystals — although more complex estimators have been envisaged [15-5].

Even in the areas not covered by the preshower detector the energy containment of the clustered crystals is not complete. The reconstructed energy distribution, E_{meas}/E_{true}, shows a peak at a few percent less than unity, and a long tail on the low side due to unrecovered bremsstrahlung energy. The Gaussian part of the distribution corresponds, roughly, to the energy that would be reconstructed from an electron in the absence of bremsstrahlung. The amount of tracker material varies strongly with η, and thus so does the amount of bremsstrahlung radiation, so a variation in the fraction of events in the tail as a function of η is expected. This inevitably leads to a small variation in the peak position as a function of η.

The energy scale is "calibrated" using corrections designed to place the peak in E_{meas}/E_{true} at 1.0. The corrections are parameterized in terms of the number of crystals in the cluster ($f(N_{cry})$ corrections). This helps to minimize the residual dependence on both E and η of the energy scale. Figure 15-3 shows, as an example, E_{meas}/E_{true} as a function of the number of crystals in a reconstructed Hybrid super-cluster, for electrons with $10 < P_T < 50$ GeV/c, together with a fitted polynomial function.

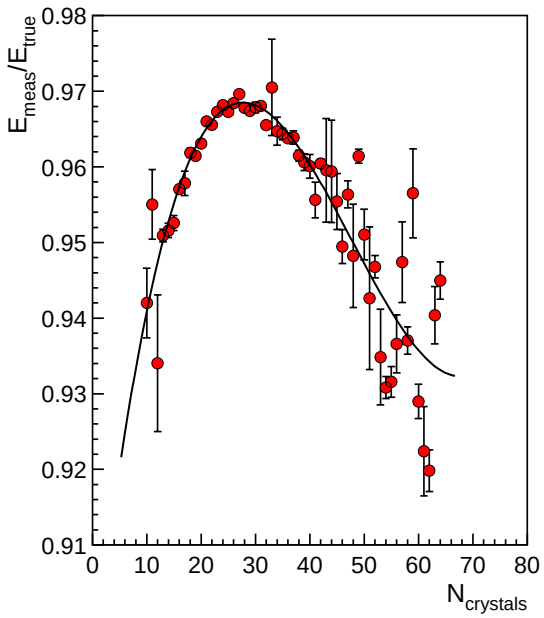


Figure 15-3 E_{meas}/E_{true} as a function of the number of crystals in a Hybrid super-cluster together with a fitted polynomial function.

15.2.3.3 Energy and Position Measurement Performance

The performance figures given in this section have been obtained using CMSIM/GEANT3 simulation and the CMS125 geometry described in Section 14.2.2. For these performance figures an isolated Level-1 Trigger is demanded, which results in a ~8% inefficiency mainly due to bremsstrahlung radiation in the isolation region (~6%), but no trigger threshold is applied.

Figure 15-4 shows the distribution of E_{meas}/E_{true} for $P_T = 35$ GeV/c electrons reconstructed using the Hybrid algorithm in the barrel, and using the Island algorithm and the preshower in the endcap. The energy

resolution is parametrized in two ways: in terms of the fitted width of the Gaussian part of the distribution (fitted between -1.5σ and $+2.0\sigma$), and in terms of σ_{eff}, defined as the half-width containing 68.3% of the distribution — if the distribution is Gaussian then σ_{eff} is just the Gaussian sigma, if the distribution has more significant tails then σ_{eff} provides some measure of this. The parameter σ_{eff} provides a convenient measure of performance which adequately reflects final physics performance (in extracting signal significance etc.).

Figure 15-5 shows the position resolution in η and φ for the same sample. Table 15-3 gives a more complete list of performance results for electron reconstruction in both the barrel and endcap. Values for $P_T = 35$ GeV/c electrons are given, together with values for electrons having a flat P_T spectrum in the range $10 < P_T < 50$ GeV/c. The σ_{eff} value is not given for the η resolution because it is, in all cases, the same as the Gaussian fit value. It is worth noting, as a comparison, that unconverted photons with a flat P_T spectrum in the range $10 < P_T < 50$ GeV/c can be reconstructed, in the barrel using a 5×5 fixed window, and in the endcap using a 3×3 fixed window with impact position correction, achieving in both cases a resolution of $\sigma_{eff}/E = 0.9\%$ at low luminosity.

Table 15-3 Energy and position resolution performance for barrel and endcap ECAL using electron samples simulated with different pileup conditions.

electron sample	Energy resolution		Position resolution		
	σ/E	σ_{eff}/E	$\sigma(\eta)$	$\sigma(\varphi)$	$\sigma_{eff}(\varphi)$
Barrel reconstruction; $\lvert\eta\rvert < 1.4442$ (Hybrid algorithm)					
$P_T = 35$ GeV/c (no pileup)	1.1%	2.2%	1.1×10^{-3}	1.7 mrad	2.5 mrad
$P_T = 35$ GeV/c (2×10^{33} cm^{-2}s^{-1})	1.2%	2.3%	1.1×10^{-3}	1.7 mrad	2.5 mrad
$P_T = 35$ GeV/c (10^{34} cm^{-2}s^{-1})	1.5%	2.7%	1.1×10^{-3}	1.9 mrad	2.7 mrad
$10 < P_T < 50$ GeV/c (10^{34} cm^{-2}s^{-1})	1.5%	3.4%	1.2×10^{-3}	2.1 mrad	3.4 mrad
Endcap reconstruction; $1.566 < \lvert\eta\rvert < 2.5$ (Island algorithm, and preshower)					
$P_T = 35$ GeV/c (no pileup)	1.2%	2.1%	1.8×10^{-3}	2.2 mrad	3.4 mrad
$P_T = 35$ GeV/c (2×10^{33} cm^{-2}s^{-1})	1.6%	2.4%	1.8×10^{-3}	2.3 mrad	3.5 mrad
$P_T = 35$ GeV/c (10^{34} cm^{-2}s^{-1})	2.7%	3.4%	2.0×10^{-3}	2.9 mrad	4.2 mrad
$10 < P_T < 50$ GeV/c (10^{34} cm^{-2}s^{-1})	2.9%	4.2%	2.2×10^{-3}	2.7 mrad	5.1 mrad

15.2.4 Level-2.0 Selection of Electrons and Photons

The first step of the High-Level Trigger, using only calorimeter information, is to reconstruct an ECAL super-cluster in a region specified by the Level-1 Trigger. The super-cluster is first required to fall within the precision physics fiducial region of the ECAL, which is obtained by removing the barrel/endcap transition region from the overall coverage of $\lvert\eta\rvert<2.5$ [15-6]. This transition region is strongly shadowed by tracker cables exiting through the gap between the barrel and endcap. The shadowed region comprises the first trigger tower in the endcap. The last two crystals in the barrel are also removed. The excluded area thus covers $1.4442 < \lvert\eta\rvert < 1.5660$.

The super-cluster is then required to have E_T above a threshold which is chosen to give 95% efficiency for electrons at the same point on the E_T scale at which the Level-1 Trigger has 95% efficiency. The same threshold is required for both objects in the double trigger. At 2×10^{33}cm^{-2}s^{-1} the thresholds are 26 GeV

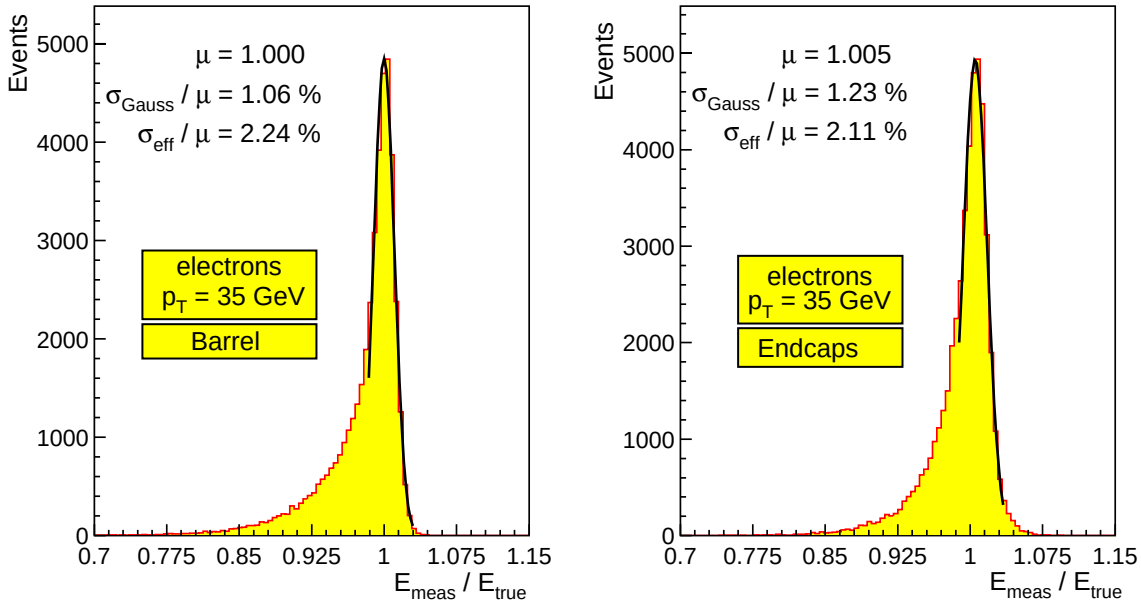


Figure 15-4 Distribution of E_{meas}/E_{true} for P_T = 35 GeV/c electrons, a) in the barrel ECAL fully digitized without pileup, and reconstructed with the Hybrid super-clustering algorithm, b) the same distribution for electrons in the endcap, reconstructed with the Island super-clustering algorithm, and with preshower energy included.

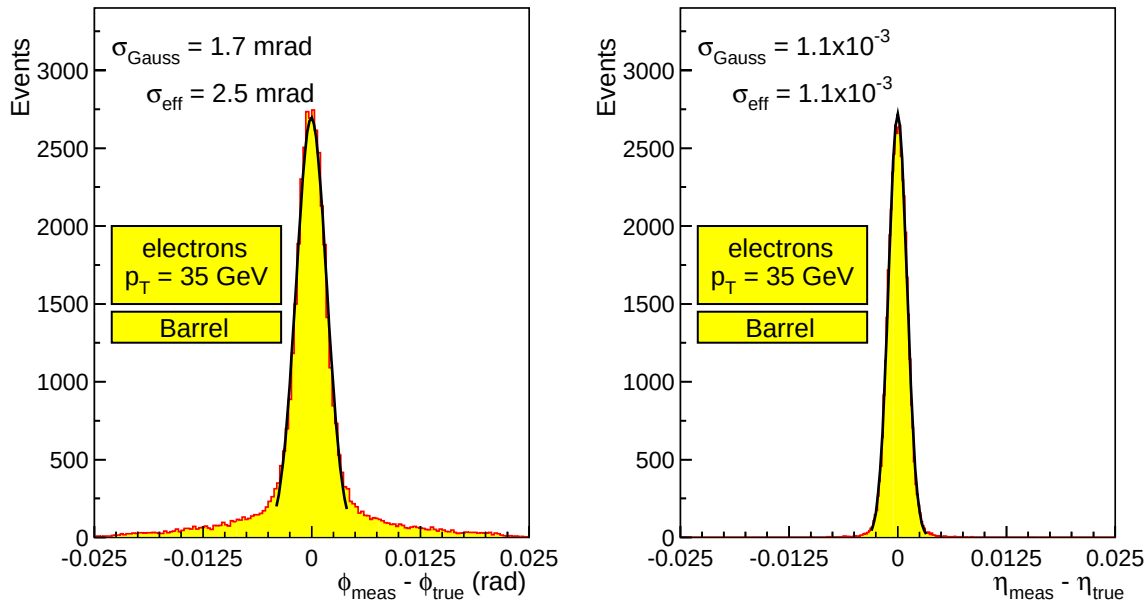


Figure 15-5 Position resolution for P_T = 35 GeV/c electrons in the barrel ECAL, fully digitized without pileup, and reconstructed with the Hybrid super-clustering algorithm.

for the single, and 14.5 GeV for the double. The corresponding thresholds at 10^{34} $cm^{-2}s^{-1}$ are 31 GeV and 16.9 GeV. The rejection obtained from this cut on the transverse energy reconstructed in the ECAL is quite modest — about a factor of two on each object.

Some refinements of this Level-2.0 selection can be envisaged, but have not been used in the basic selection procedure for which results are here presented. Firstly, the Level-2 selection uses cuts on purely ca-

lorimetric quantities — ECAL isolation and an H/E (hadronic/electromagnetic) cut — which could be applied at an earlier stage. Secondly, a significant contribution to the transverse energy resolution comes from the lack of knowledge, at this stage, of the longitudinal vertex position. The contribution to the fractional E_T resolution, as a function of η, due to a 1 σ shift if the vertex longitudinal position (5.3 cm) is shown in Figure 15-6. A small improvement in efficiency would be obtained by making the electron threshold cuts in two stages: a looser cut at Level-2.0 followed by the full set of cuts at a later stage, after the pixel matching of Level-2.5 which gives a precise longitudinal vertex position.

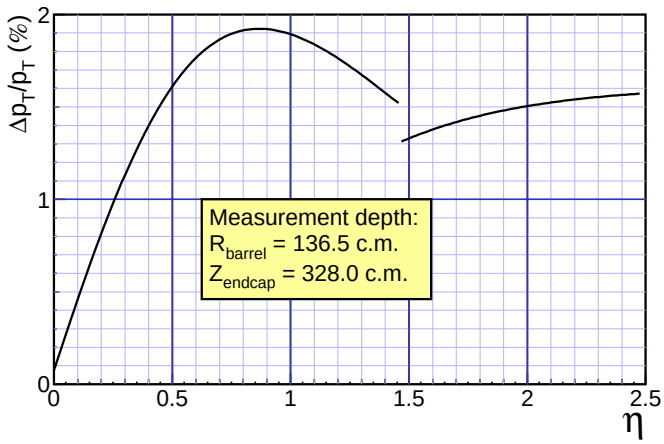


Figure 15-6 Fractional E_T resolution, as a function of η, induced by a 1 σ shift of the longitudinal vertex position.

15.2.5 Level-2.5: Matching of Super-clusters to Hits in the Pixel Detector

The matching of super-clusters reconstructed in the calorimeter to hits in the pixel detector takes advantage of the fact that the energy-weighted average impact point of an electron and the bremsstrahlung photons it has radiated is precisely where a non-radiating electron would have impacted. It is this space-point that the position measurement of the super-cluster attempts to determine. This point can be propagated back through the field to obtain an estimate of the direction of the electron at the vertex, and the hit positions expected in the pixel detector. Since most of the tracker material lies after the pixel detector, most electrons do not radiate significantly before it, and most photon conversions take place after it. So matching hits are given by most electrons and by few photons.

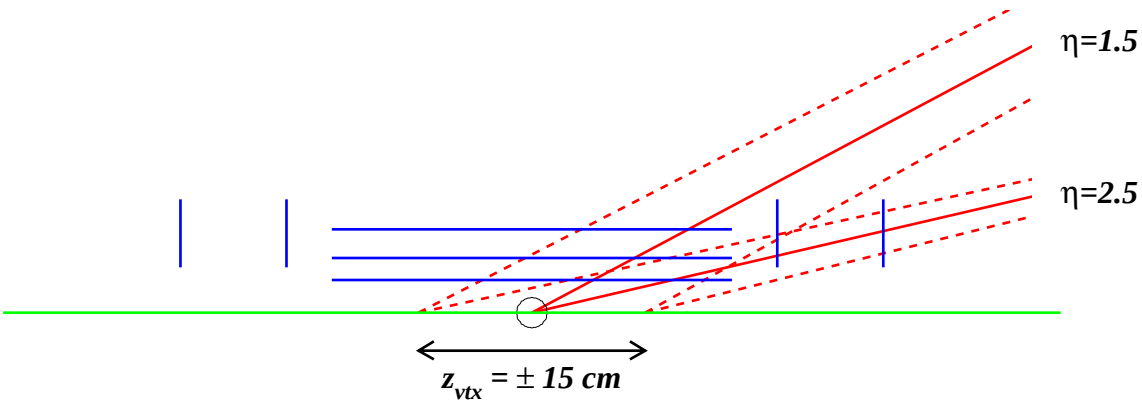


Figure 15-7 Full pixel detector (high luminosity) with continuous lines pointing from the nominal vertex to the edges of the ECAL barrel and endcap, and dashed lines pointing from z = ± 15 cm.

The layout of the pixel system is shown in Figure 15-7. The continuous lines show the maximum η values of the ECAL barrel and endcaps and the dashed lines correspond to electrons generated z = ±15 cm away from the nominal vertex. For the clusters at the ends of the ECAL barrel, pixel disks are needed as well as barrel pixel layers. For endcap clusters the mix of pixel barrel and pixel disk hits depends on both the pseudorapidity and the z-vertex of the electron. With the currently simulated geometry there is a loss of efficiency in the high |η| region of the ECAL endcap where, in some cases, the electron goes through only one pixel layer.

As described in Section 14.2.1, the outer barrel layer and the outermost endcap disks of the pixel detector are “staged” and will not be present at start-up. The simulation of the HLT selection at low luminosity is made by discarding pixel information in these layers.

The method proceeds as follows:

1. The electromagnetic cluster gives the energy and the position of the electron candidate. The transverse momentum is computed from the measured energy and position of the cluster. The expected hit position on the pixel layers is estimated by propagating the electron inwards to the nominal vertex using the magnetic field map. A search area is defined in the innermost pixel layer. It is unrestricted in z, and its width in ϕ (typically 40mrad) is the main parameter used to control the tightness of the pixel matching. The errors on the calculated hit positions are due to the spread in z of the vertex position and the precision of the j measurement in the calorimeter. At this stage the error on the cluster E_T measurement contributes very little to the uncertainty on the expected hit position in the pixel detector.
2. If a compatible hit is found within the search area on the innermost pixel layer, a better estimate of the longitudinal (z) vertex position is obtained by interpolating the line from the cluster through the corresponding hit to the beam line. Nominal values (0,0) for x and y coordinates of the vertex are assumed. If no hit is found in the search area of the innermost pixel layer, the search is repeated in the next layer.
3. The track is propagated from the newly estimated vertex to the next pixel layer through the compatible hit in the first (or second) layer. The dominating uncertainty, in the r-φ plane, now comes from the estimate of the E_T of the electromagnetic cluster, and thus the radius of curvature of the electron track. But this is a very small error since the distance from one pixel layer to the next is short. Indeed, assuming a hit on the r = 4 cm layer, the distance between the hit predictions on the r = 11 cm layer for P_T = 15 GeV/c and P_T = 20 GeV/c electron would be 78 μm — less than the size of a pixel. In the r-z plane, the main uncertainty comes from the vertex z.
4. If another compatible hit is found, the cluster is identified as an electron, if not, it is rejected as a jet. If there are no compatible hits in the current layer, there may be one more pixel layer left, and the search is repeated there.

The search is made twice, once for each charge. In the first step of the search the electron and positron search areas can overlap, but in the second step, when a compatible hit is propagated to another pixel layer, the P_T needed for the search areas of different charges to overlap is almost 1 TeV/c.

For low luminosity both the “staged” and full pixel detector configurations are investigated. Using the inner pixel layers alone results in a large inefficiency (≈7%) in the staged pixel configuration, so the low luminosity staged pixel detector algorithm supplements the pixel hits with hits found in the first silicon strip layer of the tracker, and regains some of the lost efficiency. More details of the Level-2.5 pixel matching algorithm are given in reference [15-7].

The rejection versus efficiency obtained from the Level-2.5 pixel matching is shown in Figure 15-8, at a) low luminosity (2×10^{33} $cm^{-2}s^{-1}$), and b) high luminosity (10^{34} $cm^{-2}s^{-1}$). The efficiency is calculated using

an electron sample passing the Level-2.0 threshold. The rejection is calculated from the single electron triggers in the jet background passing the Level-2.0 threshold, and refers to rejection from the electron stream. In practice, single and double photon streams are created by applying an E_T threshold to the events rejected by the pixel matching.

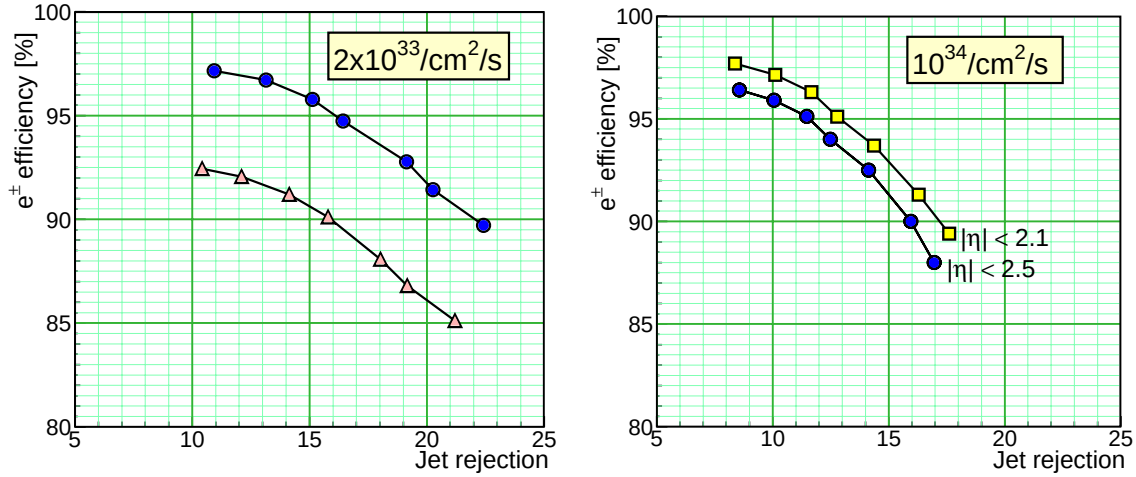


Figure 15-8 Rejection versus efficiency obtained from the Level-2.5 pixel matching. Left: at low luminosity (2×10^{33} $cm^{-2}s^{-1}$); the top curve shows the performance when the full pixel detector is used while the lower curve shows the performance for the staged pixel scenario (see text). Right: at high luminosity (10^{34} $cm^{-2}s^{-1}$); the lower curve is the nominal detector configuration; the top curve corresponds to $|\eta|<2.1$.

For the low luminosity case two different curves of the efficiency versus rejection, corresponding to the full and staged pixel detector configurations, are shown in Figure 15-8(left).

The performance at high luminosity, shown in Figure 15-8(right), is very similar to the low luminosity performance without staging with respect to electron efficiency, but there is a noticeable loss of rejection power which can be attributed to pileup hits. The high luminosity figure also shows the performance in a more restricted central region ($|\eta|<2.1$) where the efficiency is noticeably better. Although not shown in the figure, the same is also true at low luminosity. The curves show seven connected points, in each case the set of points corresponds to the same set of seven cuts in $\Delta\varphi$.

15.2.6 Inclusion of Full Tracking Information: “Level-3” Selection

The Level-3 selection includes all further requirements needed to reach an acceptable rate to final offline storage. The full event information, including tracks, is available, but some of the cuts used — hadronic/electromagnetic energy fraction and calorimetric isolation — use only calorimetric information, and might, in a fully optimized selection chain be made at an earlier stage.

15.2.6.1 Electrons

The Level-3 selection for electrons starts with electron track-finding, seeded by the Level-2.5 pixel match. To maintain high efficiency track-finding is made with very loose cut parameters. Cuts are then made on both E/P and on the distance between the super-cluster position and the extrapolated track position in the ECAL in the longitudinal coordinate, $\Delta\eta$(track - cluster), which is only slightly distorted by

bremsstrahlung. In the endcap a cut on the energy found behind the super-cluster, in the HCAL, expressed as a fraction of the super-cluster energy, H/E, is found to give useful additional rejection

Figure 15-9 shows, as an example, the E/P distribution for barrel electrons, and for jet background electron candidates in the barrel after selection at Level-2.5 followed by loose track finding seeded with the Level-2.5 pixel matches. When the distributions are broken down according to the number of hits associated in the tracks, the width and proportion of events in the tail of the distribution for electrons is found to vary (electrons which radiate little have tracks with more hits, and a better measured momentum). The ratio of signal to background also varies with the number of hits. So increased performance can be obtained by optimizing the E/P cut as a function of the number of hits in the track.

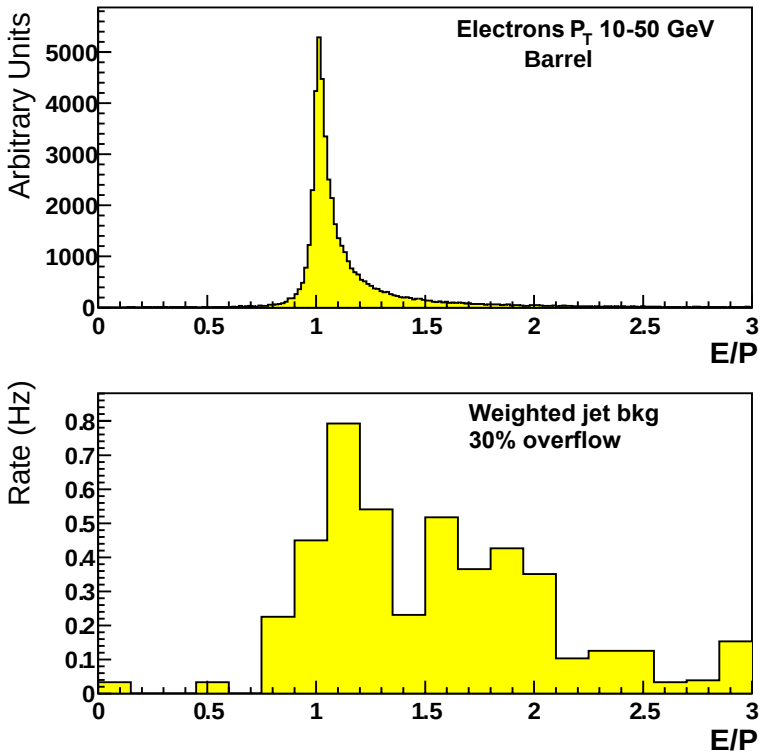


Figure 15-9 E/P for (upper plot) electrons and (lower plot) jet background candidates in the barrel, after Level-2.5 selection followed by track finding seeded by the Level-2.5 pixel hits (2×10^{33} cm^{-2} s^{-1}). The background distribution has 30% overflows.

At high luminosity the additional rejection power of isolation cuts is used to reduce the background to single electrons from jets. Three isolation techniques have been studied: ECAL isolation, pixel-track isolation, and full-track isolation. Track isolation has the advantage that it is less sensitive to pileup, which is the dominant source of signal inefficiency at high luminosity, because only tracks associated with the primary vertex are selected for the isolation cuts. The results presented in the tables below use no isolation for electrons at low luminosity, and a simple pixel-track isolation cut at high luminosity.

Figure 15-10 shows the rejection against jet background versus the efficiency for signal electrons from W's when a pixel-track isolation cut is applied after the Level-2.5 selection at high luminosity. In addition, only events where a track can be made from the pixel hits of the Level-2.5 candidate are used. In practice the isolation cut is applied after the Level-3 selection where the efficiency and rejection power are rather similar, but the size of our jet background event samples are limited and a full mapping out of the efficiency versus rejection curve after Level-3 is subject to larger statistical errors. The different

points on the plot correspond to different cuts on the number of 3-hit pixel lines with P_T > 1GeV/c found within cones of different sizes.

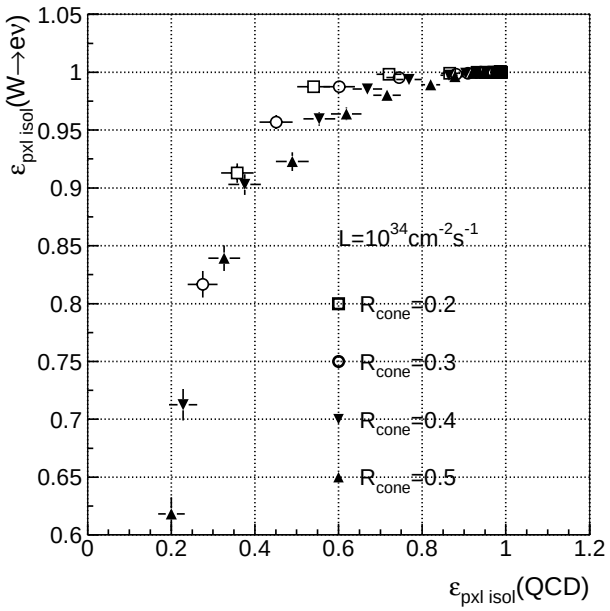


Figure 15-10 Rejection against jet background versus the efficiency for electrons from *W*s when a pixel-track isolation cut is applied after the Level-2.5 selection at 10^{34} $cm^{-2}s^{-1}$.

15.2.6.2 Photons

Further E_T thresholds, higher than those applied at Level-2.0, are applied to super-clusters of single and double triggers that fail the Level-2.5 pixel matching. The events passing these cuts form the photon stream. The double photon thresholds are asymmetric, chosen to be 5 GeV lower than the offline analysis cuts envisaged for the Standard Model H→γγ search [15-6]. The single photon thresholds are chosen to give an acceptable rate. Table 15-4 lists the thresholds and the rates before further Level-3 selection.

Table 15-4 Photon stream thresholds and rates before additional Level-3 cuts.

	Threshold	**Rate @ 2×10^{33} $cm^{-2}s^{-1}$**	**Rate @ 10^{34} $cm^{-2}s^{-1}$**
Single photon	E_T > 80 GeV (2×10^{33}) E_T > 100 GeV (10^{34})	7 Hz	10 Hz
Double photons	E_T^1 > 35, E_T^2 > 20 GeV	85 Hz	382 Hz

Background rejection is available using isolation and π^0 rejection based on lateral shower shape. Track isolation has the advantage that it is less sensitive to pileup at high luminosity, but to take this advantage a vertex must be defined. Defining the longitudinal coordinate of the vertex is a significant issue for the H→γγ signal channel. For events where one or more of the photons has converted in the tracker, the track segment and the ECAL cluster will be used to locate the vertex. The vertices in the remaining events will be found using algorithms that choose the track vertex associated with the largest track activity. These algorithms have not yet been incorporated into the High-Level Trigger selection.

Figure 15-11 shows the efficiency for H→γγ events versus the rejection against jet background events when track isolation is applied at $10^{34}cm^{-2}s^{-1}$.

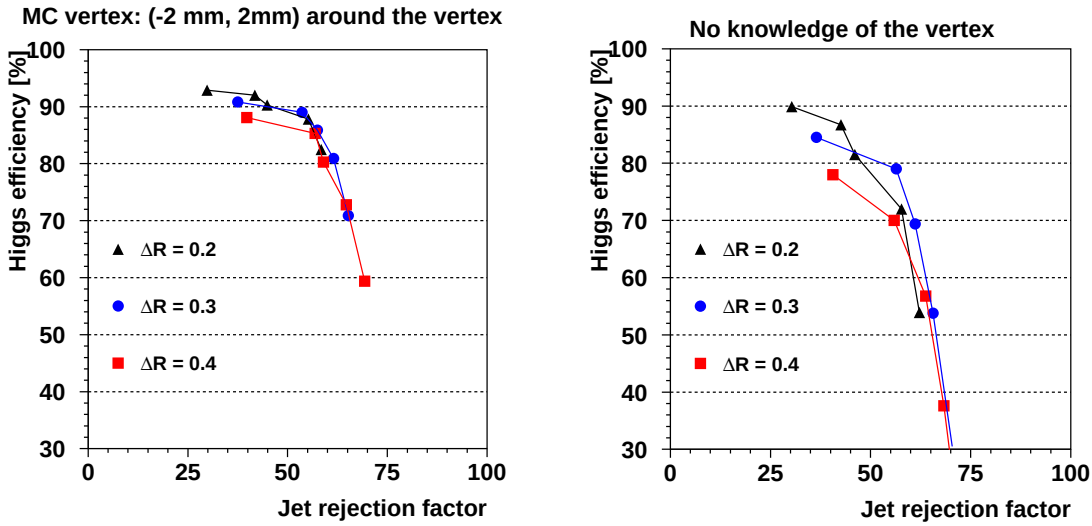


Figure 15-11 Efficiency for $H \rightarrow \gamma\gamma$ events versus the rejection against jet background events when track isolation is applied at 10^{34} $cm^{-2}s^{-1}$. The different points correspond to different choices of isolation cone size and P_T cut on the tracks. The left-hand plot is for the case where the event vertex is known, and the right-hand plot for the case where tracks from all vertices are used.

15.2.7 Summary of Electron and Photon HLT Selection

15.2.7.1 Final Rates to Permanent Storage

The electron and photon rates output by the HLT at both low and high luminosity, broken down by contribution, are listed in Table 15-5. For the low-luminosity selection a loose calorimetric isolation has been applied to the photon streams, but no isolation (beyond that of the Level-1 Trigger) has been applied to the electron streams. To control the two-photon rate the thresholds have been raised to $E_T^1 > 40$ GeV, $E_T^2 > 25$ GeV (equal to the final offline cuts envisaged for H→γγ). This reduces the rate from 11Hz to 5Hz, and has a negligible effect on the efficiency, as is shown in the second column in Table 15-7. A fully optimized selection would probably involve track isolation on the photon streams (wholly or partly replacing the calorimetric isolation and the raised threshold) and track isolation in the single electron stream. This would reduce the total rate to about 26 Hz, of which only half is background, with the introduction of only a small further inefficiency. For the high-luminosity selection, pixel-track isolation has been applied to the electron stream, and full track isolation has been applied to the photon streams (no track with $P_T > 2$ GeV/c in a cone of ΔR=0.2).

15.2.7.2 Signal Efficiencies for Electron and Photon HLT

The streams where the most work is required to control the background rates are the single-electron and double-photon streams, so, the efficiencies for the decays $W \rightarrow e\nu$ and $H \rightarrow \gamma\gamma$ are used as benchmark. Table 15-6 lists the efficiency for single electrons from *W* decay through the complete selection chain, at

Table 15-5 Electron and photon rates output by the HLT at low and high luminosity.

	$2x10^{33}cm^{-2}s^{-1}$			$10^{34}cm^{-2}s^{-1}$		
	Signal	**Background**	**Total**	**Signal**	**Background**	**Total**
Single electron	$W \rightarrow e\nu$: 10Hz	$\pi^{\pm}/\pi^0$ overlap: 5Hz π^0 conversions: 10Hz $b/c \rightarrow e$: 8Hz	33Hz	$W \rightarrow e\nu$: 35Hz	$\pi^{\pm}/\pi^0$ overlap: 15Hz π^0 conversions: 19Hz $b/c \rightarrow e$: 6Hz	75Hz
Double electron	$Z \rightarrow ee$: 1Hz	~0	1Hz	$Z \rightarrow ee$: 4Hz	~0	4Hz
Single photon	2Hz	2Hz	4Hz	4Hz	3Hz	7Hz
Double photon	~0	5Hz	5Hz	~0	8Hz	8Hz
TOTAL:			43Hz			94Hz

$2\times10^{33}cm^{-2}s^{-1}$ and at 10^{34} $cm^{-2}s^{-1}$. Events are preselected requiring the generated electrons to be within the ECAL fiducial region of $|\eta| < 2.5$, with the region $1.4442 < |\eta| < 1.5660$ excluded. The geometric acceptance is approximately 60% and is not included in the efficiency. The second and fourth columns list the efficiencies for electrons that have P_T greater than the Level-1 and Level-2 95% efficiency point.

Table 15-6 Ffficiency for electrons from *W* decay through the complete selection chain.

	Low luminosity		**High luminosity**	
	All fiducial electrons	**Fiducial electrons with P_T > 29 GeV/c**	**All fiducial electrons**	**Fiducial electrons with P_T > 34 GeV/c**
Level-1	63.2%	87.2%	51.1%	83.2%
Level-2	88.8%	99.4%	82.9%	99.3%
Level-2.5	93.1%	94.6%	92.8%	94.1%
Level-3	81%	82%	77%	78%
HLT (Level-2 to Level-3)	67%	77%	59%	73%

The efficiencies at 10^{34} $cm^{-2}s^{-1}$ are only slightly lower. The main difference comes from the loss due to the additional isolation cuts — typically a 5% loss per object.

Table 15-7 lists the efficiency for $H \rightarrow \gamma\gamma$ for a Higgs with mass $M_H = 115$ GeV/c^2 through the complete selection chain, at $2\times10^{33}cm^{-2}s^{-1}$. As in the previous Table, events are preselected, requiring that the generated photons fall within the ECAL fiducial region. The geometric acceptance is 65%. The second column shows the efficiency for events where the two generated photons satisfy, in addition, the cuts currently assumed for offline analysis in this channel — $P_T^1 > 40$ GeV/c, $P_T^2 > 25$ GeV/c.

15.2.7.3 CPU Usage for Electron and Photon HLT

Table 15-8 shows the CPU usage of the HLT selection, benchmarked on 1GHz Pentium III processor, for jet background events at low luminosity. At high luminosity the time taken for the unoptimized global search for ECAL clusters at Level-2.0 is greatly increased and the overall total CPU time per Level-1 event, at a luminosity of 10^{34} $cm^{-2}s^{-1}$ is about three times as large, as at 2×10^{33} $cm^{-2}s^{-1}$. Preliminary results indicate that this clustering time can be reduced by a factor ~10 using regional reconstruction.

Table 15-7 Efficiency for $H \rightarrow \gamma\gamma$ (M_H=115 GeV/c^2) through the complete selection chain, at 2×10^{33} $cm^{-2}s^{-1}$.

	Both photons in fiducial region	Photons passing offline P_T cuts
Level-1	90.8%	92.3%
Level-2	98.7%	99.4%
Level-2.5	93.4%	99.1%
Level-3	92%	92%
Overall (Level-1 × HLT)	77%	83.7%

Table 15-8 CPU usage of the HLT selection at $2\times10^{33}cm^{-2}s^{-1}$.

HLT level	Mean CPU time (ms)
Level-2.0	154 /Level-1 event
Level-2.5	32 /Level-2 event
Level-3	100 /Level-2.5 event
Total	**162 ms/Level-1 event**

Two steps are not included in the values listed in Table 15-8. The sub-detector data will have to be checked for consistency and formatted. This will be done, initially, only for data in the region of the Level-1 Trigger. The time required for this step has not been measured yet, however there is no reason to expect that it will contribute very significantly to the total. The energy in the ECAL crystals has then to be reconstructed from the time samplings. In the Monte Carlo this is done for the entire detector, and the time taken (0.8 s) is dominated by the pre-shower detector (0.5s). Since only data in a small region need be reconstructed, and the formatted data can be sub-divided more finely than a ReadOut unit, the time required for this step will be 100 times less. Figure 15-12 shows the distribution of the total CPU time taken for the full HLT processing of e/γ triggers at $2\times10^{33}cm^{-2}s^{-1}$.

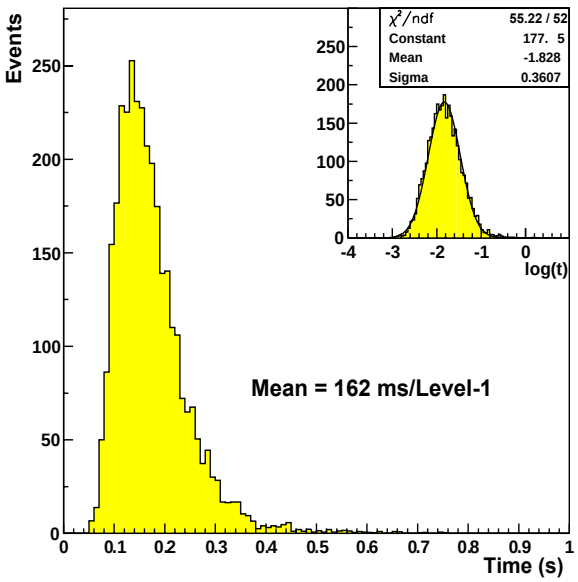


Figure 15-12 Distribution of total CPU time taken by the HLT for the selection of electrons and photons at $2\times10^{33}cm^{-2}s^{-1}$. The inset shows the same time plotted against log(t), and with a Gaussian in log(t) fitted to it.

15.3 Muon Identification

The muon selection for the High-Level Trigger proceeds in two steps: firstly, muons are reconstructed in the muon chambers, which confirms the Level-1 decision and refines the P_T measurement using more precise information; secondly, the muon trajectories are extended into the tracker, which further refines the P_T measurement. After each step, isolation is applied to the muon candidates—the calorimeter being used after the first step and the tracker after the second.

The muon track reconstruction algorithm used by the High-Level Trigger is seeded by the muon candidates found by the Level-1 Global Muon Trigger (no more than four), including those candidates that did not necessarily lead to a Level-1 accept by the Global Trigger. The algorithm uses the reconstructed hits built from the digitized signals in the muon system, and constructs tracks according to the Kalman filter technique, as described in Section 14.4.1. The resulting trajectories are used to validate the Level-1 decision as well as to refine the muon measurement in this "Level-2" muon selection. The basis of the "Level-3" muon selection is to add silicon tracker hits to the muon trajectory, thus greatly improving the muon momentum measurement and sharpening the trigger threshold. Isolation criteria can be applied to the muon candidates to provide additional rejection: at Level-2 using the calorimetric energy sum in a cone around the muon, and at Level-3 using the number of pixel tracks in a region around the projected muon trajectory. This suppresses non-prompt muons from b, c, π, and K decays.

Sections 15.3.1 and 15.3.2 describe the general "tools" that are available to form the muon HLT: muon reconstruction and isolation. Section 15.3.3 describes how these tools are used to form the muon HLT, including additional cuts to suppress fake triggers. Finally, Section 15.3.4 describes the efficiency, rate, and CPU usage of the muon HLT.

15.3.1 Muon Reconstruction

15.3.1.1 Muon Standalone Reconstruction and Level-2 Selection

Reconstructed track segments from the muon chambers are used for muon identification and selection at "Level-2". The state vectors (track position, momentum and direction) associated with the segments found in the innermost chambers are propagated outwards through the iron yoke using the GEANE package [15-8], which takes into account the muon energy loss in the material, the effect of the multiple scattering, and the non-constant magnetic field in the muon system. The estimate of the momentum from the Level-1 Global Muon Trigger is used initially for the track propagation in the magnetic field. The predicted state vector at the next measurement surface is compared with existing measured points and updated accordingly using a Kalman filter technique. In the barrel chambers, reconstructed track segments are used as measurements in the Kalman filter procedure; in the endcap chambers, where the magnetic field is inhomogeneous, it is the individual reconstructed hits belonging to the track segments that are used. Reconstructed hits from RPC chambers are also included. The procedure is iterated until the outermost measurement surface of the muon system is reached, at which point a constrained fit to the track parameters, working from the outside in, is performed under the assumption that the muon candidate originated from the interaction region (defined by the beam spot size: σ_{xy}=15 μm and σ_z=5.3 cm). In both the forward and backward propagations just described, a measurement is not added to the muon trajectory if its contribution to the total χ^2 exceeds 25. The resulting track parameters, propagated inward to the collision vertex, are used to reject or accept the event for further Level-3 processing. More details on this stand-alone muon reconstruction can be found in reference [15-9].

Figure 15-13 shows the resolution of the transverse momentum determined by the Level-2 constrained fit as expressed by the distribution of the quantity $(1/P_T^{rec} - 1/P_T^{gen}) / (1/P_T^{gen})$, where P_T^{gen} and P_T^{rec} are the generated and reconstructed transverse momenta, respectively. Muons from W decays at high luminosity form the reference sample. The distributions are broken up into three pseudorapidity intervals: barrel ($|\eta|<0.8$), overlap ($0.8<|\eta|<1.2$) and endcap ($1.2<|\eta|<2.1$). In these three regions, the fitted P_T resolutions are 10%, 15%, and 16%, respectively.

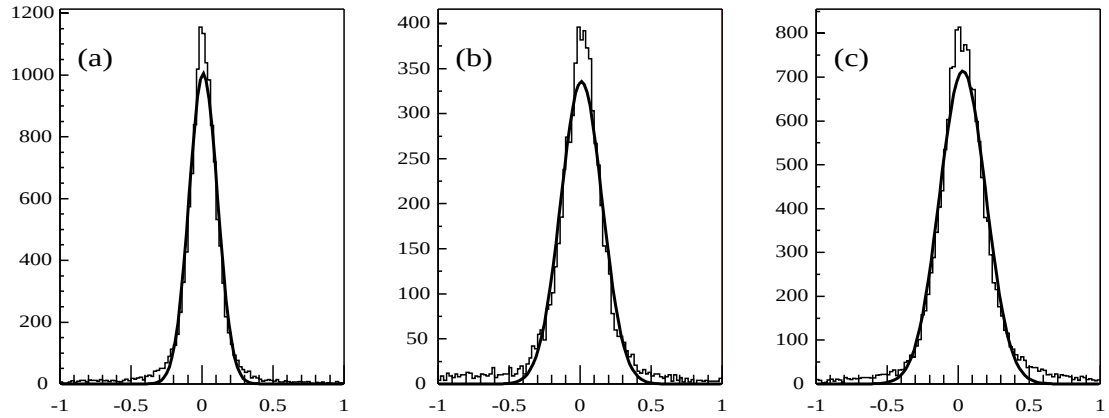


Figure 15-13 Distribution of $(1/P_T^{rec} - 1/P_T^{gen}) / (1/P_T^{gen})$ where P_T^{gen} and P_T^{rec} are the generated and Level-2 reconstructed transverse momenta respectively, shown in three pseudorapidity intervals: a) $|\eta| < 0.8$, b) $0.8 < |\eta| < 1.2$, and c) $1.2 < |\eta| < 2.1$.

15.3.1.2 Inclusion of Tracker Information and Level-3 Selection

The “Level-3” muon reconstruction consists of extending the muon trajectories to include hits in the silicon tracker system. Starting from a Level-2 reconstructed muon, the muon trajectory is extrapolated from the innermost muon station to the outer tracker surface, taking into account the muon energy loss in the material and the effect of multiple scattering. As with Level-2, the GEANE package is currently used for the propagation through the iron and calorimeters. Silicon layers compatible with the muon trajectory are then determined, and a region of interest within them is defined to perform regional track reconstruction. The determination of the region of interest is based on the track parameters and uncertainties of the extrapolated Level-2 muon trajectory, obtained with the assumption that the muon originates from the interaction point as described in the previous section. This has a strong impact on the reconstruction efficiency, fake rate, and CPU reconstruction time: well measured muons are reconstructed faster and with higher efficiency than poorly measured ones.

Inside the region of interest, initial candidates for the muon trajectory (regional seeds) are built from pairs of reconstructed hits. The two hits forming a seed must come from two different tracker layers, and all combinations of compatible pixel and double-sided silicon strip layers are used in order to achieve high efficiency. In addition, a beam spot constraint is applied to muon candidates above a given transverse momentum threshold to obtain initial trajectory parameters.

Starting from the regional seeds, a track reconstruction algorithm based on the Kalman filter technique described in Section 14.4.1, is used to reconstruct tracks inside the selected region of interest. The track reconstruction algorithm consists of the following steps: trajectory building (seeded pattern recognition), trajectory cleaning (resolution of ambiguities) and trajectory smoothing (final fit). In the first step, the trajectory builder transforms each seed into a set of trajectories. Starting from the innermost layer, the trajectory is propagated to the next tracker layer that is reachable, and updated with compatible measurements

found on that layer. In the second step, the trajectory cleaner resolves ambiguities between multiple trajectories that may result from a single seed on the basis of the number of hits and the chi-square of the fit. In the final step, all reconstructed tracks are fit once again with reconstructed hits in the muon chambers included from the original Level-2 reconstructed muon, and selected on the basis of a chi-square cut.

Figure 15-14 shows the resolution of the transverse momentum determined by the Level-3 constrained fit as expressed by the distribution of the quantity $(1/P_T^{rec} - 1/P_T^{gen}) / (1/P_T^{gen})$, where P_T^{gen} and P_T^{rec} are the generated and reconstructed transverse momenta, respectively. Muons from W decays at high luminosity are used as the reference sample. The distributions are broken up into three pseudorapidity intervals: barrel ($|\eta|<0.8$), overlap ($0.8<|\eta|<1.3$) and endcap ($1.3<|\eta|<2.1$). In these three regions, the fitted P_T resolutions are 1.0%, 1.4%, and 1.7%, respectively. The improvement over the stand-alone muon measurement from Level-2 is substantial.

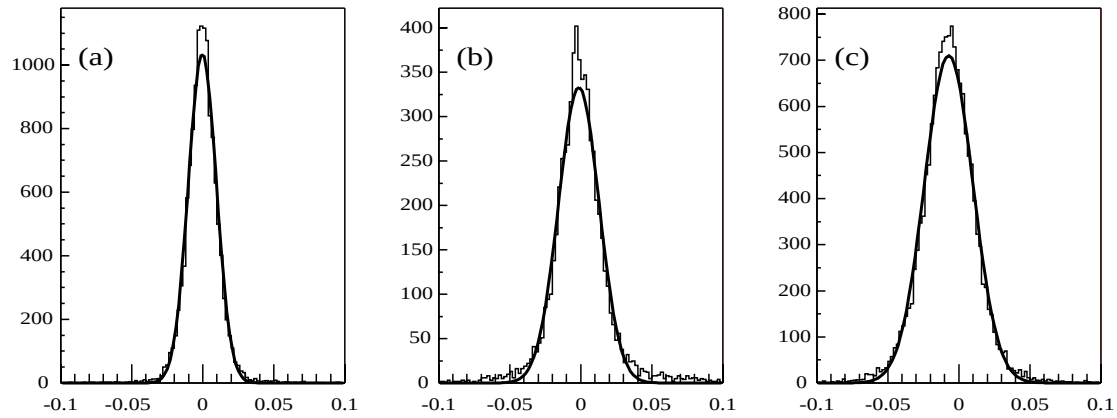


Figure 15-14 Distribution of $(1/P_T^{rec} - 1/P_T^{gen}) / (1/P_T^{gen})$ where P_T^{gen} and P_T^{rec} are the generated and Level-3 reconstructed transverse momenta, respectively, shown in three pseudorapidity intervals: a) $|\eta| < 0.8$, b) $0.8 < |\eta| < 1.2$, and c) $1.2 < |\eta| < 2.1$.

The efficiency of the Level-3 tracking algorithm relative to the Level-2 selection is shown in Fig. 15-15 as a function of η for single muons with $P_T>10$ GeV/c and no pile-up. The muons were generated flat in P_T up to 100 GeV/c. The algorithmic efficiency is typically 99%, except in the pseudorapidity interval $0.8<|\eta|<1.2$, where the DT and CSC systems overlap and the efficiency is about 97%.

15.3.2 Muon Isolation

The integrated rate of muons at LHC is dominated by muons from b, c, K, and π decays, as shown in Figure 14-1. These muons are generally accompanied by other nearby particles, so they can be suppressed by isolation cuts.

Three isolation techniques have been studied. The first (*calorimeter isolation*) is based on the standard technique of summing the calorimeter energy in a cone around the muon, and can be used with the stand-alone muon reconstruction at Level-2. However, as it is based on the calorimeter, this technique becomes less effective at high luminosity as more pile-up is included in the sum. The second technique (*pixel isolation*) is based on the partial reconstruction of tracks in the silicon pixel detector; in this case, isolation is determined on the basis of the sum of the transverse momenta of the tracks in a cone around the muon. This method, which can be applied when tracker information is included at Level-3, is less sensitive to pile-up, as only tracks originating from the same collision vertex are considered. However, it re-

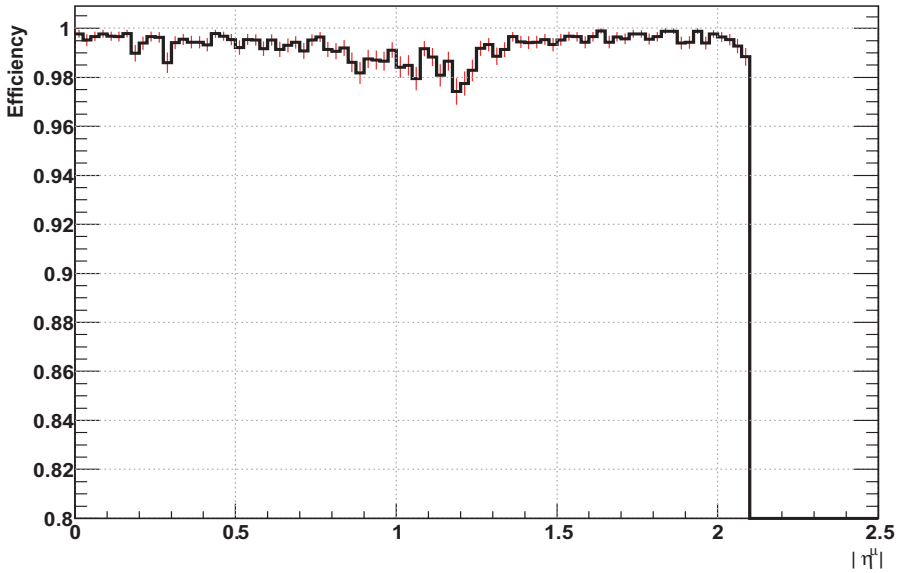


Figure 15-15 Algorithmic efficiency of the Level-3 tracking algorithm as a function of η for single muons generated flat over $10 < P_T < 100$ GeV/c. No pile-up was included.

quires the reconstruction of three pixel hits out of the three layers in the pixel detector for every track; for this reason, it is sensitive to inefficiencies and may not be useful in staging scenarios where only two pixel layers are installed. The third technique, tracker isolation, uses full tracks reconstructed regionally. This method is more robust than pixel isolation, but is more time consuming especially at high luminosity.

For all three techniques, cones are defined by the condition $\Delta R \leq \Delta R_{max}$, where $\Delta R = \sqrt{(\Delta\eta)^2 + (\Delta\phi)^2}$, with $\Delta\eta$ and $\Delta\phi$ the distances from the muon direction in pseudorapidity and azimuthal angle, respectively. The ΣE_T deposited in the cone in the case of calorimeter isolation or the ΣP_T of tracks in the cone in the case of pixel and tracker isolation are computed after subtracting the muon contribution (*veto value*) and compared with a predefined threshold. For each algorithm, both the cone size and the thresholds are chosen by maximising the rejection for a *reference background* sample while keeping the efficiency for a *reference signal* sample above a given nominal value (*nominal efficiency*). The threshold is determined independently in 52 bins in η, in order to guarantee a flat signal efficiency as a function of η.

The rejection power of muon isolation algorithms depends on the P_T of the muon, since low-P_T muons are usually accompanied by low energy jets that may be below the isolation energy or momentum threshold. This is particularly relevant at Level-2, where the feed-through of low-P_T muons contaminates the P_T spectrum even for high thresholds. To exclude these muons, the reference background sample is defined as a sample of minimum-bias events containing only muons with P_T above 22 GeV/c (16 GeV/c) for high (low) luminosity. The direct $W \rightarrow \mu\nu$ decay is used as reference signal since it contains well isolated muons with adequate P_T spectrum.

The result of the optimisation procedure is that for any predefined nominal efficiency value a cone size is chosen, with thresholds defined in bins of pseudorapidity. For all three isolation techniques, typical values of the optimal cone size ΔR_{max} vary from 0.2 to 0.3.

The main features of the three isolation algorithms are briefly described in the following sections. A detailed description can be found in reference [15-10].

15.3.2.1 Calorimeter Isolation

The calorimeter isolation algorithm uses the muon direction at the impact point for the definition of the cone axis. The extraction of the energy deposits is done independently in the ECAL and the HCAL. The total energy is obtained as a weighted sum, $E_T = \alpha\, E_T^{ECAL} + E_T^{HCAL}$, where $\alpha \approx 1.5$ is found to be an optimum value for the isolation technique. Thresholds on E and E_T in individual ECAL crystals and HCAL towers are applied in order to reject noise and pile-up deposits. The energy deposit in a small cone around the extrapolated position of the muon on the ECAL/HCAL boundary is used as veto value and subtracted from the measurement in the cone. Thresholds on the summed E_T vary from 6.5 to 9 GeV for typical cone sizes of 0.2.

15.3.2.2 Pixel Isolation

A pixel reconstruction algorithm looks for pixel hits compatible with tracks with transverse momenta as low as 1 GeV/c. The track candidates are used to fit primary vertices; track candidates with no association to reconstructed vertices are rejected. The algorithm returns a list of vertices with the corresponding tracks and their momenta. All pixel tracks contributing to the ΣP_T in the cone are required to come from the same primary vertex as the Level-3 muon, thus reducing the effect of pile-up. The veto value is defined as the P_T of the muon candidate, i.e. of the pixel track closest in direction to the muon, within $\Delta R<0.015$. Thresholds on the summed P_T vary from 1.8 to 3.8 GeV/c for typical cone sizes of 0.2. The pixel isolation algorithm can also be applied to Level-2 muon candidates, but because the muon trajectory is less well determined than Level-3 candidates, the primary vertex requirement must be dropped and the cone size increased. The performance of the isolation algorithm is therefore different at Level-2 than at Level-3.

15.3.2.3 Tracker Isolation

The tracker isolation algorithm is based on the ΣP_T of tracks reconstructed in a cone around the direction of the Level-3 muon, neglecting the contribution from the muon itself. Tracks are reconstructed using regional tracking, *i.e.* track seeds are created using pairs of pixel hits in a region of interest. The region is defined by a vertex constraint, the minimum transverse momentum for the tracks to be reconstructed as well as a constrain on the track direction at the vertex. Thresholds on the summed P_T varied from 2.0 to 3.0 GeV/c for typical cone sizes of 0.2.

15.3.2.4 Performance

A general investigation of the isolation efficiency for signal muons and rejection of background is reported here, whereas the performance with respect to specific Level-1 and Level-2 triggers is included in Sections 15.3.3 and 15.3.4. The isolation optimisation procedure guarantees by construction that the efficiency for the reference signal ($W\rightarrow\mu\nu$ events) is flat as a function of the pseudorapidity and equal to (or greater than) the chosen nominal efficiency. The efficiency for the background depends on the P_T of the muon, as shown in Fig. 15-16a for the high luminosity case when the efficiency for the reference signal is set to 97%. To show the performance of isolation algorithms, it is therefore necessary to specify the background sample and the minimum generated muon P_T to be taken into account. This is particularly relevant for the Level-2 calorimeter isolation algorithm. In this case, the rate is dominated, for any P_T threshold, by the feed-through of very low-P_T muons, which cannot be rejected by the isolation algorithms but should be rejected by the refined P_T threshold at Level-3. Figure 15-16b shows the rejection power of the isolation algorithms, expressed as the efficiency for background muons in minimum-bias events with $P_T^{gen}>22$ GeV/c at high luminosity, as a function of the pseudorapidity of the muon.

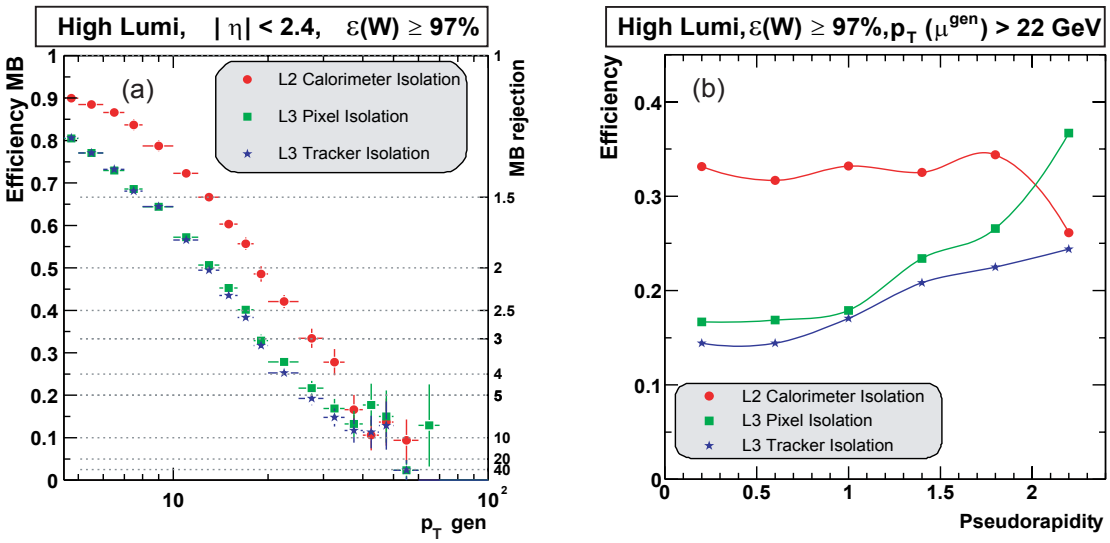


Figure 15-16 Efficiency for background muons to pass the three isolation algorithms as a function of (a) the muon P_T and (b) the muon pseudorapidity, at high luminosity, for a nominal efficiency of 97% to select muons from *W* decays.

The efficiency for reference background muons versus the efficiency for reference signal muons is shown in Fig. 15-17, at low luminosity with P_T^{gen}>16 GeV/c and at high luminosity with P_T^{gen}>22 GeV/c. The background rejection can be adjusted by choosing different efficiencies for the reference signal.

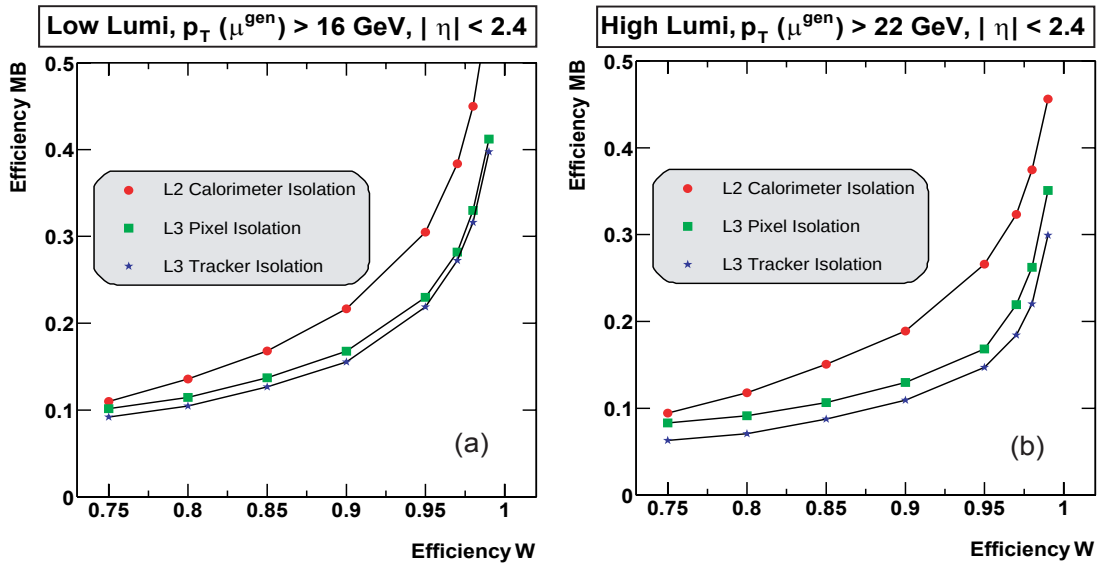


Figure 15-17 Efficiency of the three isolation algorithms on the reference background muons as a function of efficiency for the reference signal muons at (a) low and (b) high luminosity.

The effect of isolation algorithms on the inclusive HLT rates and the required CPU times are presented in later sections.

15.3.3 Muon HLT Selection

Here we describe a prototype inclusive muon trigger based on the reconstruction and isolation tools discussed above to demonstrate the performance possible in the muon HLT. Other combinations of cuts are also possible, as are more sophisticated and more exclusive triggers.

15.3.3.1 Single-muon HLT Selection

A single muon inclusive trigger is formed from the following requirements. At Level-1, low quality CSC tracks must be matched with RPC tracks by the Global Muon Trigger (Section F.2.5) in order to ensure a well-measured P_T. At Level-2, a muon must be reconstructed in the muon system and have a valid extrapolation to the collision vertex. In the barrel region, at least one DT track segment reconstructed as described in Chapter 13 is required, and the sum of the number of DT segments and RPC hits must exceed three. At Level-3, a muon must have more than 5 silicon hits in total from the pixels and silicon strips.

The overall efficiency for muons to pass the Level-1 through Level-3 single muon trigger criteria cumulatively as a function of the generated η is shown in Fig. 15-18. Muons were generated flat in the intervals $5<P_T<100$ GeV/c and $|\eta|<2.1$ without any pile-up. The average combined Level-1 through Level-3 efficiency without any requirements on the reconstructed P_T is 97%, but is lower in some particular regions because of gaps in the geometrical coverage of the chambers.

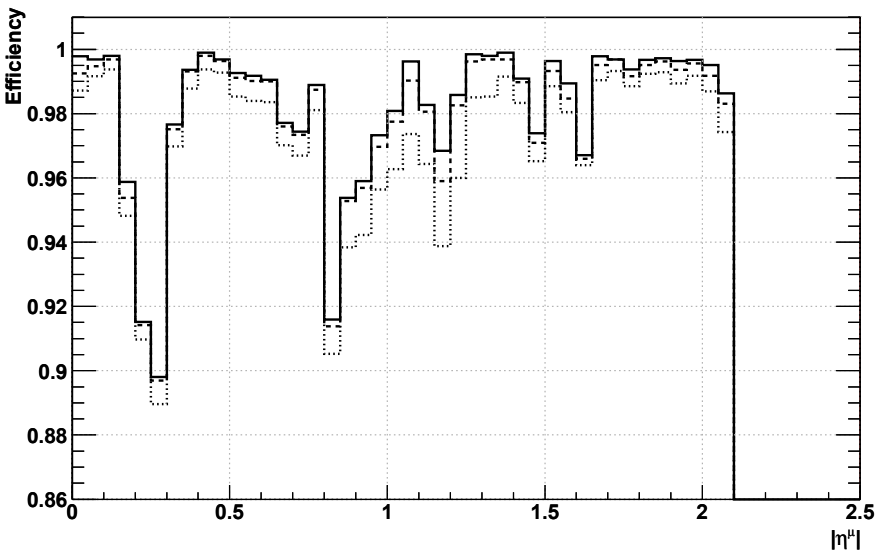


Figure 15-18 Cumulative efficiency for single muons to pass the Level-1 (solid), Level-2 (dashed), and Level-3 (dotted) triggers as a function of the generated muon pseudo-rapidity. No thresholds on P_T are applied. Note the suppressed zero on the y-axis. The dips at $|\eta| \sim 0.3$ and 0.8 are due to gaps in the muon chamber coverage.

The efficiency turn-on curves as a function of the generated P_T for several different P_T thresholds are shown in Fig. 15-19. The efficiency shown is the cumulative Level-1 through Level-3 efficiency. The threshold at each trigger level is defined at 90% efficiency (relative to the plateau efficiency), and it can be seen that the improved P_T resolution at each successive level sharpens the turn-on curve. The efficiency at Level-3 for high P_T muons varies from 97% (no threshold) to 90% ($P_T>40$ GeV/c).

Additionally, for the HLT trigger, Level-2 muon candidates must satisfy the calorimeter isolation criteria at the 97% efficiency point for the reference signal. At Level-3, candidates must satisfy the tracker and

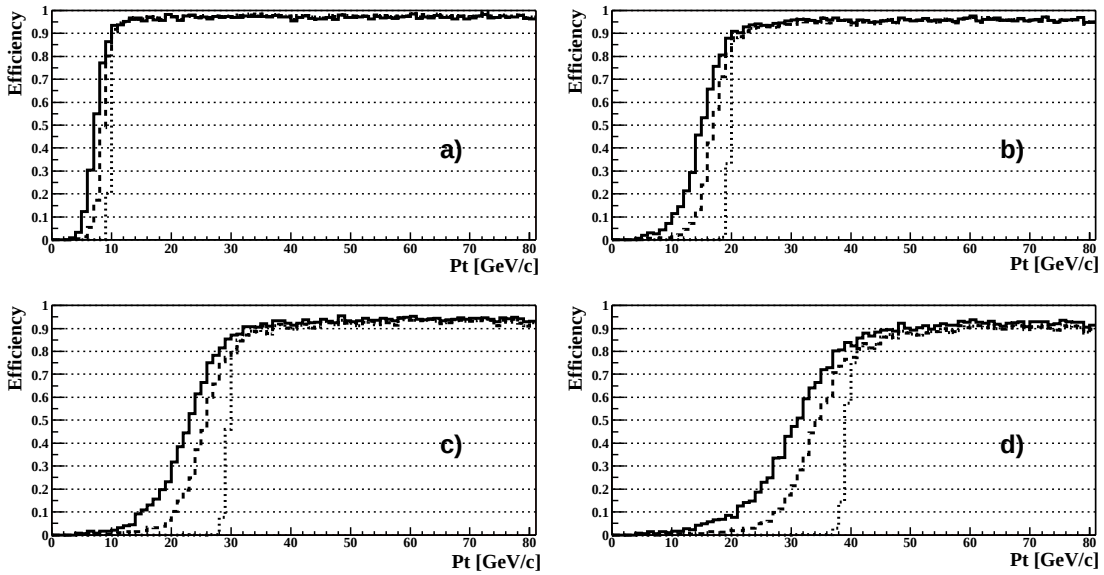


Figure 15-19 Cumulative efficiency for single muons to pass the Level-1 (solid), Level-2 (dashed), and Level-3 (dotted) triggers as a function of the generated P_T for several trigger thresholds: a) $P_T > 10$ GeV/c, b) $P_T > 20$ GeV/c, c) $P_T > 30$ GeV/c, and d) $P_T > 40$ GeV/c.

pixel isolation criteria (hereafter collectively referred to as “tracker isolation” in the figures that follow) both at the 97% efficiency point for the reference signal.

The single muon trigger rates as a function of the P_T threshold are shown in Figure 15-20 for both low luminosity ($L=2\times10^{33}$ cm^{-2}s^{-1}) and high luminosity ($L=10^{34}$ cm^{-2}s^{-1}). The rates are shown separately for Level-1, Level-2, and Level-3, with and without isolation applied at Levels 2 and 3. Also shown is the single muon rate that was generated in the simulation. At low luminosity, the Level-3 inclusive single-muon trigger rate can be reduced to 30 Hz with a P_T threshold of 18 GeV/c when the isolation criteria are applied. The rate is about 100 Hz without isolation at Level-3 for the same threshold. At high luminosity, a threshold of 38 GeV/c reduces the single-muon Level-3 rate to 30 Hz with isolation (50 Hz without isolation), and a threshold of 31 GeV/c yields a rate of 50 Hz (100 Hz without isolation). The reason why isolation achieves less rejection at higher thresholds can be seen in Fig. 15-21, which shows the contributions to the Level-3 trigger rate at high luminosity from all sources of muons before and after all isolation criteria have been applied. The isolation criteria strongly suppress the contributions from b, c, K, and π decays. This reduces the HLT rate less at high thresholds, however, where the single-muon rate is dominated by $W\rightarrow\mu\nu$ decays. After the isolation criteria, W decays account for 50% (80%) of the inclusive single-muon rate for a threshold of 18 GeV/c (31 GeV/c) for low and high luminosity, respectively.

The efficiency of the HLT single-muon trigger to select $W\rightarrow\mu\nu$ and $t\bar{t}\rightarrow\mu$+X events, where one $W\rightarrow\mu\nu$ decay is required in the latter, is shown in Figure 15-22 as a function of the P_T threshold at low luminosity. Thresholds are defined as the P_T value for which the efficiency for muons is 90% of the maximum attainable efficiency. Approximately 70% of both the W and top quark decays that occur in the fiducial region $|\eta|<2.1$ are recorded by the isolated single-muon trigger for a P_T threshold of 18 GeV/c. The efficiencies are 42% (53%) for W (top) decays for a P_T threshold of 31 GeV/c.

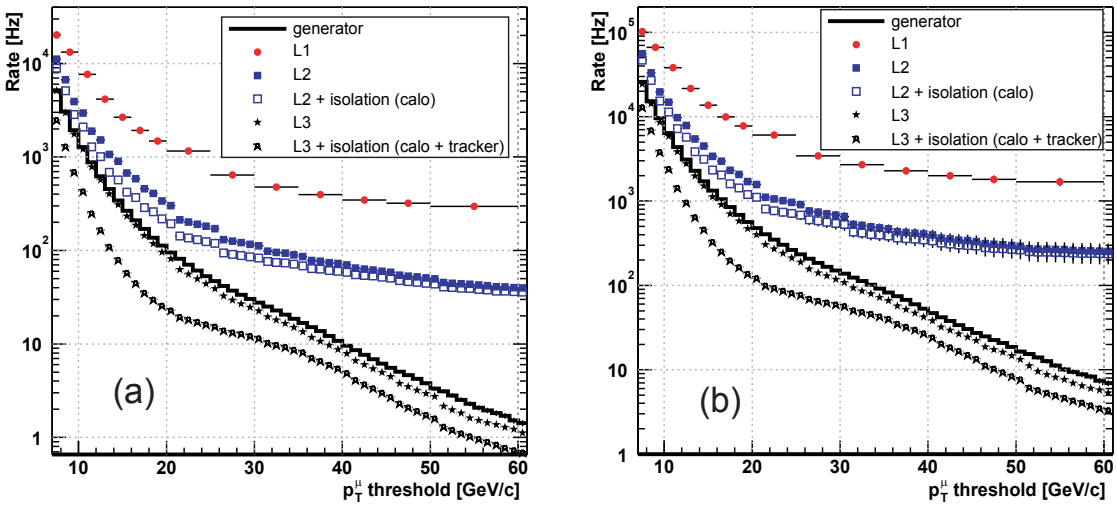


Figure 15-20 The HLT single-muon trigger rates as a function of the P_T threshold for (a) low luminosity and (b) high luminosity. The rates are shown separately for Level-1, Level-2, and Level-3, with and without isolation applied at Levels 2 and 3. The rate generated in the simulation is also shown.

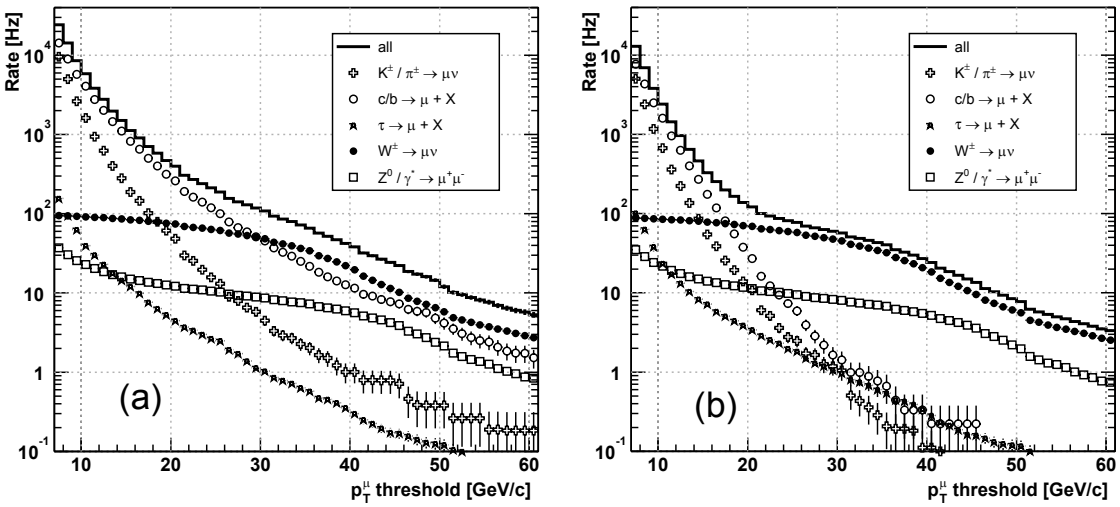


Figure 15-21 Contributions to the Level-3 trigger rate at high luminosity from all sources of muons (a) before and (b) after all isolation criteria have been applied.

15.3.3.2 Di-muon HLT Selection

The selection criteria for each muon in an inclusive di-muon trigger are the same as those for the single muon trigger, except that the isolation criteria need only be satisfied by one of the two muons. In addition, at Level-3, both muons are required to have originated from the same vertex in z to within 5 mm (to reduce triggers from muons in separate pp collisions), whereas di-muons that have $\Delta\phi<0.05$, $|\Delta\eta|<0.01$, and $\Delta P_T<0.1$ GeV/c are rejected in order to remove ghost tracks.

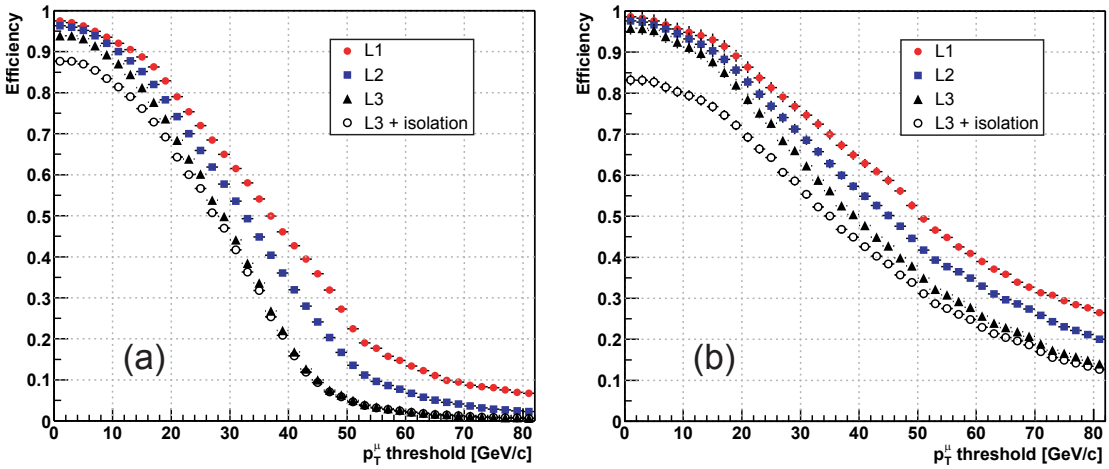


Figure 15-22 Efficiency to select (a) $W \rightarrow \mu\nu$ events and (b) $t\bar{t} \rightarrow \mu + X$ events (where one $W \rightarrow \mu\nu$ decay is required) as a function of the HLT single-muon P_T threshold. Thresholds are defined at 90% efficiency with respect to the plateau value, and efficiencies shown are for low luminosity.

The inclusive di-muon trigger rates as a function of a symmetric P_T threshold applied to both muons are shown in Fig. 15-23 for both low and high luminosity. The rates are shown separately for Level-2, Level-3, and with various combinations of the isolation criteria.

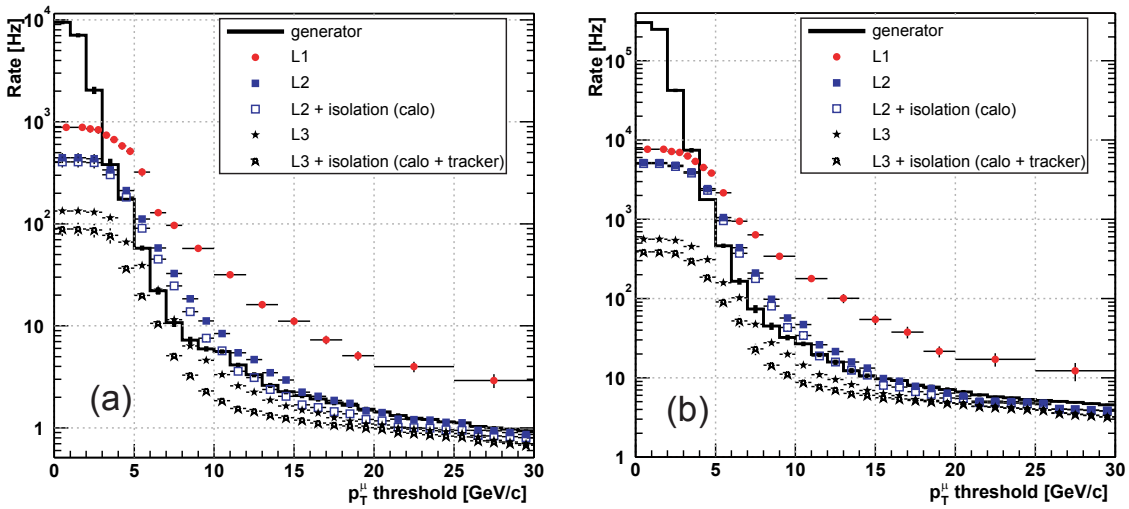


Figure 15-23 The HLT di-muon trigger rates as a function of a symmetric P_T threshold applied to both muons for (a) low luminosity and (b) high luminosity. The rates are shown separately for Level-1, Level-2, and Level-3, with and without isolation applied at Levels 2 and 3. The di-muon rate generated in the simulation is also shown.

The combined single and di-muon trigger rates are shown in Fig. 15-24 as a function of both the symmetric di-muon P_T threshold and the single muon P_T threshold for both low and high luminosity. A possible working point at low luminosity for a target rate of 30 Hz is a single muon P_T threshold of 19 GeV/c and a symmetric di-muon threshold of 7 GeV/c. Possible working points at high luminosity are a single muon P_T threshold of 38 GeV/c and a symmetric di-muon threshold of 12 GeV/c for a combined rate of 33 Hz, or a single muon P_T threshold of 31 GeV/c and a symmetric di-muon threshold of 10 GeV/c for a combined rate of about 55 Hz.

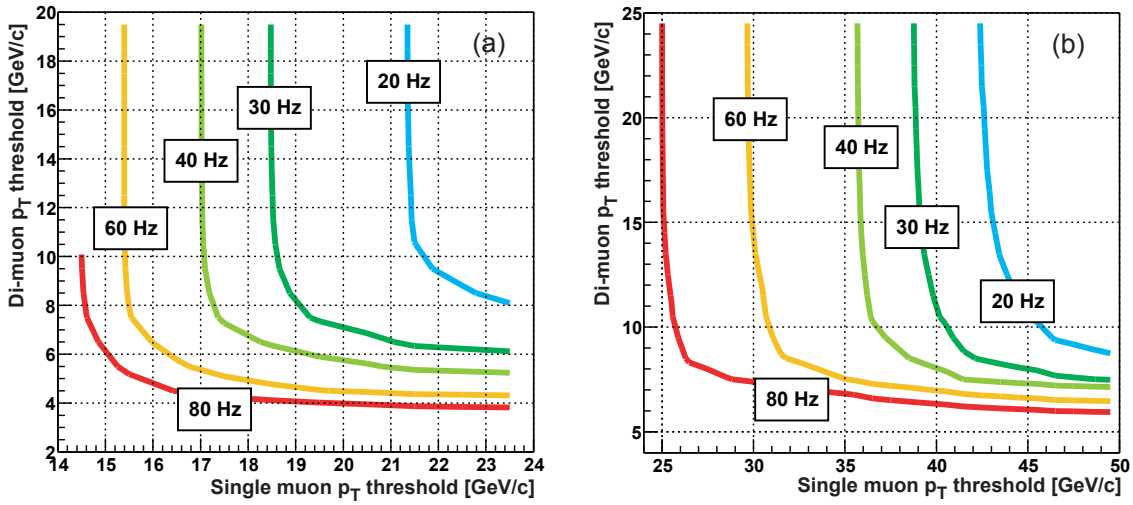


Figure 15-24 Combined single and di-muon trigger rates as a function of both the symmetric di-muon P_T threshold and the single muon P_T threshold for (a) low and (b) high luminosity.

The di-muon differential rate at Level-3 with respect to the di-muon invariant mass is shown in Figure 15-25 for the low luminosity case and for a symmetric di-muon threshold of 7 GeV/c. The separate contributions from minimum-bias collisions, Drell-Yan di-muon production, and top-quark decays are shown.

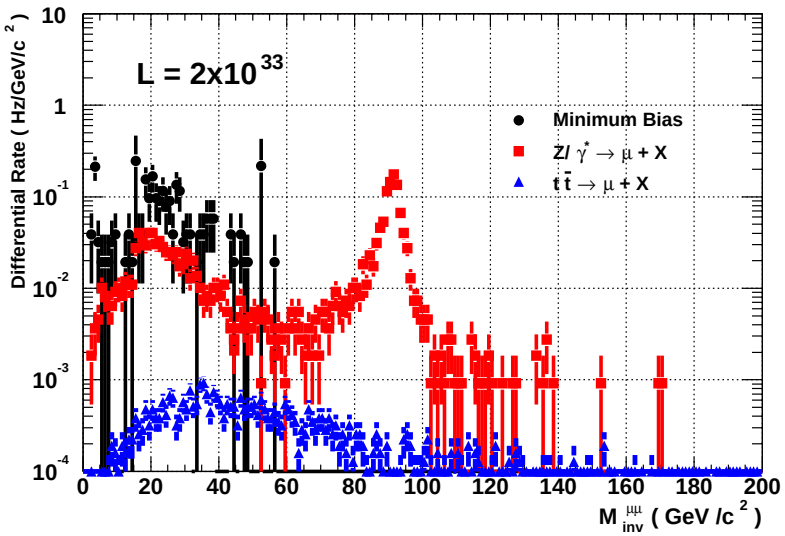


Figure 15-25 Di-muon differential rate at Level-3 with respect to the di-muon invariant mass for the low luminosity case and for a symmetric di-muon threshold of 7 GeV/c.

15.3.4 Muon HLT Performance and Timing

15.3.4.1 Efficiencies on Higgs Signals

The single muon and di-muon efficiencies of the HLT selection have been studied as a function of the Higgs mass in the channel $H \rightarrow WW^{(*)} \rightarrow \mu\mu\nu\nu$. The efficiency versus the single muon and symmetric di-muon thresholds, is shown in Figure 15-26 a,b, respectively, for Higgs masses of 120, 160 and 200 GeV/c^2. At low luminosity, for a single muon threshold of 19 GeV/c and a symmetric di-muon threshold of 7 GeV/c, the combined HLT efficiency is 92% for a Higgs mass of 160 GeV/c^2. For the

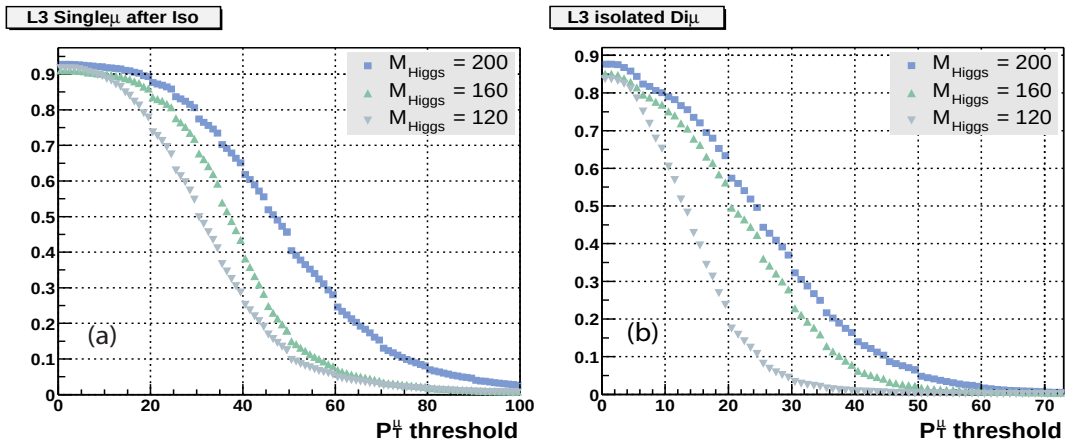


Figure 15-26 Efficiency to select $H^0 \rightarrow WW \rightarrow 2\mu\ 2\nu$ decays after Level-3 and isolation cuts are applied as a function of (a) the single muon P_T threshold and (b) the symmetric di-muon P_T threshold. Efficiencies for Higgs masses of 120, 160, and 200 GeV/c^2 are shown.

channel $H \rightarrow ZZ^{(*)} \rightarrow \mu\mu\mu\mu$., the combined efficiency of the HLT is 98% (99%) for a Higgs mass of 150 GeV/c^2 (200 GeV/c^2) for the same thresholds at low luminosity. The efficiency becomes 97% (99%) for the thresholds at high luminosity. These efficiencies are relative to those events with at least one muon inside the geometric acceptance of the trigger, $|\eta|<2.1$, and all final-state muons (two or four, depending on the channel) inside the full acceptance of the muon system: $|\eta|<2.4$.

15.3.4.2 Final Rates Written to Permanent Storage

The overall muon HLT rates and efficiencies are summarized in Table 15-9 for low luminosity and Table 15-10 for high luminosity. The Level-1 thresholds used are a single muon P_T threshold of 14 GeV/c and a symmetric di-muon threshold of 3 GeV/c at low luminosity, and a single muon P_T threshold of 20 GeV/c and a symmetric di-muon threshold of 5 GeV/c at high luminosity. The HLT operating point at low luminosity is a single muon P_T threshold of 19 GeV/c and a symmetric di-muon threshold of 7 GeV/c, and at high luminosity the operating point is a single muon P_T threshold of 31 GeV/c and a symmetric di-muon threshold of 10 GeV/c. The efficiencies for selecting *W*, *Z*, and top-quark decays to muons are also listed. The geometric acceptance is factored out and is 50% for W decays, 71% for Z decays, and 86% for top decays with at least one muon satisfying $|\eta|<2.1$.

Table 15-9 Muon rates and efficiencies for the low luminosity selection. Both absolute and relative efficiencies are shown, where the relative efficiency is with respect to the preceding level (except for Level-3, which is respect to Level-2).

	Rate (Hz)		Efficiency for $W \to \mu\nu$		Efficiency for $t\bar{t} \to \mu + X$		Efficiency for $Z \to \mu\mu$	
Level	**Single**	**Double**	**Relative**	**Absolute**	**Relative**	**Absolute**	**Relative**	**Absolute**
Level-1	2700	900		0.90		0.94		0.99
Level-2	335	25	0.89	0.80	0.93	0.88	0.99	0.98
Calo isolation	220	20	0.97	0.77	0.90	0.79	0.98	0.95
Level-3	100	10	0.93	0.74	0.95	0.84	0.99	0.97
Level-3+calo +tracker isolation	25	4	0.94	0.69	0.86	0.72	0.95	0.92
Total	29			0.69		0.72		0.92

Table 15-10 Muon rates and efficiencies for the high luminosity selection. Both absolute and relative efficiencies are shown, where the relative efficiency is with respect to the preceding level (except for Level-3, which is respect to Level-2).

	Rate (Hz)		Efficiency for $W \to \mu\nu$		Efficiency for $t\bar{t} \to \mu + X$		Efficiency for $Z \to \mu\mu$	
Level	**Single**	**Double**	**Relative**	**Absolute**	**Relative**	**Absolute**	**Relative**	**Absolute**
Level-1	6200	1700		0.82		0.90		0.97
Level-2	700	35	0.70	0.58	0.83	0.74	0.96	0.94
Calo isolation	590	25	0.97	0.56	0.91	0.68	0.97	0.91
Level-3	100	10	0.78	0.45	0.89	0.66	0.97	0.91
Level-3+calo +tracker isolation	50	5	0.94	0.42	0.88	0.58	0.94	0.86
Total	55			0.42		0.58		0.86

15.3.4.3 CPU Usage

The CPU usage of the muon HLT algorithms has been benchmarked on an Intel 1 GHz Pentium-III CPU using a sample of properly weighted minimum-bias events and *W* decays, which give the dominant contribution to the rate. The results are given in Table 15-11 for low and high luminosity. Each line represents the average time to process an event passing the previous level. For Level-2, this represents the time to process an event passing a Level-1 single muon trigger with a threshold of P_T>10 GeV/c (P_T>18 GeV/c) at low (high) luminosity. This value, which is lower than that proposed in the previous section, leads to a conservative CPU estimate because lower P_T muons take more time to reconstruct. The same P_T thresholds are applied at Level-2 and Level-3.

The distribution of CPU times measured for low luminosity events at Level-2 and Level-3 is shown in Figure 15-27a and Figure 15-27b, respectively. Fits to a log-normal distribution are overlaid. Also shown in the figures, and in the table, is the time to complete the HLT algorithm excluding the GEANE routine for propagation through iron, which is significantly less that the total time. Clearly there are substantial

Table 15-11 CPU usage of the muon HLT algorithms at low and high luminosity. The values given represent the average time to process an event passing the previous trigger level. Also listed is the time without the contribution of the GEANE propagation routine.

	Mean CPU Time (ms/event) $L=2\times10^{33}cm^{-2}s^{-1}$, $P_T>10$ GeV/c		Mean CPU Time (ms/event) $L=10^{34}cm^{-2}s^{-1}$, $P_T>18$ GeV/c	
HLT Algorithm	**Total**	**Excluding GEANE**	**Total**	**Excluding GEANE**
Level-2	640	100	580	100
Calorimeter isolation	100	25	90	40
Level-3	420	200	590	420
Pixel isolation	65	65	320	320
Tracker isolation	190	190	370	370
Total/L1 event	710	125	660	150

gains to be made by replacing GEANE with a faster method[1]. The total time listed in the last row represents the average time spent per Level-1 event by the muon HLT algorithms, factoring in the rejection power at each level that reduces the rate to the next level. This time amounts to approximately 700 ms per L1A, including the time spent in GEANE.

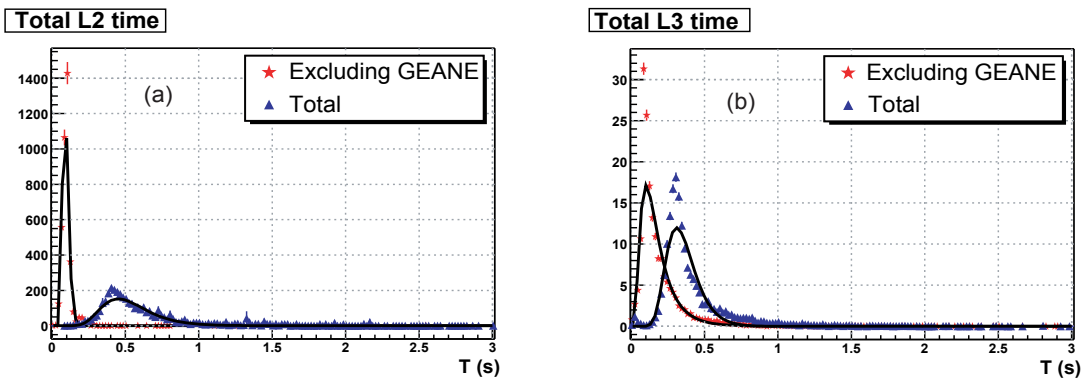


Figure 15-27 Distribution of the CPU time spent for low luminosity events processed by (a) the muon Level-2 and (b) the muon Level-3 algorithms on a 1 GHz Intel PIII processor. Also shown is time taken exclusive of the GEANE propagation routine.

1. For calorimeter isolation (Section 15.3.2.1) GEANE is used for propagation to the ECAL/HCAL boundary.

15.4 Jet and Neutrino Identification

15.4.1 Jet-finding Algorithm

Before jet-finding is executed in the HLT, the calorimeter data is organized into “towers" using the $\eta-\phi$ segmentation of the HCAL (0.087×0.087 in the barrel, near the edge of the endcap the η segmentation is as large as 0.175). There are 4176 such towers. In the ECAL barrel region, i.e. $|\eta|<1.479$, there are 25 ECAL crystals per HCAL tower whereas in the endcap region, the number of crystals per tower varies with η. The towers are treated as massless particles with energy equal to the scalar sum of the energies of the components of the tower, and using the center of the tower and z=0 for the nominal interaction point to define the angles.

15.4.1.1 Basic Algorithm for Jet-finding

Global jet-finding is done using a simple iterative cone algorithm. In this algorithm, a list of towers is made, and a “protojet” is formed using the direction of the tower from the list with the highest E_T (the “seed tower”) as the protojet direction. The direction of the protojet is calculated from the transverse-energy-weighted angles of the towers in a cone around the protojet direction in η-ϕ space, and the transverse energy of the protojet is calculated using the direction of the protojet and the sum the energies of the towers in the cone. The direction of the protojet is used to seed a new protojet. The procedure is repeated until the energy of the protojet changes by less than 1% between iterations and the direction of the protojet changes in $\Delta\eta^2+\Delta\phi^2$ by less than 0.01, or until 100 iterations is reached. When a stable protojet is found, the towers in the stable jet are removed from the list of objects, and the procedure is repeated until no objects are left in the list or until the tower with the highest E_T is below a “seed” threshold, which is a parameter of the algorithm. This simple iterative algorithm has no sophisticated splitting and merging presciption, and is therefore fast. Offline, a more complete algorithm, such as those suggested by the ‘‘Run II Jet Algorthms Working Group” [15-11], will eventually be implemented. The offline algorithm will also use more sophisticated algorithms that use more information than just the tower energies and give improved energy resolution. There are currently promising results on energy flow algorithms and new types of algorithms for the very high luminosity conditions at the LHC [15-12]. For the HLT, however, a simple algorithm is clearly preferable.

Jet-finding is currently carried out using all calorimeter towers, instead of just those around Level-1 jet candidates. However, different HLT triggers can require that the HLT jet candidates match to Level-1 candidates to simplify trigger efficiency calculations.

15.4.1.2 Parameter Choice for Jet-finding Algorithm

The simple cone algorithm has three parameters, namely the size of the cone, the seed threshold, and the minimum jet E_T. These parameters were optimized for high luminosity conditions. The resulting values are 0.5 (cone size), 2 GeV (seed threshold) and 10 GeV (jet threshold). Studies of different parameter settings for low luminosity are ongoing.

At the parton level, a jet has a reasonably narrow core in $\eta-\phi$ space that contains most of its energy. However, because of soft gluon emission, the cone size that gives the best energy resolution for the parton can be quite large. When the jet is reconstructed in the calorimeter, however, a large cone will have a larger noise contribution, both from electronic noise and from energy from particles from pileup and from underlying event. Figure 15-28 displays the resolution at high luminosity for particle-level (“generator”) jets

formed using a cone size of 0.7 when reconstructed in the calorimeter using a cone size of 0.5 and 0.7, for the two regions $|\eta|<1$ and $3.5<|\eta|<4.5$. The cone 0.5 has signficantly better resolution at high rapidity and low E_T. It has similar resolution everywhere else.

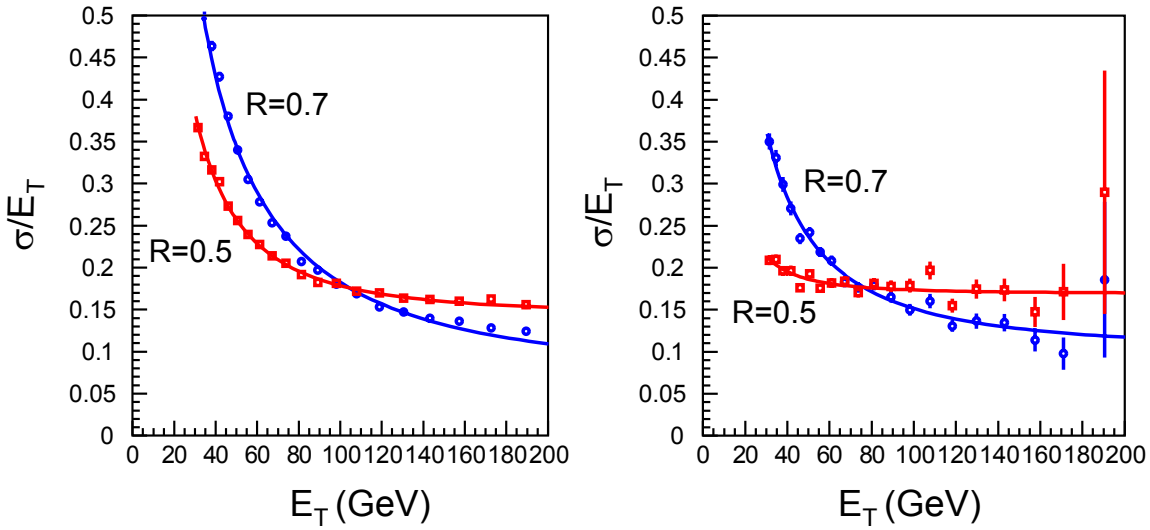


Figure 15-28 Jet E_T resolution as a function of generator jet E_T, for two cone sizes (0.5 and 0.7) for jets with $|\eta|<1$ (left) and $3.5<|\eta|<4.5$ (right). The resolution is defined as the R.M.S. of the difference between the E_T of a generator-level jet, found in a 0.7 cone, and the offline jet divided by the E_T of the generator-level jet.

Figure 15-29 shows, for a high luminosity $t\bar{t}$ Monte Carlo sample, the invariant mass of the jets that best match the generator level quarks from *W* decay for both cone 0.5 and cone 0.7 jets. The cone 0.5 jets give a slightly better resolution.

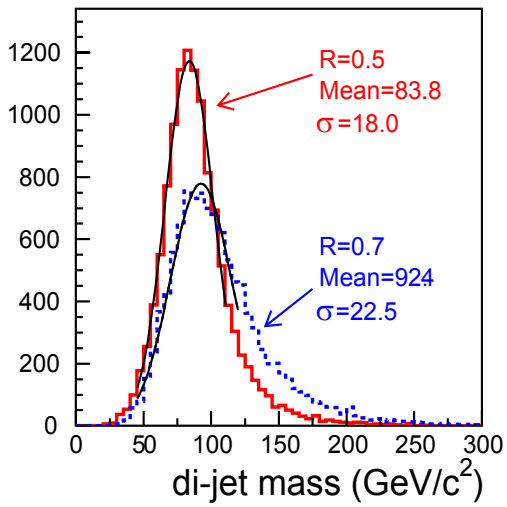


Figure 15-29 The reconstructed di-jet mass, at high luminosity, for the jets that most closely match the *W* decay partons in $t\bar{t}$ events for two different cone sizes. The narrower distribution is for a cone size of 0.5 and the broader for a cone size of 0.7.

The seed thresholds cause an inefficiency for low E_T jets that depends on the details of jet fragmentation and is therefore difficult to model. While modern offline jet finding algorithms may try to be seedless (see

Reference [15-11]), at the trigger level a seed helps reduce the number of fake jets, which can otherwise be created by noise and pileup and which typically have E_T<50 GeV, and speeds up the time for jet finding. Figure 15-30 shows the rapidity distribution of reconstructed jets in a sample of Z′ events, with M(Z′)=120 GeV/c^2, for two different seed thresholds. The left-hand plot, which shows the distribution for a 1 GeV seed, shows a significant excess of jets with low transverse energy with central rapidity. These fake jets are greatly reduced when a higher seed (3 GeV) is used, as shown in the right hand side of Figure 15-30. Figure 15-31 shows the efficiency to reconstruct jets as a function of the E_T of the jet for different values of the seed threshold. Since none of the current triggers have jet thresholds below 30 GeV, a 2 GeV seed is used.

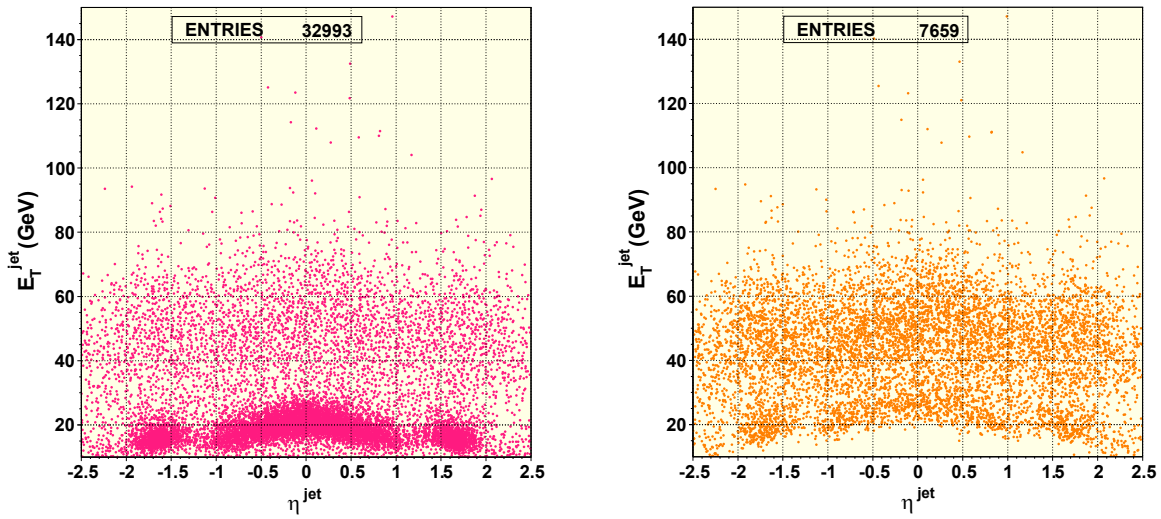


Figure 15-30 The distribution of η versus E_T for jets in Monte Carlo simulation of the decay of a *Z'* with mass 120 GeV/c^2. Left: a seed of 1 GeV is used; right: a seed of 3 GeV is used.

A low minimum jet E_T is needed for calibrating the jets for the Monte Carlo in order to avoid biases that come from the effects of noise and pileup, as described in the following section. The cut used in this analysis is 10 GeV.

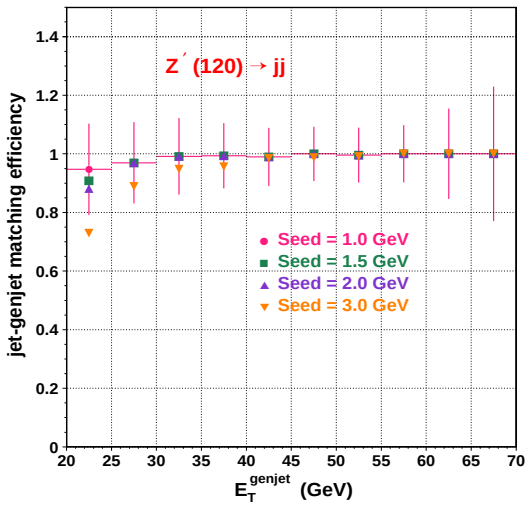


Figure 15-31 Efficiency to find an offline jet that matches a generator-level jet, as a function of the generator-level jet E_T at high luminosity.

15.4.1.3 Jet Energy Scale Corrections

The CMS calorimeter system is optimized for the precision measurement of electrons and photons, and has a non-linear response to pions. Since the energy of a typical pion in a jet is roughly proportional to $1/\sin(\theta)$, the response of the calorimeter to jets of a given transverse energy varies with η. In addition, because only those channels that are above a threshold are read out ("one-sided zero-supression"), instead of those both above one threshold and below another ("two-sided zero-supression"), and which is set at a value equal to $\approx 2\sigma$ of the noise pedestal in the HCAL and ECAL, a random cone on the calorimeter will have a positive energy due to noise alone. Table 15-12 shows the size of the offset due to noise versus the setting for the zero-supression threshold (in number of noise σ). A two-sided zero-supression threshold is currently under investigation. At high luminosity, there will be an additional offset due to energy from particles from pileup interactions.

Table 15-12 Amount of energy in a cone of 0.5 due to noise alone for various values of a zero-supression threshold.

threshold (number of noise σ)	energy in cone (GeV)
none	0.0
0	45.2
1	27.3
2	7.6
3	0.5

Corrections for these effects are introduced both at Level-1 and in the HLT. At the start of the experiment, the corrections will be derived from Monte Carlo data by comparing the energies of jets found using the generator-level particles of the hard interaction (excluding particles from "pileup" interactions) to jets found in the calorimeter. As soon as sufficient data is collected, new corrections will be derived from single photon plus jets data and Z boson plus jet data.

For the purposes of this TDR, the Monte Carlo jets are calibrated so that the mean response for a generator jet is equal to the generator jet energy. This technique is expected to yield results that are virtually identical to those that would be obtained from a full calibration using events with photon and a jet. Figure 15-32 shows the size of the typical correction for HLT jets, obtained from simulation, for low luminosity and high luminosity running conditions. The correction can be as large as 20% for low-E_T (40 GeV) jets.

The current correction is obtained by fitting the mean reconstructed E_T, $\langle E_T^{rec} \rangle$, as a function of the generator-level E_T, E_T^{gen}, to a quadratic function: $\langle E_T^{rec} \rangle = A(E_T^{gen})^2 + BE_T^{gen} + C$ where A, B and C are functions of η. The resolution of the E_T measurement for jets before and after corrections, at both low and high luminosity, is shown in Figure 15-33 for cone 0.5 jets. The main effect of the corrections is the removal of the η dependence of the trigger efficiency, while the resolution is improved only slightly.

The resolution for high E_T jets (E_T>300 GeV) is similar at low and high luminosity, while the resolution at lower E_T values is significantly worse at high luminosity. As an example, the resolution for jets with E_T=40 GeV is approximately 19% at low luminosity and 24% at high luminosity. Work is ongoing in improving the jet energy resolution at high luminosity.

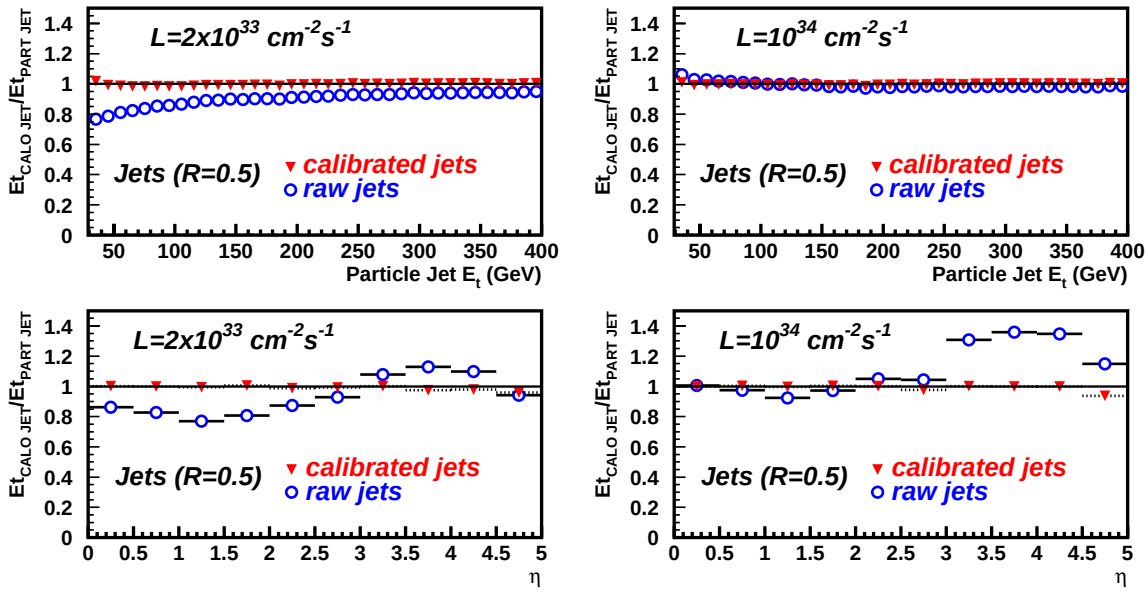


Figure 15-32 Ratio of reconstructed to generated jet E_T for HLT jets versus η and versus E_T, before and after jet energy scale corrections. Left: low luminosity; right: high luminosity.

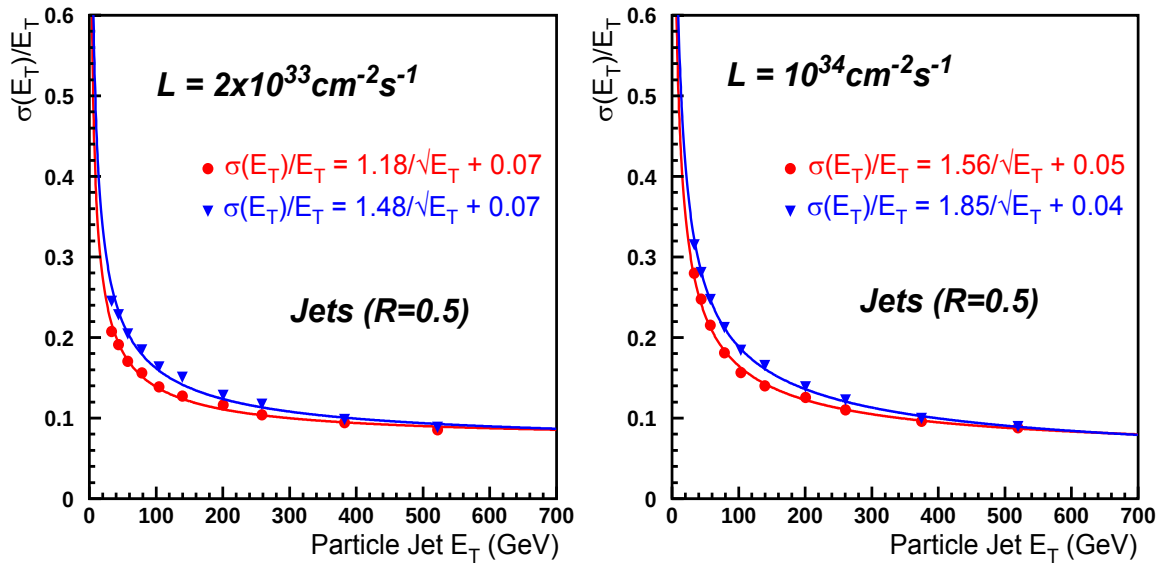


Figure 15-33 Resolution for HLT jets before (triangles) and after (circles) jet energy scale calibration. Left: low luminosity; right: high luminosity.

15.4.1.4 Fake Jet Supression

At high luminosity, with a high number of interactions per crossing, low E_T jets from the hard scattering can be mismeasured as high E_T jets. This happens mainly when either particles or a small jet from one pileup interaction impact the calorimeter near one of the low E_T jets creating a jet with higher E_T, a “fake jet”. Fake jets can also be created when particles from different interactions impact near each other in the calorimeter. In addition, the magnetic field sweeps charged particles with transverse momentum below 0.77/c GeV out of the barrel and into the endcap region of the calorimeter. This sweeping effect prevents

more than half of the charged particles pointed toward the barrel from actually striking the calorimeter, severely altering the energy flow in the underlying and pileup events. The accidental overlap of these particles with particles from the soft interactions causes the fake jet problem to be especially severe at the HE/HF boundary. Figure 15-34 shows the rapidity distribution of all calorimeter jets (solid) and the rapidity distribution of the subset of these that have a good match in $\eta-\phi$ to a jet made from generator-level particles from the hard scattering for jets with transverse energies above 20, 30, and 40 GeV. Above 40 GeV, the number of fake jets is small.

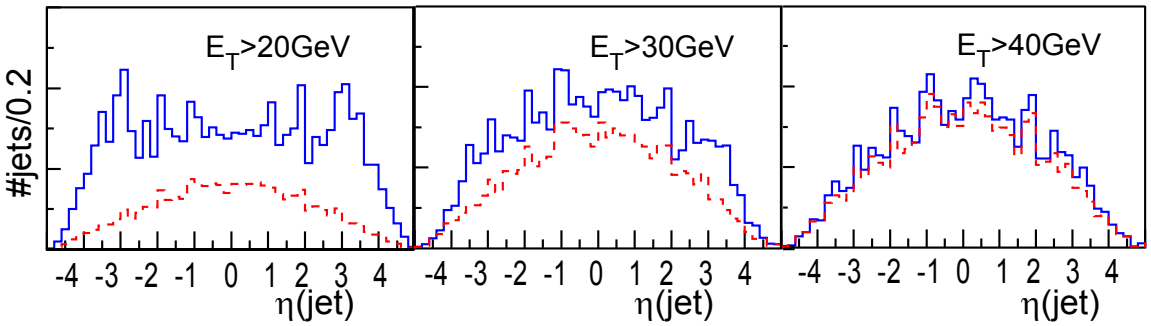


Figure 15-34 η distribution of reconstructed jets for three thresholds on the jet E_T. Solid line: all reconstructed jets. Dashed line: only jets with a good match to a generator-level jet from the hard scattering.

Methods of supressing these fake jets at the trigger level have been studied using only calorimeter information for the HLT. Ongoing work is investigating the use of tracking (pixel) information for this purpose.

There is no unambiguous way to classify a jet as “fake” or “real”. Most jets found by the jet finder contain some particles from the hard scatter and some particles from the pile up events. The degree of “fakeness” that is acceptable will vary according to the particular trigger and physics channel. To measure the “realness” of a jet, the generator-level particles are extrapolated into the jet cone and the variable:

$$R1 = max\left(\frac{E_T(vtx_i)}{E_T(jet))}\right)$$

is defined. $E_T(vtx_i)$ is the scalar sum of the E_T's of the particles extrapolating into the jet cone from the i^{th} vertex and $E_T(jet)$ is the E_T of the jet. If a jet has more than a 50% of its E_T contributed from a single vertex, i.e. R1>0.5, it is considered to be a real jet, otherwise it is considered to be a fake jet. Figure 15-35 shows R1 versus E_T and η for jets from an inclusive jet sample with $30<P_T^{hard}<50$ GeV, where P_T^{hard} is the transverse momentum transferred in the hard scattering subprocess at the parton level. Fake jets, which have low values for R1, tend to have low E_T. In η, fake jets tend to be near the HE/HF boundary.

Fake jets should have a broader transverse distribution than real jets. To study this, two variables, S1 and S2 are introduced. S1 is the E_T-weighted mean radius: $S1 = (\sum E_T r)/(\sum E_T)$, whereas S2 is the fraction of the E_T in a cone of 0.25, $S2 = (E_T(0.25))/(E_T(0.5))$. Only results from S2 are shown here, since both variables give similar results. Figure 15-36 shows the correlation between R1 and S2, showing that “fake” jets tend to have a large S2. An interesting variable is the fraction of jets with R1<0.5 that are rejected versus those with R>0.5 that are kept, as a function of a cut on S2. Figure 15-37 shows the result, showing it is possible to keep real jets with a reasonable efficiency while supressing fake jets. For example, for an 80% efficiency for selecting real jets, about half of the fake jets are rejected. However, this cut is not used in the subsequent rate plots because it does not seem necessary in order to achieve the desired rate to tape, and it is clearly preferable not to bias the jet samples with these cuts if possible. These parameters should be useful in the development of more sophisticated triggers in the future.

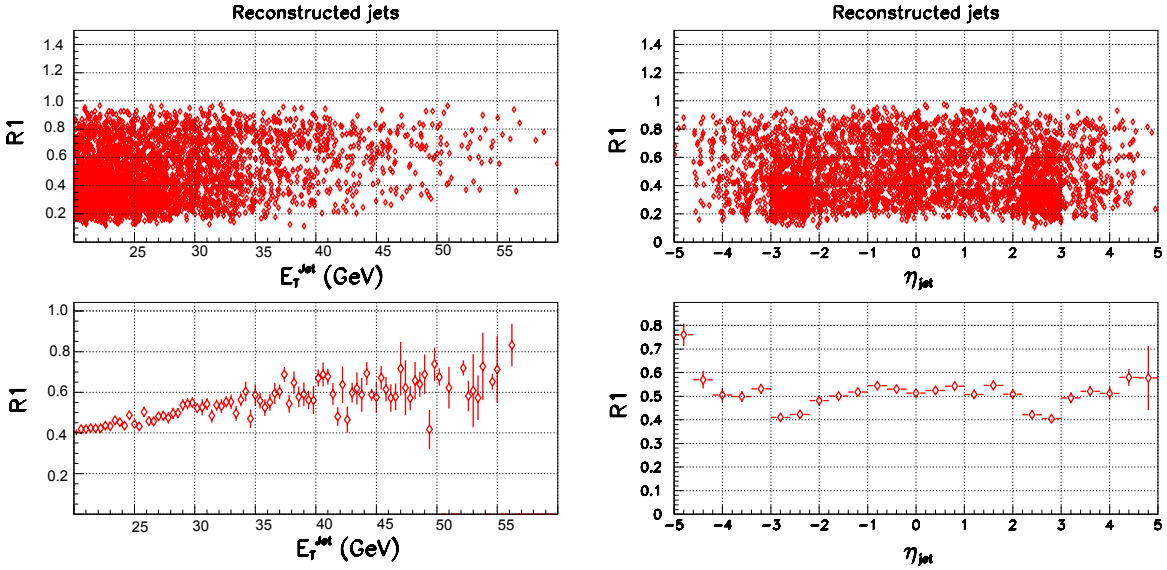


Figure 15-35 R1, a measure of the realness of a jet, versus the E_T and η of the jet. Top: scatter plot of R1 vs E_T (left) and η (right). Bottom: mean R1 vs E_T (left) and η (right). R1 is small when the jet has sizable contributions from more than one vertex.

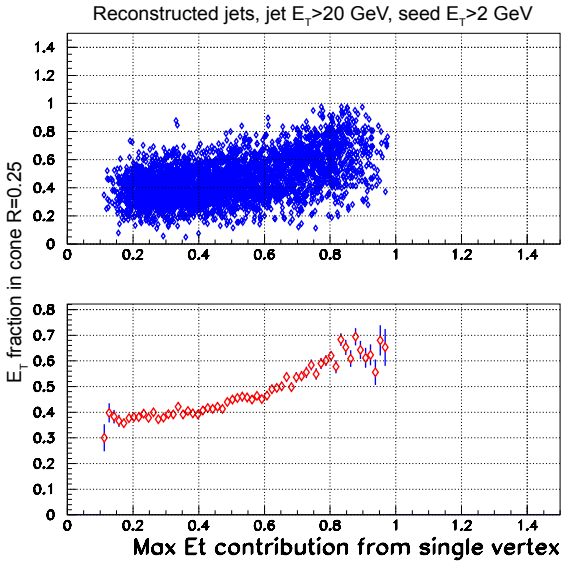


Figure 15-36 Fraction of jet E_T in a cone of radius 0.25 versus R1, the measure of fakeness of a jet. A small value of R1 means the jet has non-negligible contributions from more than one primary vertex.

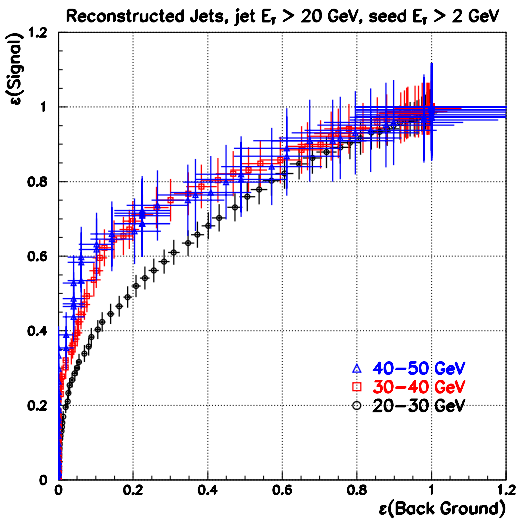


Figure 15-37 Efficiency versus rejection for a cut on the fraction of the jet E_T in a cone of 0.25 for removing fake jets.

15.4.1.5 Jet Rates

The jet rates are insensitive to the details of the simulation and are robust against reasonable changes in jet algorithms. For example, Figure 15-38 shows the rates at low and high luminosity for a 1 jet trigger for jets reconstructred at the generator level using particles from only the hard scattering (labeled in the Figure as Signal GenJet) and using generator-level particles from all interactions (labeled in the Figure as in-time GenJet). The Figure also shows the HLT jet rates both before and after jet energy scale correc-

tions. The rates for all these different types of jets are basically the same. At fixed E_T, the high luminosity rates are about five times larger than the rates at low luminosity, as expected.

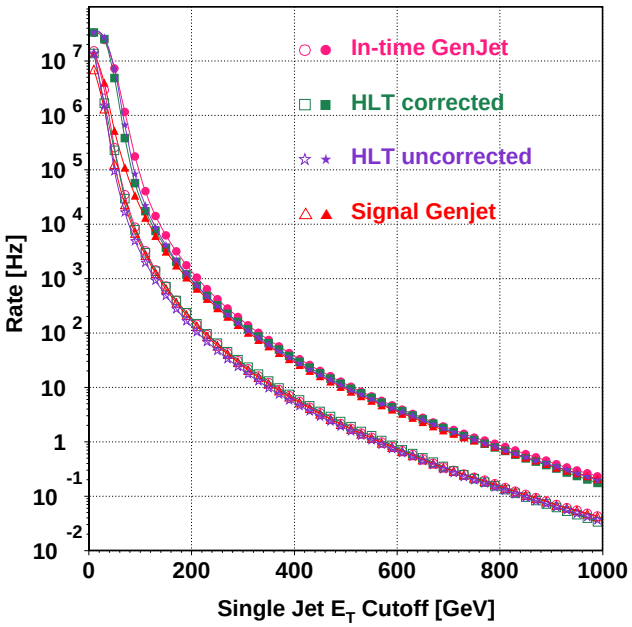


Figure 15-38 Rates for a single- jet trigger at the generator-level using only particles from the hard scattering to make the jets and also using all particles, including those from the pileup interactions, at both low and high luminosity. Also shown are the HLT single-jet trigger rates both before and after jet energy scale corrections at low and high luminosity. The open (filled) symbols show the low (high) luminosity rate.

Figure 15-39 shows the rate for single-jet, 3-jet, and 4-jet triggers at low and high luminosity. The x-axis is the threshold on the calibrated jet E_T for the HLT rate curve. Figure 15-40 shows the single-jet rate and the incremental rate from 2-jet and 3-jet triggers rates.

The effect of different algorithms and different detectors can be studied by comparing the rate as a function of the generator-level jet E_T. The latter is approximated by the cut on the offline jet E_T that gives 95% efficiency for the generator-level jet E_T. The relationship between the value of the cut and the generator-level jet E_T is linear, and is given by

- $E_T(gen) = 1.13E_T(cut) + 11.1GeV$ for HLT jets at low luminosity;
- $E_T(gen) = 1.12E_T(cut) + 17.4GeV$ for HLT jets at high luminosity;
- $E_T(gen) = 1.18E_T(cut) + 17.2GeV$ for Level-1 jets at low luminosity;
- $E_T(gen) = 1.17E_T(cut) + 19.8GeV$ for Level-1 jets at high luminosity;

Figure 15-39 and Table 15-13 show the results.

The thresholds for triggers that use only jets are very high. Most HLT triggers on physics channels with jets will need something besides jets in the trigger to have an acceptable rate and an acceptably low threshold on jet E_T. As shown in the sections on SUSY (15.8.3.2) and the invisible Higgs (15.8.3.3), it is usually not difficult to find something else in the event to use as part of the trigger.

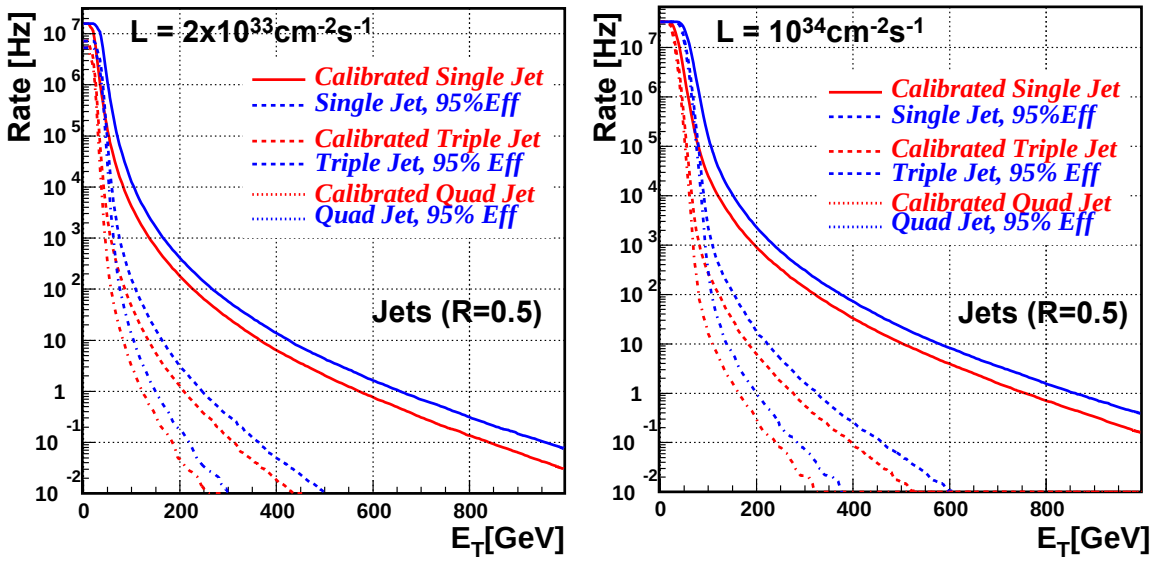


Figure 15-39 Rates for 1, 3, and 4-jet triggers as a function of calibrated jet E_T (x axis). Also shown is the rate for the cut that gives 95% efficiency for the generator-E_T shown on the x-axis are given. Left: low luminosity. Right: high luminosity.

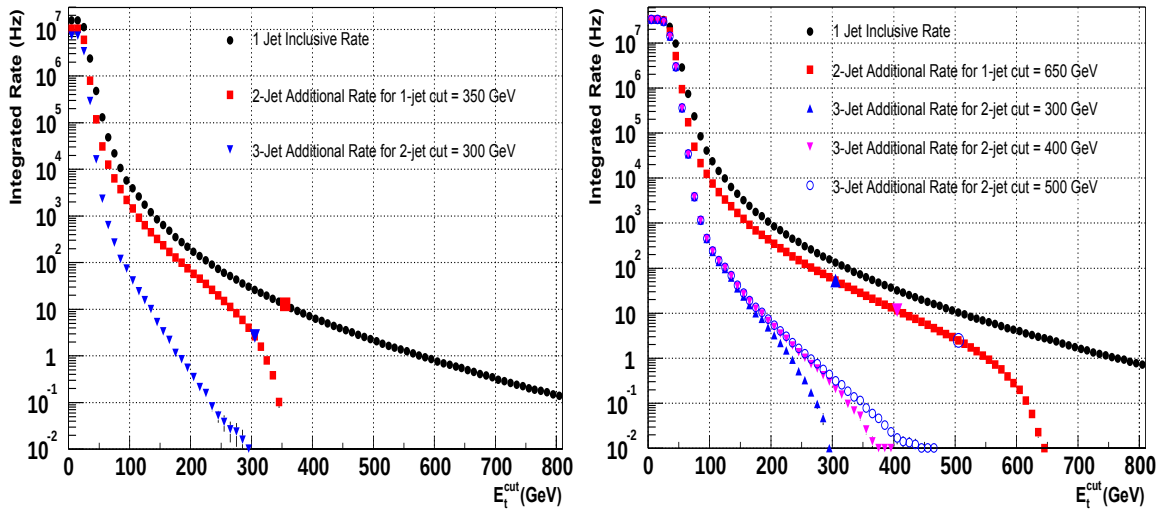


Figure 15-40 The rates for 1-jet events, and the incremental 2-jet and 3-jet rates, given a 1-jet threshold of 350 GeV, at low luminosity (left), and a 1-jet threshold of 650 GeV at high luminosity (right).

Table 15-13 Jet rate summary table. The table gives the generator-level jet E_T where the cut (in GeV) on the reconstructed jet E_T gives 95% efficiency for this generator-level jet E_T and also gives by itself a rate of 1 kHz (Level-1) and 1 Hz (HLT). The actual value of the cut on E_T that corresponds to the 95% efficiency points is given in parentheses.

	1-jet trigger	2-jet trigger	3-jet trigger	4-jet trigger
low luminosity Level-1	177 (135)	140 (104)	85 (57)	70 (45)
high luminosity Level-1	248 (195)	199 (153)	112 (79)	95 (64)
low luminosity HLT	657 (571)	564 (489)	247 (209)	149 (122)
high luminosity HLT	860 (752)	748 (652)	326 (275)	199 (162)

15.4.2 Neutrino Identification

15.4.2.1 Basic Algorithm

To identify neutrinos in the HLT, the calorimeter information is used to look for missing transverse energy (E_T^{miss}). The current algorithm calculates E_T^{miss} as a simple vector sum of the towers over a threshold of E_T(tower)> 500 MeV. The angles for the towers are calculated with respect to *z=0*. Other algorithms that include jet energy scale corrections have also been studied. "Type-I" E_T^{miss} corrections use jet energy scale corrections for jets with E_T>30 GeV and no corrections for unclustered towers. "Type-II" E_T^{miss} corrections use jet energy scale corrections for jets with E_T>30 GeV and the corrections for 30 GeV jets for unclustered towers. The uncorrected E_T^{miss} algorithm is found to have as good performance as these two algorithms, as will be shown in Section 15.4.2.4.

15.4.2.2 E_T^{miss} Scale and Resolution

Figure 15-41 shows the mean of the difference between the generator-level and reconstructed E_T^{miss} as a function of the generated E_T^{miss} for the different E_T^{miss} algorithms for a SUSY sample with squark and gluino masses ~500 GeV/c^2 decaying to jets plus E_T^{miss}. Figure 15-42 shows the RMS of this difference. The jet energy scale corrections help the energy scale for the E_T^{miss}, but do not improve the resolution.

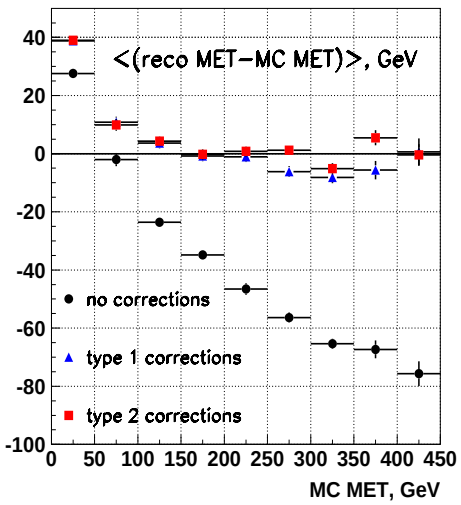


Figure 15-41 Mean difference between the generator-level E_T^{miss} and the reconstructed E_T^{miss} for three different algorithms for calculating E_T^{miss} as a function of generator-level E_T^{miss}.

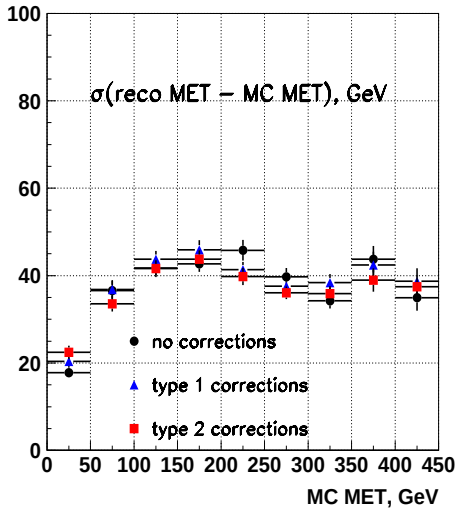


Figure 15-42 R.M.S of the difference between the generator-level E_T^{miss} and the reconstructed E_T^{miss} for three different algorithms for calculating E_T^{miss} as a function of generator-level E_T^{miss}.

15.4.2.3 Dijet Supression in E_T^{miss} Triggers

A large source of backgrounds for inclusive E_T^{miss} triggers is inclusive dijet production. Especially, if one jet goes into the forward calorimeters, while the other is central, the central jet will have lower energy for the same E_T, and thus a lower π^0 content, leading to a lower response, and also a worse resolution. There-

fore, the E_T^{miss} will tend to be along the direction of the central jet. Several ways to help remove this kind of background have been considered: a cut on the opening angle in ϕ, $\Delta\phi$, between the E_T^{miss} and the leading jet, between the E_T^{miss} and the jet with the second highest E_T, on the opening angle between the leading and second jet, the opening angle between the leading jet and the sum of the second and third jet, and the opening angle between the E_T^{miss} and the sum of the second and third jet. Figures 15-43 and 15-44 show the distributions for these variables for inclusive dijet events with $80<P_T^{hard}<120$ GeV. The variables that correlate the direction of the E_T^{miss} with the jet directions turn out to be inefficient for SUSY type events (see Section 15.8.3.2, "Supersymmetry Searches"). The best variable seems to be the opening angle between the leading and second jet. However, this cut is only effective for very high values of E_T^{miss}. Current plans for the HLT selection do not include a cut of this type.

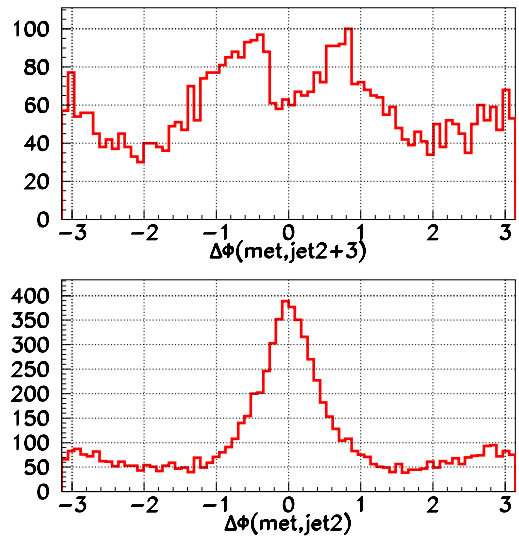


Figure 15-43 Upper plot: Opening angle in ϕ between the E_T^{miss} and the direction of the sum of the second and third jets with the highest E_T. Lower plot: the opening angle in ϕ, between the E_T^{miss} and the second highest E_T jet. The plot contains inclusive dijets with $80 < P_T^{hard} < 120$ GeV/c and $E_T^{miss} > 50$ GeV.

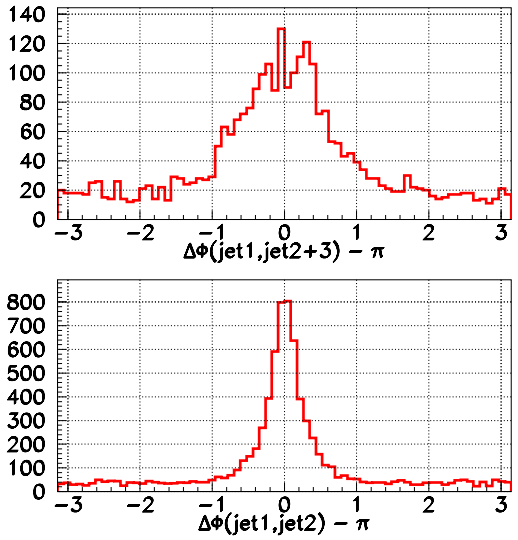


Figure 15-44 Upper plot: Opening angle in ϕ between the direction of the highest E_T jet and the direction of the sum of the second and third highest E_T jets. Lower plot: and the opening angle in ϕ, between the highest E_T jet and the second highest E_T jet minus π. The plot contains inclusive dijets with $80 < P_T^{hard} < 120$ GeV/c and $E_T^{miss} > 50$ GeV.

15.4.2.4 E_T^{miss} Rates

Figure 15-45 shows the E_T^{miss} rates at the generator level at both low and high luminosity. At the generator-level the E_T^{miss} is due to neutrinos produced in the decays of heavy flavors and due to particles which have $|\eta|>5.0$ and are therefore outside of the coverage of the calorimeter. In general, most of the rate below 60 GeV is due to the finite coverage of the calorimeter, while above 60 GeV the rate is dominated by heavy flavor production. The effect of the finite coverage of the calorimeter is more pronounced at high luminosity. Figure 15-45 also shows the Level-1 and HLT E_T^{miss} rates at low and high luminosity. There are currently no plans for triggers demanding only E_T^{miss} and no other physics object, e.g. a jet, at the HLT. Triggers using jets and E_T^{miss} are discussed in the section on physics selection. The above rates are shown for illustrative purposes only. The HLT rates are much larger than the generator-level rates, and are

very sensitive to the details of the simulation. The non-smoothness of the rates is due to insufficient statistics in the lower bins in P_T^{hard}.

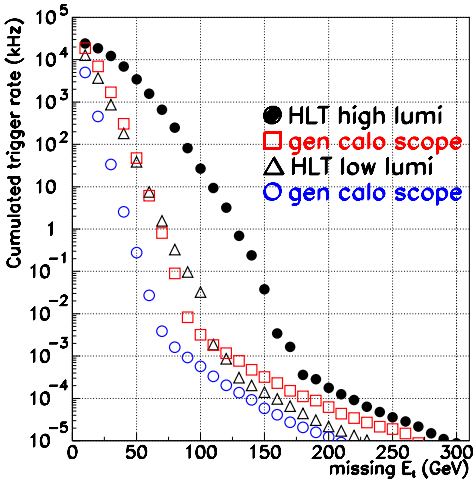


Figure 15-45 Generator-level E_T^{miss} rates at low and high luminosity. Also shown are the HLT E_T^{miss} rates at low and high luminosity.

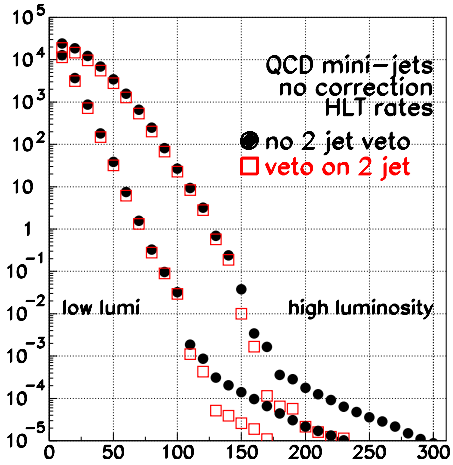


Figure 15-46 HLT E_T^{miss} rates at low and high luminosity before and after requiring the two leading jets to not be back-to-back in ϕ within 0.5 radians.

Figure 15-46 shows the effect of requiring the opening angle in ϕ between the two leading jets be larger than 0.5. This cut is only effective for large values of E_T^{miss}. Figure 15-47 shows the rates when a jet with E_T above a threshold is also required as part of the trigger.

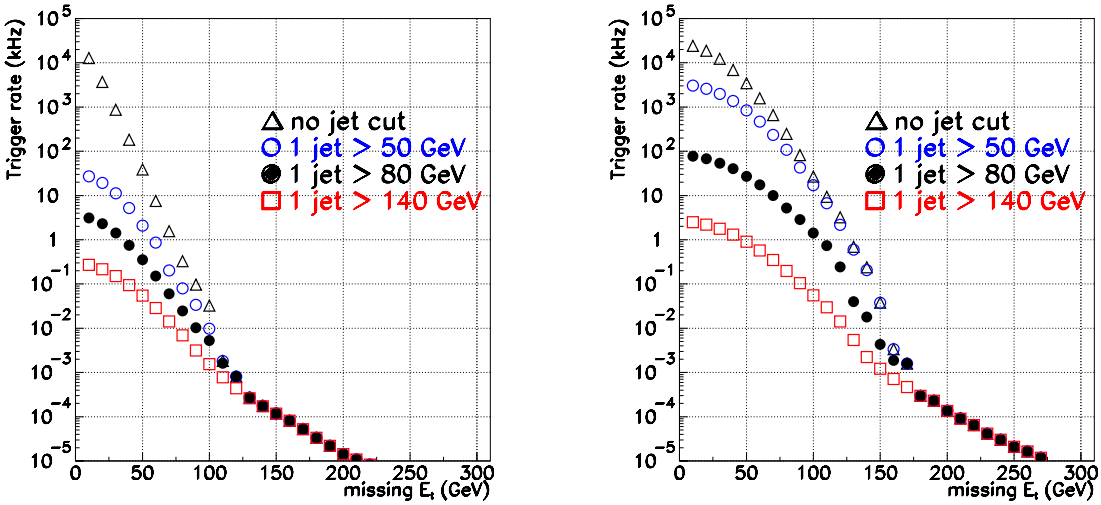


Figure 15-47 Event rates as function of E_T^{miss} when requiring a jet above various thresholds. Left: low luminosity; right: high luminosity.

To faciliate comparisons between different algorithms and different experiments, it is convenient to plot the rates as a function of the cut that gives 95% efficiency for a given generator-level E_T^{miss}. Since inclusive dijet events have very little generator-level E_T^{miss}, we use a signal sample to make a mapping be-

tween generator-level E_T^{miss} and HLT reconstructed E_T^{miss}. For a given value of generator-level E_T^{miss}, the E_T^{miss} reconstructed in the HLT is plotted. The value for offline E_T^{miss} above which 95% of the events lie is then found and mapped to the generator E_T^{miss}. The rate plots will therefore have some dependence on the signal sample that was used to make this mapping. Above generator-level E_T^{miss} of about 50 GeV, the relationship between the cut value and the generator-level E_T^{miss} is approximately linear, and is given by

- $E_T(cut) = 0.76E_T(gen) - 1.4GeV$ at low luminosity;
- $E_T(cut) = 0.72E_T(gen) - 1.8GeV$ at high luminosity.

Figure 15-48 shows these rates at both low and high luminosity for Level-1 and HLT E_T^{miss}, where a *qqH→ll* sample (Higgs produced via vector-boson fusion, with the Higgs decaying to *WW* and then to two leptons) with a Higgs mass of 180 GeV/c^2 was used to define the mapping between generator-level E_T^{miss} and reconstructed E_T^{miss}. These plots show that the E_T^{miss} algorithms with jet energy scale corrections do not give a better performance than the simple E_T^{miss} algorithm.

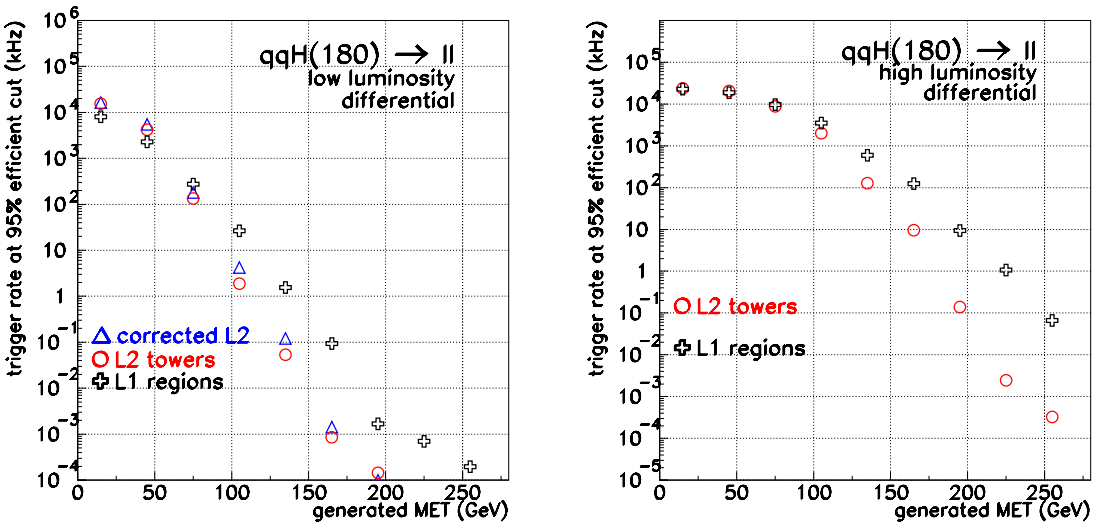


Figure 15-48 Rate vs. the cut that gives 95% efficiency for a given generated E_T^{miss} at low (left) and high (right) luminosity. The signal sample used to define the mapping between generated E_T^{miss} and a cut on offline E_T^{miss} was production of Higgs via vector boson fusion, with the Higgs decaying to *WW* and then to two leptons. The crosses give the Level-1 rate, the circles the HLT rate, and the triangles the rate for an alternative algorithm for the HLT E_T^{miss}.

15.4.3 Timing Studies for Jet/E_T^{miss} Reconstruction

Timing studies of the jet and missing transverse energy reconstruction algorithms have been carried out. The results are shown in Table 15-14. The CPU time needed to reconstruct the jets and the E_T^{miss} are about the same for signal and background samples.

At low luminosity, tower-building takes about two-thirds of the reconstruction time, and jet and E_T^{miss} reconstruction about 1/3. At high luminosity, tower-building and jet-finding each take about 1/2 the time, and the time required for the reconstruction of E_T^{miss} is small.

Table 15-14 CPU requirements for Jet and E_T^{miss} reconstruction in the HLT. Times are in ms for a Pentium-III, 1 GHz CPU.

	$A^0/H^0 \rightarrow \tau\tau$ events, M_H=500 GeV/c^2		inclusive dijets with 50<P_T^{hard}<80 GeV	
	Low lum	**High Lum**	**Low Lum**	**High Lum**
tower-building	26	38	24	36
Jet reconstruction	9	34	9	45
E_T^{miss} calculation	5	7	5	7
Total jet/E_T^{miss} reconstruction	14	42	14	43

15.5 τ-lepton Identification

The High-Level Trigger algorithms for τ identification are designed to be used in the selection of isolated τ leptons such as those expected in the MSSM Higgs decays $A^0/H^0 \rightarrow \tau\tau$ and $H^+ \rightarrow \tau\nu$. The final-state signatures involve events with a lepton plus a τ-jet, two τ-jets or only one τ-jet. The algorithms are studied at both low and high luminosities, and for the low luminosity case two detector configurations are considered, a full pixel system and the possible staged pixel system (Section 14.2.1). The evaluation of the algorithm performance consists of measuring the selection efficiencies of the benchmark signal samples $A^0/H^0 \rightarrow \tau\tau$ and $H^+ \rightarrow \tau\nu$ and the corresponding efficiency of the QCD di-jet background. An important element of the analysis, like all other HLT analyses, is the minimization of the CPU time required for the selection. Both signal and QCD di-jet background samples are required to pass the Level-1 Trigger selections described in Section 15.1.3, "Level-1 Trigger Settings and Rates".

The QCD two-jet events were generated with PYTHIA in several different bins of P_T^{hard}. Most of the contribution to the Level-1 τ-trigger rate (~85–95%) comes from three bins: $50 < P_T^{hard} < 80$ GeV/c, $80 < P_T^{hard} < 120$ GeV/c and $120 < P_T^{hard} < 170$ GeV/c. These bins were used to evaluate the rejection factor of the HLT against the QCD background. Signal events were selected at the generator level with loose offline analysis requirements on leptons (e and μ) and τ-jets [15-13], [15-14] and [15-15]:

1. for $A^0/H^0 \rightarrow 2\tau \rightarrow l$ + τ-jet: $P_T^l > 14$ GeV/c, $P_T^{\tau\text{-jet}} > 30$ GeV/c, $|\eta^{l,\,\tau\text{-jet}}| < 2.4$;
2. for $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$-jet: $P_T^{\tau\text{-jet}} > 45$ GeV/c, $|\eta^{\tau\text{-jet}}| < 2.4$;
3. for $H^+ \rightarrow \tau\nu \rightarrow \tau$-jet: $P_T^{\tau\text{-jet}} > 80$ GeV/c, $|\eta^{\tau\text{-jet}}| < 2.4$.

The following definitions are used in the rest of the current section:

- first (second) “Level-1 τ-jet” is the first (second) jet from the τ-jet list provided by the Global Calorimeter trigger. Jets in this list are ordered in E_T (the first jet has the highest E_T).
- first (second) “Calo jet” is a jet reconstructed with the calorimeter in a region centred on the first (second) Level-1 τ-jet.

As in the previous sections, the nominal pixel detector configuration with three barrel layers and two forward disks is referred to as the “full” pixel detector whereas the start-up detector with only two barrel layers and one forward disk is referred to as the “staged” pixel detector configuration.

The identification of τ-jets involves information from both the calorimeter and tracking detectors. The following sections describes the details.

15.5.1 Calorimeter-based τ Selection.

The τ lepton decays hadronically 65% of the time, producing a “τ-jet”, which is a jet-like cluster in the calorimeter containing a relatively small number of charged and neutral hadrons. When the P_T of the τ-jet is large compared to the τ mass, these hadrons have relatively small momentum in the plane transverse to the τ-jet axis. In 77% of hadronic τ decays, the τ-jet consists of only one charged hadron and a number of π^0s (“one prong” decays). Because of these features hadronic τ decays produce narrow jets in the calorimeter. About 90% of the τ-jet energy is contained in a very small cone of radius 0.15 to 0.20 and about 98% in a cone of 0.4 for τ-jets with $E_T > 50$ GeV.

In the selection, isolation in the electromagnetic calorimeter is used. The parameter P_{isol} is defined as:

$$P_{isol} = \sum_{\Delta R < 0.4} E_T - \sum_{\Delta R < 0.13} E_T$$

where the sums run over transverse energy deposits in the electromagnetic calorimeter, and ΔR is the distance in η-φ space from the τ-jet axis.

The identification of τ-jets begins with the reconstruction of a jet in a region centered on the Level-1 τ-jet. The Iterative Cone algorithm [15-16] with a cone size of 0.6 is used. The algorithm uses as input only the calorimeter towers located within a cone of radius 0.8 around the Level-1 τ-jet direction. Restricting the number of towers considered by the jet finder results in a considerable speed-up of the jet finding process. For each jet found, the electromagnetic isolation parameter P_{isol} is calculated. Jets with $P_{isol} < P_{isol}^{cut}$ are considered as τ candidates.

The identification of a τ-jet with the calorimeter will be referred to as the "Calorimeter Tau Trigger". The efficiency of the Calorimeter Tau Trigger for $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$-jet and QCD di-jet background events when the value of P_{isol}^{cut} is varied is shown in Figure 15-49 for both low and high luminosities when calorimeter τ identification is applied to the first calorimeter jet.

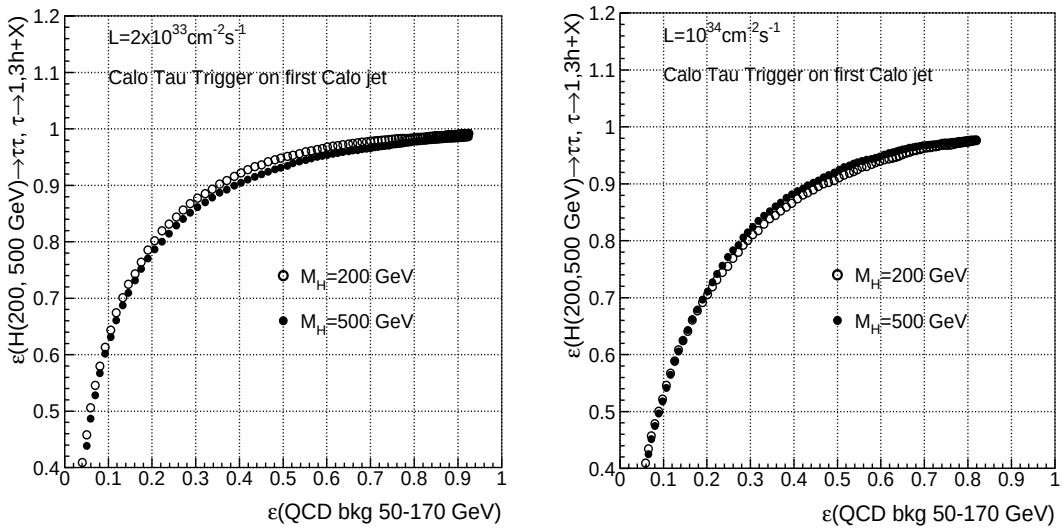


Figure 15-49 Efficiency of the Calorimeter Tau Trigger when applied to the first calorimeter jet in $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$-jet and QCD di-jet events. M_H = 200 and 500 GeV/c^2 for low (left) and high (right) luminosity.

The efficiency of the Higgs selection is essentially independent of the Higgs mass. Details of the Calorimeter Tau Trigger optimization can be found in [15-17]. The entire selection procedure is very fast, as can

Table 15-15 CPU time, in ms of a 1 GHz CPU, for τ-jet identification with the calorimeter for QCD di-jet events.

Reconstruction step / luminosity	2×10^{33} cm^{-2}s^{-1}	10^{34} cm^{-2}s^{-1}
Build calorimeter towers	24	39
Regional finding of calo jets and P_{isol} calculation	9	15
Total time	33	54

be seen in Table 15-15 where the times required to build calorimeter towers, for regional jet-finding and for the calculation of the isolation parameter P_{isol} are listed.

15.5.2 τ Identification with the Pixel Detector

The identification of a τ-jet using charged particle tracks is also based on isolation criteria. Calorimeter triggers provide a region in which isolated groups of tracks that are well matched to the jet axis given by the calorimeter can be searched for.

The corresponding isolation criteria require the reconstruction of low-P_T tracks with good efficiency and an acceptably low ghost rate. A precise measurement of the track P_T is not required. These requirements can be met by a fast track-finding algorithm using only pixel data. Identification of τ-jets with the pixel detector will be referred to as the "Pixel track Tau Trigger".

The principle of τ-jet identification using the pixel detector is shown in Figure 15-50. The direction of the τ-jet is defined by the axis of the calorimeter jet. The track-finding algorithm first reconstructs all track candidates (pixel lines) and then reconstructs the interaction vertices from the tracks using the histogramming method. Track candidates in a matching cone R_m around the jet direction and above a threshold P_T^m are considered in the search for the signal tracks. The leading signal track (tr_1 in Figure 15-50) is defined as the track with the highest P_T. Similarly, the vertex from which the leading signal track originates is considered to be the signal vertex (SV). Any other track from the SV which is in the narrow signal cone R_s around tr_1 is assumed to come from the τ decay. Tracks from the SV and with transverse momentum above a threshold P_T^i are then searched for inside a larger cone of opening angle R_i. If no tracks are found in the R_i cone, except for the ones which are already in the R_s cone, the isolation criteria are fulfilled. The matching cone size R_m, signal cone size R_s, and the thresholds P_T^m and P_T^i have been optimized for 1- and 3-prong τ-jets from $A^0/H^0 \rightarrow \tau\tau$ decays for Higgs bosons with mass $M_H \geq 200$ GeV/c^2 (the values are: $R_s = 0.07$, $R_m = 0.10$, $P_T^m = 3$ GeV and $P_T^i = 1$ GeV) [15-18]. The only remaining free parameter is the size of the isolation cone R_i.

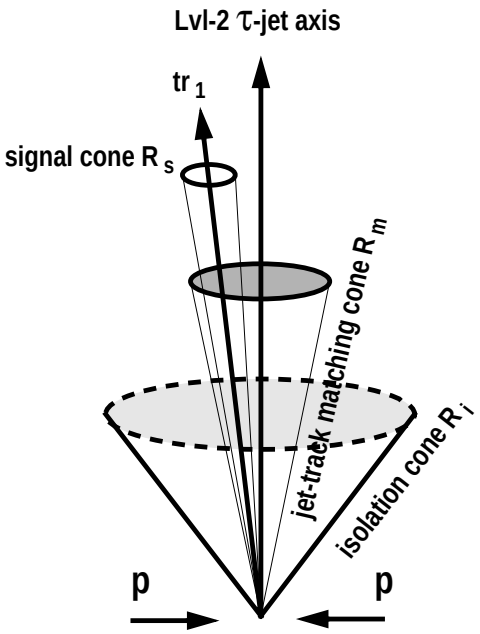


Figure 15-50 Sketch of the basic principle of τ-jet identification using charged particle tracks.

The efficiency of the Pixel track Tau Trigger, at both low and high luminosity is shown in Figure 15-51 when the size of the isolation cone is varied in the range 0.20-0.50. Pixel τ identification has been applied for the first calorimeter jet in $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$-jet and QCD di-jet events. The degradation of the performance at high luminosity with respect to low luminosity is due to the larger readout inefficiency of the pixel detector at high luminosity as well as the contamination of charged-particle tracks from pile-up in the isolation cone. The selection is essentially independent of the Higgs mass. Comparison between the full and staged pixel systems shows that, at constant QCD background rate, the signal efficiency is degraded by approximately 10% in the staged scenario.

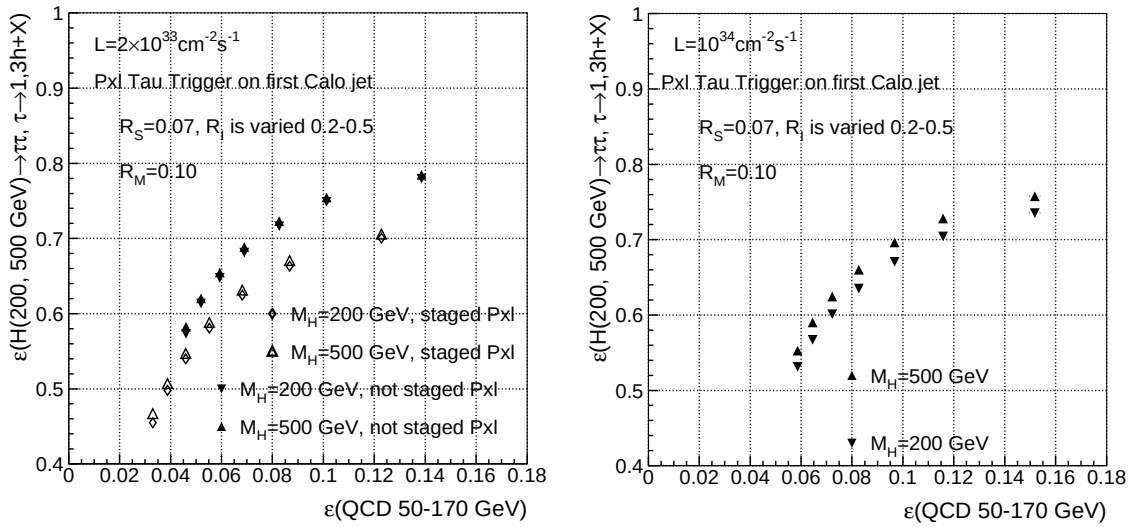


Figure 15-51 Efficiency of Pixel track Tau Trigger for the first calorimeter jet in $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$ - jet, for two Higgs masses, M_H = 200 and 500 GeV/c^2, versus the efficiency for QCD di-jet background events. Left: at a luminosity of 2×10^{33}cm^{-2}s^{-1}, for both the full and the staged pixel systems, and right: at 10^{34}cm^{-2}s^{-1}.

The timing performance of the Pixel track Tau Trigger algorithm is shown in Table 15-16. It includes the time to reconstruct clusters from the pixel digitized data and the time for reconstruction of the pixel lines and vertices. The time used by the isolation algorithm itself is negligible.

Table 15-16 CPU time, in ms of a 1 GHz Intel Pentium-III CPU, for τ-jet identification with the pixel detector for QCD di-jet events.

	Time (ms)	
Reconstruction step	**L=2×10^{33} cm^{-2}s^{-1}**	**L=10^{34} cm^{-2}s^{-1}**
Build clusters from pixel “digis”	44	100
Reconstruct pixel lines and vertices	34	262
Total time	78	362

15.5.3 τ Identification Using Regional Track Finding

In this section an algorithm to select τ-jets using information from reconstructed tracks is described. This selection will be referred to as the “Track Tau Trigger”. The Track Tau Trigger performs the reconstruction of only those tracks in restricted regions of interest (“regional tracking”), defined by the cones around

each jet direction given by the calorimeter jet reconstruction. The signal vertex is first defined using the pixel detector, as the vertex with the maximum $\Sigma|P_T|$ of the associated pixel lines.

The regional tracking concept is applied at the seeding level ("regional seeding") in order to minimize the seed multiplicity, significantly reducing the CPU time of the algorithm. A special track finder uses pixel lines in a cone of $\Delta R = 0.5$ around the calorimeter jet direction as track seeds. After track reconstruction only tracks with z_0 compatible with the signal vertex are taken into account[1]. A further improvement of the timing performance is obtained by stopping the track reconstruction when six hits have been associated with the helix. This gives an acceptable resolution on the track parameters and a low ghost rate.

At the seeding level the "2-hit recovery" option which uses two pixels hits out of three pixels layers has been adopted since it increases the seeding efficiency.

15.5.3.1 Track Tau Trigger Algorithm

The Track Tau Trigger [15-19] relies on an isolation requirement similar to the one used in the Pixel Tau Trigger. To reduce the contamination from soft tracks, only tracks with $P_T > 1$ GeV/c and z-impact parameter compatible with the z position of the signal vertex are considered. The number of tracks within a signal cone and within an "isolation cone" are counted. The cones are defined around the direction of the leading track, defined as the highest-P_T track found in the "matching cone" (ΔR_M=0.1) around the calorimeter jet direction. The isolation requirement is that there be no tracks in the isolation cone except those contained within the inner, signal, cone. The cone sizes are chosen to optimize the signal efficiency and background rejection, and also to minimise the dependence of the efficiency on the jet energy. Higher background reduction can be obtained by requiring the transverse momentum of the leading track, P_T^{LT}, to exceed a few GeV/c, but due to the strong dependence of the track P_T spectrum on the Higgs mass and on the τ hadronic decay mode, particular attention was required in optimising this cut.

15.5.3.2 Track Tau Trigger Performance for Single-τ Tagging

Figure 15-52 shows the Track Tau Trigger performance when applied to the first calorimeter jet in $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$-jet signal events, and to QCD di-jet background events. The different points correspond to different sizes of the isolation cone ΔR_{ISO} which was varied between 0.2 and 0.45. The left plot shows the performance at 2×10^{33} $cm^{-2}s^{-1}$ for both the full and staged pixel detector configurations. For the same QCD di-jet background efficiency, the signal efficiency is reduced by about 15% in the staged pixel scenario. Since the algorithm has to fulfill stringent CPU time limitations (discussed later in this chapter), particular attention has been paid to its optimization.

15.5.4 $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$ - jet Selections with Calorimeter and Pixel Triggers

A complete HLT selection for $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$-jet events can be defined using the Calorimeter Tau Trigger selection applied to the first calorimeter jet, and the Pixel track Tau Trigger selection applied to both calorimeter jets. The performance of this selection is presented in this section. Table 15-17 gives the purity of the selected jets in $gg \rightarrow bbA^0/H^0$, $A^0/H^0 \rightarrow \tau\tau \rightarrow 2\tau$-jet events with $M_H = 500$ GeV/c^2, defined as the fraction of calorimeter jets which correspond to a true τ-jet. To increase the purity of the second jet the following search algorithm is applied: if a second Level-1 τ-jet does not exist in the list or it exists but the E_T of the second calorimeter jet is less than 50 (70) GeV at low (high) luminosity, a new calorimeter jet in

1. z_0 is the z coordinate of the point on the track which has minimum distance from the z-axis.

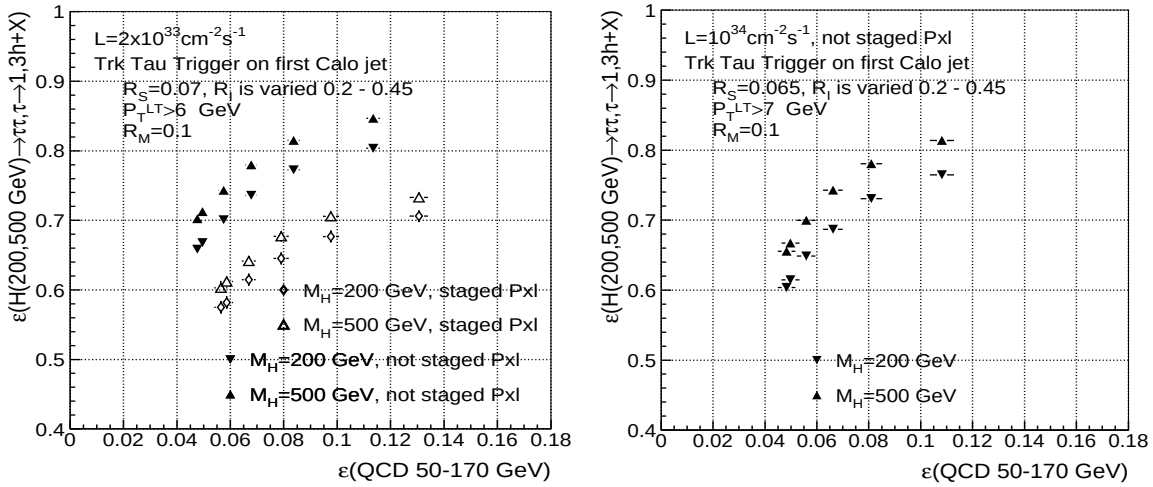


Figure 15-52 Efficiency of Track Tau Trigger for the first calorimeter jet in $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$ - jet, for two Higgs masses, M_H = 200 and 500 GeV/c^2, versus the efficiency for QCD di-jet background events. Left: at a luminosity of 2×10^{33}cm^{-2}s^{-1}, for both the full and the staged pixel systems, and right: at 10^{34}cm^{-2}s^{-1}.

the region centred on first Level-1 Central jet is reconstructed. The purity of the second jet when chosen in such a way is increased to ~90% (the numbers in parentheses in Table 15-17 correspond to the purity of the selection when this re-definition algorithm is not applied).

Table 15-17 Purity of calorimeter jets in events at low and high luminosity. Numbers for the 2nd jet without (with) parentheses are the purity after (before) re-definition of the 2nd jet.

L=2×10^{33}cm^{-2}s^{-1}		**L=10^{34}cm^{-2}s^{-1}**	
1st jet	**2nd jet**	**1st jet**	**2nd jet**
0.98	0.90 (0.63)	0.98	0.88 (0.65)

Usage of the Calorimeter Tau Trigger as a pre-selector before applying the Pixel track Tau Trigger allows considerable reduction of the total CPU time per event. The cuts are optimized by examining the background rejection, S_{calo}, of the Calorimeter Tau Trigger step, the efficiency for the signal, and the CPU time usage, while keeping a suppression factor of the full High-Level Trigger selection of ~ 10^3. Results of such a study for M_H = 200 GeV/c^2 are presented in Table 15-18 for the low luminosity scenario with the full pixel system. It has been found that for for a total suppression factor of 10^3, at both low and high luminosity, a calorimeter suppression factor of three yields the highest signal efficiency. At this operating point the total time T_{tot} of the full path is 59 ms for low luminosity and 174 ms for high luminosity.

Table 15-18 Efficiency for $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$ - jet events, and total CPU time, as a function of the Calorimeter isolation cut and its background rejection factor at a luminosity of 2×10^{33}cm^{-2}s^{-1}. An overall suppression factor 10^3 for background events is maintained. The bolded column corresponds to the operating point.

Cut on the isolation parameter P_{isol}, GeV	-	10.4	7.6	**5.6**	4.6	4.0	3.4	3.2	2.6
Background rejection factor, S_{calo}	1.0	1.5	2.0	**3.0**	4.0	5.0	6.2	7.5	10.0
$T_{tot}=T_{calo} + T_{pixel} / S_{calo}$ (ms)	110	85	72	**59**	52	50	45	43	41
Efficiency of Calo + Pixel Tau selection	0.35	0.37	0.40	**0.41**	0.40	0.39	0.37	0.36	0.35

Figure 15-54 shows the efficiency of the Calorimeter+Pixel track Tau Trigger selection for the signal and for QCD di-jet background events at low and high luminosity. The size of the isolation cone ΔR_i is varied in the range 0.20 to 0.50, and the optimal suppression factor of 3 for the Calorimeter Tau Trigger is used. For a total suppression factor of ~10^3, there is little difference in efficiency between the staged and full pixel configurations at low luminosity.

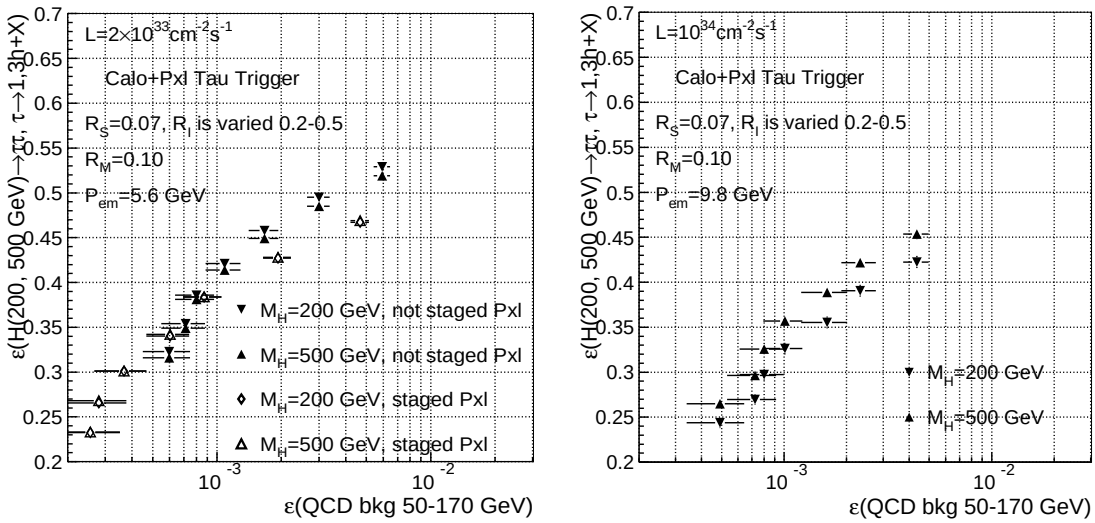


Figure 15-53 Efficiency of the Calo+Pxl Tau Trigger path for $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$ - jet and QCD 2-jet background events when the size of the isolation cone R_i is varied in the range 0.20-0.50. The optimal suppression factor of three for the Calo Tau Trigger is taken. Results for two Higgs masses, M_H=200 and 500 GeV/c^2, are shown. Left: L=2×10^{33}cm^{-2}s^{-1}, with the full and staged pixel systems. Right: L=10^{34}cm^{-2}s^{-1}.

15.5.5 $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$ - jet Selections with Track Tau Trigger.

In this section the performance of a complete HLT selection for $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$-jet is discussed, when the Calorimeter Tau Trigger selection, applied on the first calorimeter jet, is followed by the Track Tau Trigger on both calorimeter jets (Calo+Track Tau Trigger path). Figure 15-54 shows the signal vs. background algorithmic efficiency when the Track Tau Trigger selection is applied to both calorimeter jets, at low luminosity (left plot), for both the complete and staged pixel detector, and at high luminosity (right plot). The different points correspond to different sizes of the isolation cone, ΔR_{ISO}, which is varied between 0.2 and 0.45. At a QCD di-jet background efficiency of ~10^{-3} the signal efficiency with the staged pixel detector is reduced by about 20%. A cut on P_T on the leading track of 6 (7) GeV/c is required at low (high) luminosity to reach a background rejection factor ~10^3.

Figure 15-54 shows the algorithmic efficiency of the signal vs that of the background when the Calorimeter Tau Trigger selection, applied on the first calorimeter jet, is followed by the Track Tau Trigger on both calorimeter jets (Calo+Track Tau Trigger path). Table 15-19 summarizes the efficiency of the Track Tau Trigger at the operating points where the background rejection factor is ~10^3. These points have been chosen taking into account the overall performance and minimizing the τ-jet energy dependency of the signal efficiency. The third and fifth rows of Table 15-19 list the efficiency at low and high luminosity for the full Calo+Track Tau Trigger selection.

The Track Tau Trigger CPU time distribution is shown in Figure 15-56 for Higgs and QCD di-jet background events at low luminosity. The time needed for a double tag is only slightly larger than the time

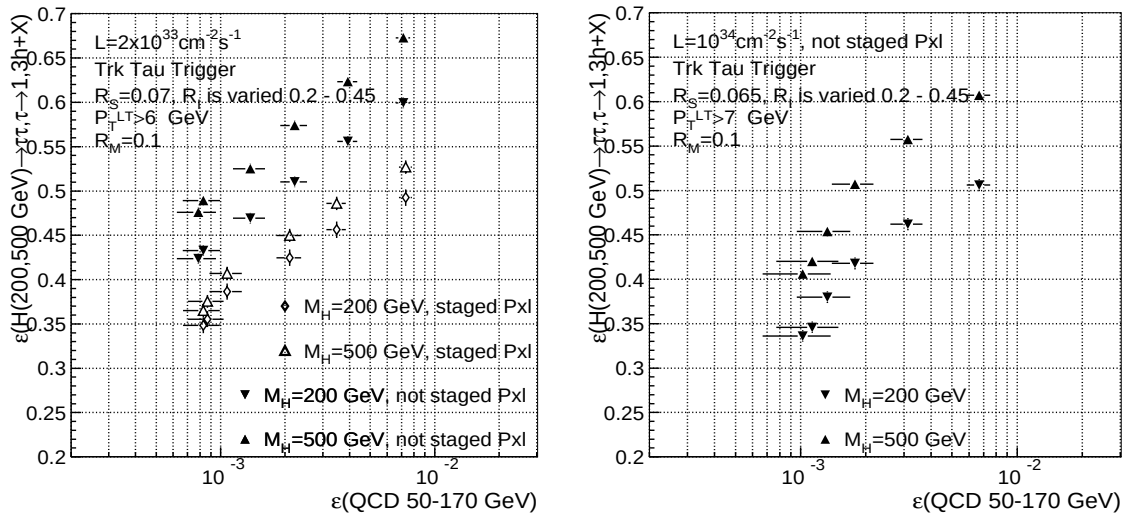


Figure 15-54 Efficiency of Track Tau Trigger applied to both Calo jets in $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$ - jet vs that in QCD di-jet event. Left: for L = 2×10^{33}cm^{-2}s^{-1}, for both the full and staged pixel systems. Right: for L = 10^{34}cm^{-2}s^{-1}. Two Higgs masses M_H = 200 and 500 GeV/c^2, are shown.

Table 15-19 Summary of Track Tau Trigger efficiency when two Calo jets are tagged. A QCD di-jet background rejection (last column) of ~10^3 is required. The third and fifth rows show the results when the Calorimeter Tau Trigger selection applied on the first Calo jet is followed by the Track Tau Trigger on both Calo jets. Due to the limited Monte Carlo statistics some statistical errors for the QCD background are large.

Luminosity	Configuration/Trigger	M_H = 200 GeV/c^2	M_H = 500 GeV/c^2	QCD
2×10^{33}cm^{-2}s^{-1}	Staged pixels, Track Tau	0.355±0.006	0.375±0.005	$(8.6\pm1.6)\times10^{-4}$
2×10^{33}cm^{-2}s^{-1}	Full pixels, Track Tau	0.433±0.006	0.489±0.005	$(8.3\pm1.6)\times10^{-4}$
2×10^{33}cm^{-2}s^{-1}	Full pixels, Calo+Track Tau	0.446±0.006	0.486±0.005	$(1.0\pm0.2)\times10^{-3}$
10^{34}cm^{-2}s^{-1}	Track Tau	0.346±0.006	0.420±0.005	$(1.13\pm0.4)\times10^{-3}$
10^{34}cm^{-2}s^{-1}	Calo + Track Tau	0.361±0.006	0.427±0.005	$(9.4\pm3.0)\times10^{-4}$

needed for a single tag, since the second calorimeter jet is analysed only in the ~ 6% of background events which pass the single tag on the first calorimeter jet. Only 10% of the QCD events require more than 400 ms to be analysed.

An important advantage of the Calo+Track Tau Trigger selection over the simple Track Tau Trigger is its better time performance, which is due to the fact that only those events which pass the calorimeter τ selection on the first jet need to be analysed by the Track Tau Trigger. Following the same method described in Section 15.5.4 and using the numbers from Table 15-18 (S_{calo} = 3, $T_{calo} \cong$ 33 ms) an average time of about 130 ms/event is expected at low luminosity.

The time required by the Track Tau Trigger at high luminosity is much larger and is currently estimated at ≅ 1 s/event. However, making use of the Calo+Track Tau Trigger selection**,** this time will be less than 400 ms/event, which is acceptable. A further improvement is expected with the use of a regional pixel reconstruction.

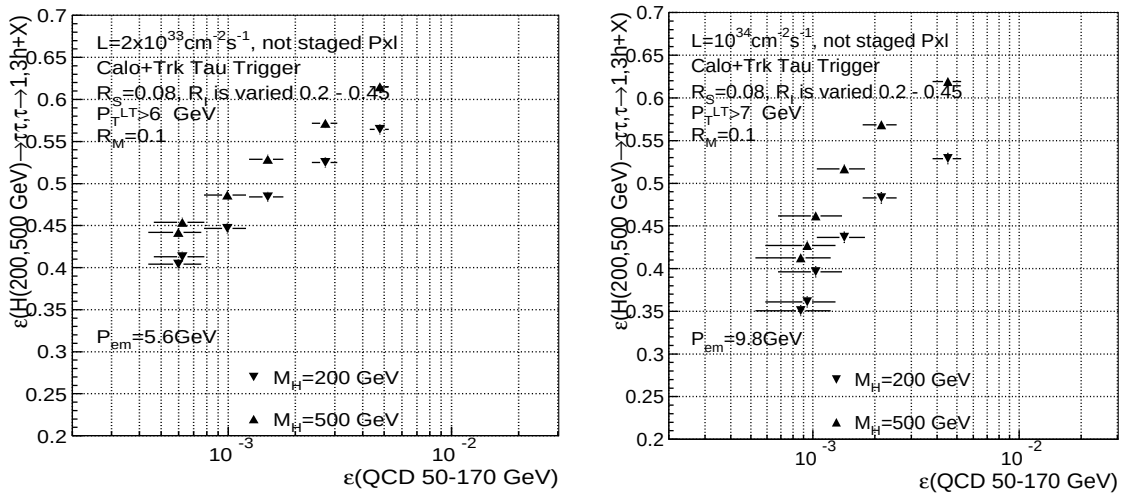


Figure 15-55 Efficiency for $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$ - jet vs that in QCD di-jet events, when the Calorimeter Tau Trigger selection applied on the first Calo jet is followed by Track Tau Trigger on both calorimeter jets (Calo+Track Tau Trigger path). Results are shown for two Higgs masses M_H = 200 and 500 GeV/c^2 for low (high) luminosity on the left (right).

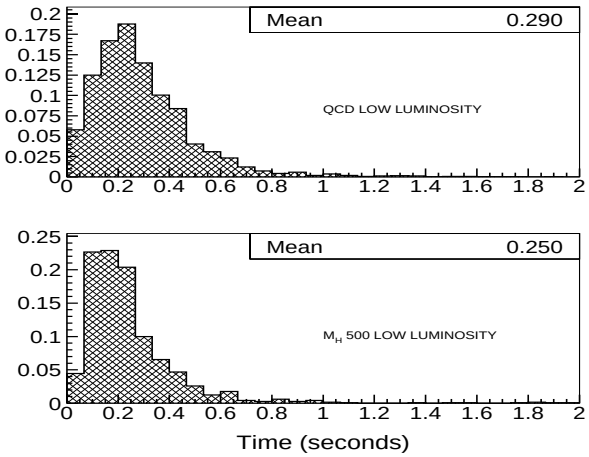


Figure 15-56 Track Tau Trigger reconstruction time (in seconds) for double tagging with the Track Tau Trigger at low luminosity. Upper plot: QCD di-jet events. Lower plot: $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$ - jet events.

15.5.6 Summary of Calo+Pixel and Calo+Track Trigger Selections for $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$ - jet

Two HLT selection paths are considered as most suitable for the selection of MSSM A^0 and H^0 bosons produced in the process $gg \rightarrow bbA^0/H^0$, $A^0/H^0 \rightarrow 2\tau \rightarrow 2\tau$-jet ($M_H$ = 200 and 500 GeV/c^2): the Calo+Pixel track Tau Trigger and the Calo+Track Tau Trigger. The efficiencies of the two selections, at the same background suppression factor ~10^3, as well as the timing performance, are compared for both low and high luminosity running conditions. The Calo+Pixel track Tau Trigger is approximately twice as fast as the Calo+Track Tau Trigger (59 vs 130 ms at low luminosity and 170 vs ~400 ms at high luminosity), but is also ~15% less efficient. The loss of Calo+Pixel track Tau Trigger efficiency could be compensated for by increasing of Level-1 bandwidth allocated to the single and double Tau trigger.

15.5.7 $H^+ \to \tau\nu \to \tau$ - jet Selections with Track Tau Trigger.

The Track Tau Trigger will work as the final part of the selection MSSM charged Higgs bosons in the process $gb(g) \to H^+t(b)$, $H^+ \to \tau\nu \to \tau$-jet, $t \to bjj$. The Level-1 single-τ trigger followed by a HLT selection of events with E_T^{miss} > 65 (95) GeV at low (high) luminosity gives an output rate of about 30 (70) Hz. The selection must provide a suppression factor of ~30 to match the output bandwidth requirements. The Track Tau Trigger selection is applied to the calorimeter jet reconstructed in the region of the first Level-1 τ-jet. The purity of the calorimeter jet in signal events is 85%.

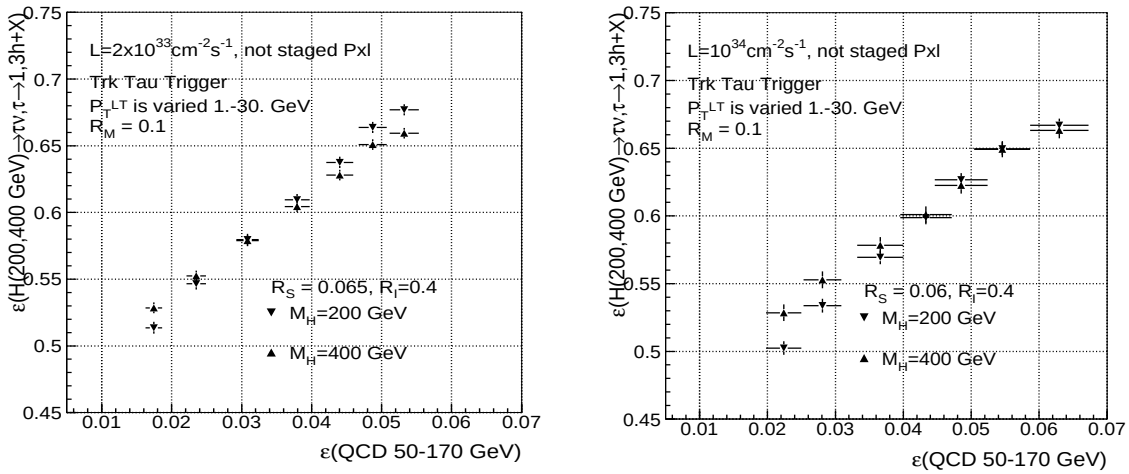


Figure 15-57 Track Tau Trigger efficiency for $H^+ \to \tau\nu \to \tau$-jet events vs the QCD di-jet background efficiency. The results are given for M_H = 200 GeV/c^2 and M_H = 400 GeV/c^2. Isolation parameters are: ΔR_M = 0.1, ΔR_{SIG} = 0.065 (0.060) for low (high) luminosity, ΔR_{ISO} = 0.4; cut on P_T of the leading track is varied from 1 to 30 GeV/c. Left plot: L = 2×10^{33}cm^{-2}s^{-1}, with the full pixel system; Right plot: L = 10^{34}cm^{-2}s^{-1}.

The isolation criteria used in the Pixel or Track Tau Triggers cannot provide the required suppression factor. Additional background rejection is obtained by applying a cut on the P_T of the leading track in the Track Tau Trigger . Figure 15-57 shows the Track Tau Trigger efficiencies for the signal and QCD background passing the Level-1 single-τ trigger. Since the cut on E_T^{miss} and the Track Tau selection efficiency are uncorrelated [15-19], a cut on E_T^{miss} has not been applied due to limited Monte Carlo statistics.

Table 15-20 Track Tau Trigger efficiency for the process $gb(g) \to H^+t(b)$, $H^+ \to \tau\nu \to \tau$-jet, $t \to bjj$ and for the QCD background. The efficiencies for the signal are presented for a background suppression factor of ~30.

Signal and background samples		**Efficiency of Track Tau Trigger**	
		Low Luminosity P_T^{LT} > 20 GeV/c ΔR_{SIG}=0.065	**High Luminosity P_T^{LT} > 25 GeV/c ΔR_{SIG} = 0.06**
$H^+ \to \tau\nu$	M_H= 200 GeV/c^2	0.580 ± 0.004	0.534 ± 0.005
	M_H= 400 GeV/c^2	0.579 ± 0.004	0.553 ± 0.006
QCD	50 < P_T < 170 GeV/c	0.031 ± 0.002	0.028 ± 0.003

The different points in Figure 15-57 correspond to varying the cut on the P_T of the leading track, P_T^{LT}, from 1 to 30 GeV/c. A rejection factor 30 can be reached with a 20 (25) GeV/c cut on P_T^{LT} at low (high)

luminosity, which is well below the foreseen offline analysis cut [15-15]. Table 15-20 summarizes the efficiency of the Track Tau Trigger at this working point with a background rejection factor of 30.

15.5.8 Triggering on Mixed Channels: $A^0/H^0 \rightarrow 2\tau \rightarrow e + \tau$ - jet

The triggering scheme for $gg \rightarrow bbA^0/H^0$, $A^0/H^0 \rightarrow 2\tau \rightarrow e+\tau$-jet has been studied for $M_H = 200$ GeV/c^2. The trigger accepts events which pass either a single-*e* or a combined *e*+τ-jet ('eTau') trigger. In what follows the combined trigger will be called the 'e+eTau' trigger. The eTau trigger requires the presence of both an electron and a τ-jet, with thresholds lower than those used in the single-*e* and single-τ-jet triggers. The τ-jet candidate at Level-1 is defined as the most energetic τ-jet that is not collinear with the electron candidate ($\Delta R > 0.3$). This condition avoids misidentification in signal events (the τ purity in signal events increases from 61% to 99%) with little influence on the overall efficiency.

15.5.8.1 High Luminosity

Figure 15-58 shows the Level-1 'e+eTau' trigger rate at 10^{34} cm^{-2}s^{-1} as a function of the electron and τ-jet thresholds used in the eTau trigger. The dots with numbers on the 'e+eTau' trigger isorate curves indicate the Level-1 selection efficiency for the signal. For a given Level-1 trigger rate, the efficiency increases when the electron threshold is reduced and the τ-jet threshold is raised. This results from the steeply falling P_T spectrum of the electron in the signal channel. Figure 15-59 shows the increase in Level-1 efficiency obtained by using the 'e+eTau' trigger, as opposed to just the single electron trigger, as a function of the extra bandwidth which one devotes to the eTau trigger. The curves are each obtained by fixing the threshold for the electron in eTau trigger and varying the threshold on τ-jet.

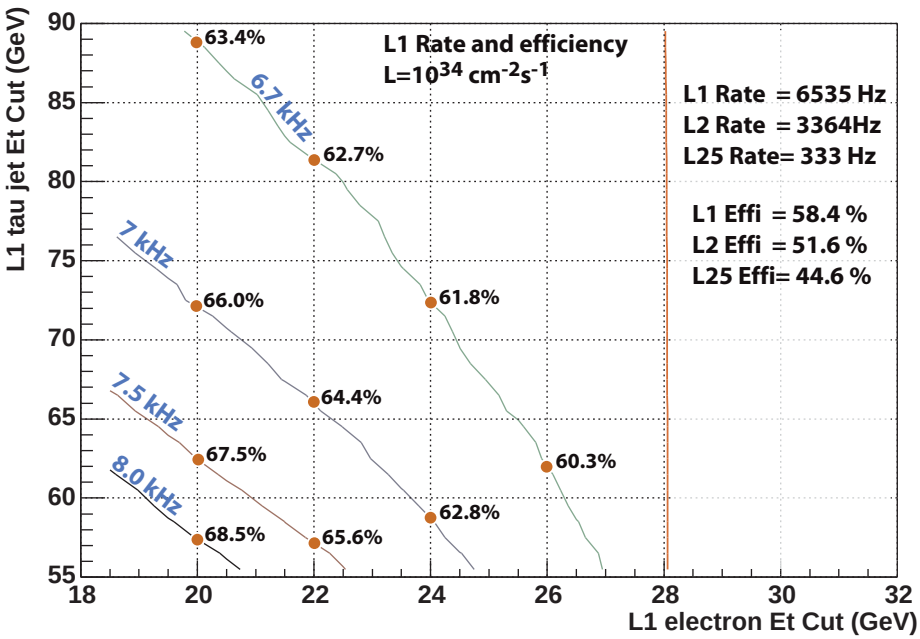


Figure 15-58 Level-1 'e+eTau' trigger rate at a luminosity of 10^{34}cm^{-2}s^{-1} as a function of the *e* and τ-jet thresholds, for a fixed single-*e* trigger threshold. The four curves correspond to additional bandwidths of 0.14 kHz, 0.39 kHz, 0.85 kHz and 1.28 kHz, respectively, being devoted to the eTau trigger, on top of the constant 6.54 kHz devoted to the single-*e* trigger. The vertical line at 28 GeV corresponds to the single-*e* trigger. The values listed in the upper right corner are the rates and efficiencies for the single-*e* trigger at Level-1, Level-2.0 and Level-2.5.

The HLT selection is applied independently on the electron stream and the eTau stream at Level-2.0 and Level-2.5. At Level-2.0, a threshold is applied only on the electron candidate (Section 15.2.4). At Level-2.5, pixel/super-cluster matching is used for the electron candidate (Section 15.2.5) and the τ-jet identification is applied as described in Section 15.5.2.

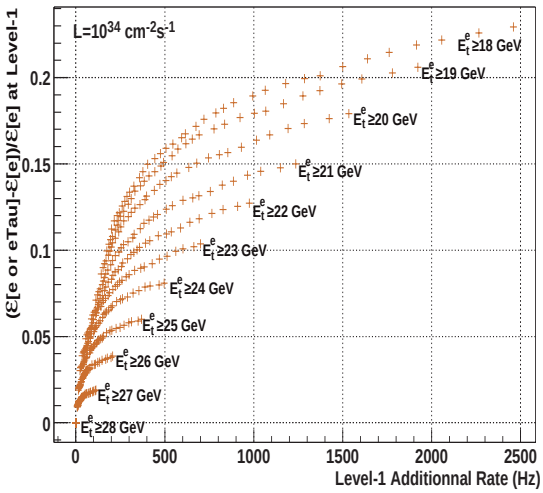


Figure 15-59 The increase in Level-1 efficiency, at $10^{34}cm^{-2}s^{-1}$, obtained using the 'e+eTau' trigger, as opposed to just the single electron trigger, as a function of the extra band-width devoted to the eTau trigger. Each curve is obtained by fixing the e threshold (E_T^e in the plot) and varying the threshold for the τ-jet.

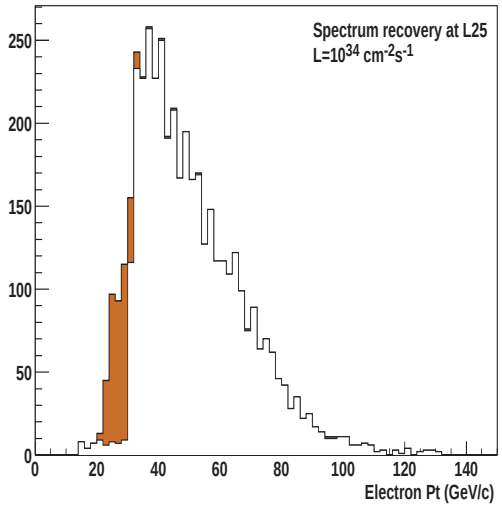


Figure 15-60 Electron P_T in $A^0/H^0 \rightarrow 2\tau \rightarrow e+\tau$ - jet events, at $10^{34}cm^{-2}s^{-1}$. The filled portion of the histogramme corresponds to the gain observed at Level-2.5 by adding the Level-1 e+τ trigger, the remaining unfilled part corresponds to electrons passing the single electron trigger.

Table 15-21 shows the details of the full selection for four scenarios. In both cases, the eTau trigger uses an electron threshold corresponding to 25.5 GeV on the Level-1 95% efficiency scale, while the τ threshold is varied in such a way that the rate added by the eTau trigger to the single-electron trigger rate is 0.14 kHz, 0.39 kHz, 0.85 kHz and 1.28 kHz respectively. Accepting an additional rate of 0.85 kHz at Level-1 leads to a relative improvement in efficiency, at Level-2.5, of about 10% at a price of a 7 Hz rate increase. This is illustrated in Figure 15-60 which shows the P_T spectrum of the electron that is recovered at Level-2.5 when the eTau trigger is added to the single electron trigger at Level-1.

15.5.8.2 Low Luminosity

Since the single electron thresholds are lower in the low luminosity scenario, less is gained by adding the combined eTau Trigger. The HLT selection scheme is the same as at high luminosity but the Level-1 thresholds are different. For a scenario where 0.82 kHz for the eTau trigger is added at Level-1, the relative gain in efficiency at Level-2.5 is ~4%.

15.5.9 Triggering on Mixed Channels: $A^0/H^0 \rightarrow 2\tau \rightarrow \mu + \tau$ - jet

The reconstruction efficiencies and online background rejection performance for $A^0/H^0 \rightarrow 2\tau \rightarrow \mu+\tau$-jet have been studied for the case when the Higgs particle is produced in association with b-quarks and the mass of Higgs particle is rather low (200 GeV/c^2). The focus is on the high-luminosity case, where the

Table 15-21 Evolution of the rate and the efficiency of the different trigger levels at high luminosity. Results for four different tau thresholds in the eTau trigger are shown, as are results obtained with no eTau trigger (just the single electron trigger).

		e + eTau	trigger		e trigger
L1 eTau thresh (GeV)	**(20, 57)**	**(20, 62)**	**(20, 72)**	**(20, 89)**	**(28)**
L1 Rate (Hz)	7819	7389	6933	6677	6535
L1 Additional Rate (Hz)	1284	854	398	142	
L1 Efficiency	0.685	0.675	0.661	0.634	0.584
L1 Additional Efficiency	0.101	0.091	0.077	0.050	
L2.0 Rate (Hz)	4219	3945	3631	3452	3364
L2.0 Additional Rate (Hz)	855	581	267	88	
L2.0 Efficiency	0.614	0.605	0.590	0.562	0.5165
L2.0 Additional Efficiency	0.098	0.089	0.074	0.046	
L2.5 Rate (Hz)	343	339	338	333	332
L2.5 Additional Rate (Hz)	11	7	6	1	
L2.5 Efficiency	0.492	0.489	0.483	0.469	0.446
L2.5 Additional Efficiency	0.046	0.043	0.037	0.023	

event selection and background reduction are more difficult. The quoted efficiencies are given with respect to "useful" events defined using Monte-Carlo parameters ($P_T^{\mu} > 14$ GeV/c, $P_T^{\tau\text{-jet}} > 30$ GeV/c, $|\eta^{\mu,\tau\text{-jet}}| < 2.4$), and which select 45% of all $H \rightarrow \mu+\tau$-jet generated decays. Due to the relatively low threshold of the Level-1 single-μ trigger ($P_T^{cut} = 20$ GeV/c), about 66% of the useful events are taken by the Level-1 single-μ or Level-1 single-τ triggers as shown in Figure 15-61.

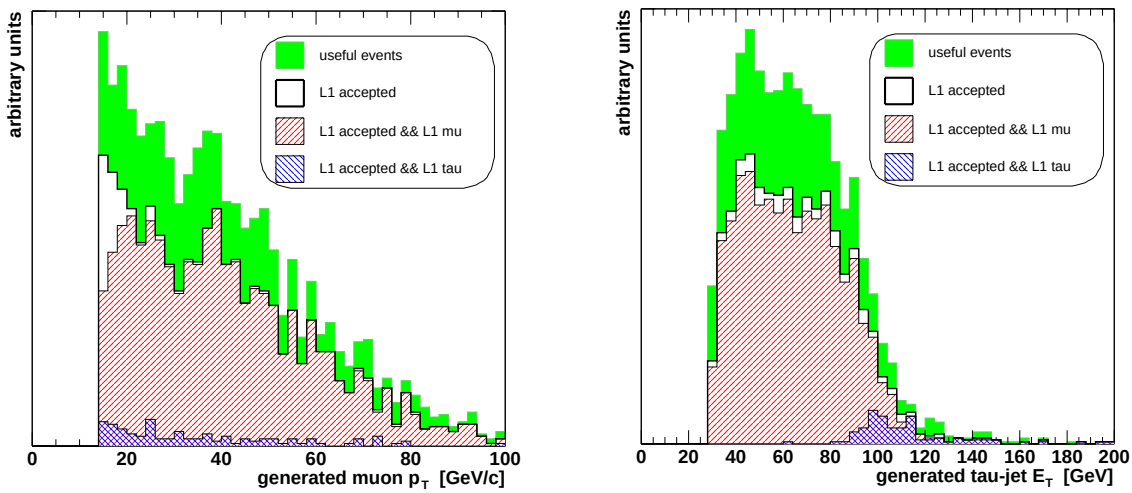


Figure 15-61 The P_T spectrum of generated muons (left) and the E_T spectrum of generated τ-jets (right). The events accepted by the μ–τ Level-1 Trigger (with combined threshold $P_T \geq 14$ GeV/c, $E_T \geq 45$ GeV) are shown on the top of useful events. The fraction of Level-1 events accepted by single-μ or single-τ trigger is indicated.

In Figure 15-62, the efficiency of the Level-1 μ–τ trigger is shown as a function of μ P_T and τ E_T. Only events where both the τ–jet and the μ are found by the trigger (irrespective of their P_T and E_T) contribute to the plot. The combined trigger increases the efficiency by up to a maximum of 73% for the loose P_T and E_T cuts. In order to preserve events passing the offline cuts $P_T > 15$ GeV/c, $E_T \geq 40$ GeV, one can choose the corresponding Level-1 working point with Level-1 $P_T \geq 14$ GeV/c and Level-1 $E_T > 45$ GeV with an overall efficiency of 70%.

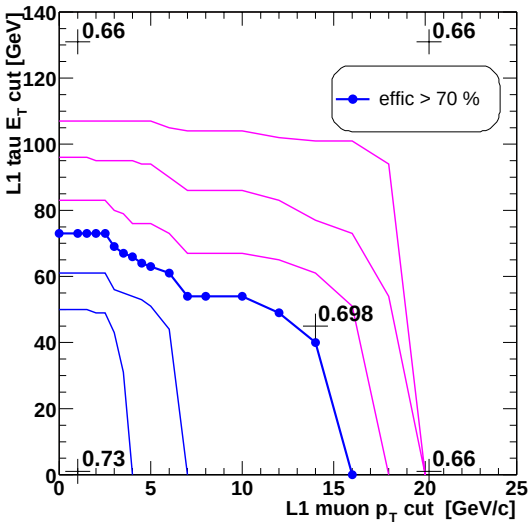


Figure 15-62 Level-1 Trigger efficiency as the function of the cuts for high luminosity. μ–τ events taken by the single-μ or single-τ trigger are included. The difference between contour lines represents a change in efficiency of 1%. The Level-1 cut corresponding to the offline cut selects ~70% of the reference events.

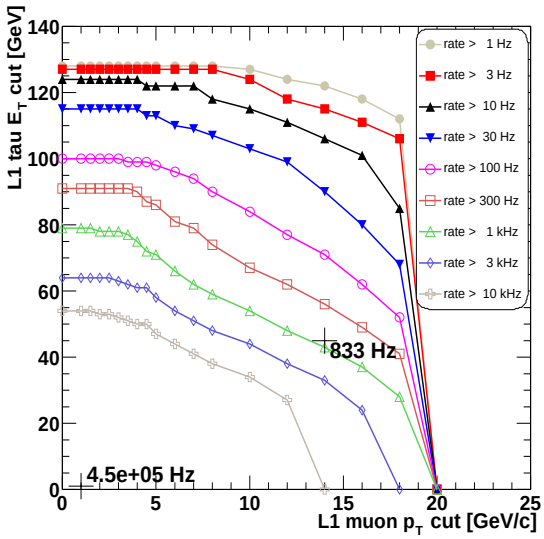


Figure 15-63 Level-1 Trigger rate at high luminosity from μ–τ events. This rate is in addition to the Level-1 single-μ and Level-1 single-τ trigger rates. The additional rate, for the Level-1 cut that corresponds to the offline cut, is 0.83 kHz.

The additional rate taken by the Level-1 combined μ–τ-jet trigger over that of the single-μ trigger (~6 kHz) and single τ-jet trigger (~2 kHz) is shown in Figure 15-63. The additional rate for the proposed working point is 0.83 kHz. The full rate for events selected by the combined trigger (both Level-1 μ and Level-1 τ found) passing the single-μ, single τ-jet or combined thresholds is 5.8 kHz.

The HLT analysis in this channel consists of:

- identification of Level-2 jet corresponding to Level-1 τ-jet with a E_T cut;
- calorimeter isolation cut for the τ;
- Level-2 μ identification and P_T cut;
- calorimeter isolation cut for the μ;
- Level-3 μ identification and P_T cut
- τ-jet identification and isolation requirement in the pixel detector
- μ isolation cut with full tracker (or with pixel detector)

The resulting HLT efficiencies and rates are shown in Figure 15-64 as the function of the HLT cuts. Only events passing the proposed Level-1 thresholds are included. Setting the HLT thresholds at the offline values preserves 32% of events in the reference useful sample. The corresponding rate from the muon mini-

mum-bias samples is about 1 Hz. The detailed list of rejection factors and efficiencies for each HLT step is given in Table 15-22.

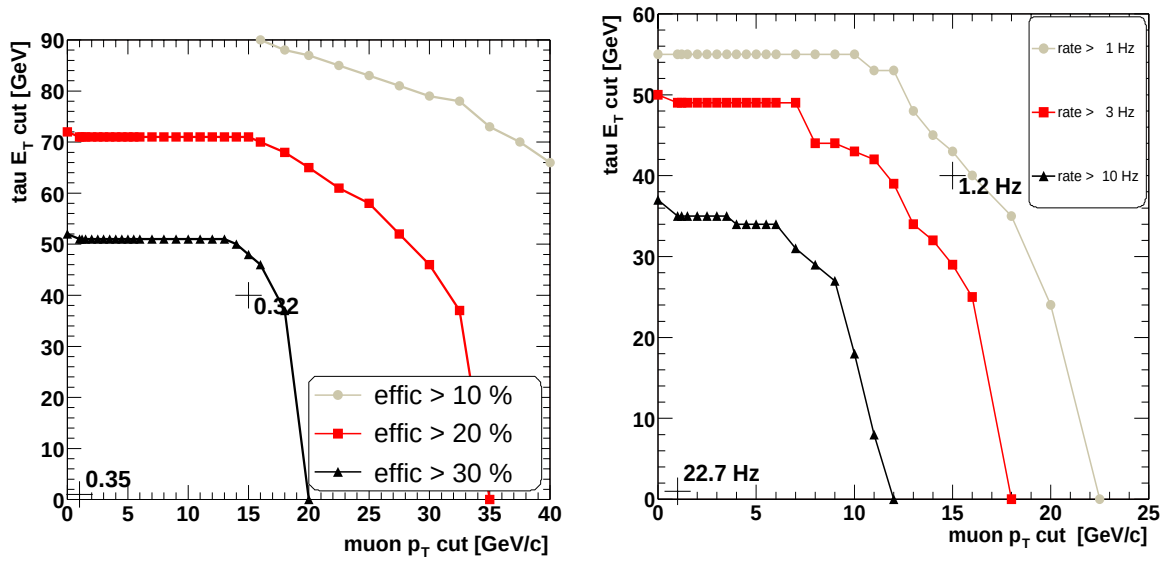


Figure 15-64 Full selection (Level-1 + HLT) in $H \rightarrow \tau\,\tau \rightarrow \mu + \tau$–jet channel. The selection efficiency (left) and corresponding background rate (right) are shown. The Level-1 combined μ–τ cut is at μ Level-1 P_T = 14 GeV/c and Level-1 τ-jet E_T = 45 GeV. Events must pass Level-1 and all HLT μ and τ selection requirements. The tau 95% efficiency scale is used for tau E_T. The μ 90% efficiency scale is used for μ P_T.The efficiency and rate for the offline threshold (P_T = 15 GeV/c, E_T = 40 GeV) are marked.

The low-luminosity case is simpler. The Level-1 single-μ threshold of 12 GeV/c is below the offline cut and there is no need to allocate bandwidth to the μ+τ-jet channel. The Level-1 single-μ and single-τ triggers (E_T >93 GeV) select about 72% of the events in the sample. The μ and τ identification and selection criteria in the HLT reduce this efficiency to 39% with a background rate of 0.2 Hz.

Table 15-22 Summary of Level-1 and HLT selection efficiencies and background rates in the $H \rightarrow 2\tau \rightarrow \mu + \tau$-jet channel for the thresholds corresponding to the offline cut at P_T = 15 GeV/c, E_T = 40 GeV. The efficiency is defined with respect to all events in the useful sample.

	Efficiency	Rate [Hz]
Events passing Level-1 single μ, single τ, or combined trigger	0.70	5.8×10^3
Events passing Level-1 combined trigger not selected by single-μ or single-τ trigger	0.04	830
L2 identification with E_T and P_T cuts	0.63	990
L2 and calo tau isolation	0.53	380
L2 and muon calo isolation	0.61	420
L2 combined	**0.51**	**150**
L3 identification with μ P_T cut	0.49	59
L3 and tau isolation	0.33	3.4
L3 and muon isolation	0.48	25
L3 combined	**0.32**	**1.2**

15.5.10 Summary of Level-1 and HLT for Higgs Channels with τ-leptons.

The following Level-1 and HLT paths are used to trigger on the MSSM A^0/H^0 and H^+ Higgs bosons with mass greater than 200 GeV/c^2:

1. $A/H \rightarrow \tau\tau \rightarrow 2\tau$-jets. At Level-1: single and double τ-jet; at HLT: calorimeter + tracker isolation.
2. $H^+ \rightarrow \tau\nu \rightarrow \tau$-jet+$E_T^{miss}$. At Level-1: single τ-jet; at HLT : calorimeter E_T^{miss} and tracker isolation on the τ-jet.
3. $A/H \rightarrow \tau\tau \rightarrow e+\tau$-jet. At Level-1: single-*e* and combined *e*+τ-jet triggers; at HLT: electron selection as described in Section 15.2 and τ-jet isolation with the tracker.
4. $A/H \rightarrow \tau\tau \rightarrow \mu+\tau$-jet. At Level-1: single-μ and combined μ+τ-jet triggers; at HLT: muon selections as described in Section 15.3 and τ-jet isolation with the calorimeter and the tracker.

The HLT output rates, the signal efficiency (for M_H=200 GeV/c^2) of the Level-1 and HLT selections as well as the CPU usage at both low and high luminosity are listed in Table 15-23

Table 15-23 The efficiency of the Level-1 and HLT selections, the HLT output rates and CPU usage for low and high luminosity (numbers in parentheses) for the MSSM Higgs to t decays discussed in the text.

Channel	Level-1 efficiency (%)	HLT output rate (Hz)	HLT efficiency (%)	HLT CPU (ms)
$A/H \rightarrow \tau\tau \rightarrow 2\tau$-jets	78 (62)	3 (8)	45 (36)	130
$H^+ \rightarrow \tau\nu \rightarrow \tau$-jet+$E_T^{miss}$	81 (76)	1 (2)	58 (53)	38
$A/H \rightarrow \tau\tau \rightarrow \mu+\tau$-jet	72 (70)	0.2 (1.2)	54 (46)	660
$A/H \rightarrow \tau\tau \rightarrow e+\tau$-jet	80 (69)	0.4 (1.8)	70 (71)	165

The efficiency of the combined triggers used for $A/H \rightarrow \tau\tau \rightarrow$lepton+τ-jet channels are only about 2-5 % higher than those of the single-lepton trigger. This is mostly due to the choice of a relatively high mass for the Higgs bosons. As an example, for Higgs masses around 120 GeV/c^2, the fusion channel $qq \rightarrow qqH$, $H \rightarrow \tau\tau$ can benefit significantly from a combined trigger.

15.6 *b*-jet Identification

Many physics channels contain hard *b*-jets in the final state which are hidden by the more copious light flavoured jets. Inclusive *b*-tagging of jet triggers can be used for selection of these channels. The algorithms used for *b*-tagging rely on the large value of the b-hadron proper time ($c\tau \sim 450\ \mu m$), which gives rise to tracks with large impact parameters with respect to the production vertex.

15.6.1 Tagging Algorithm

The algorithms which will run at the HLT stage should be robust and fast. A wide range of algorithms have been developed within CMS to tag *b*-jets [15-20]. The tag chosen for these studies relies upon the track impact parameter.

The track impact parameter can be calculated either in the transverse view (2D impact parameter) or in three dimensions (3D impact parameter): in the former case it is not affected by the uncertainty on the z-component of the primary vertex position while in the latter case a larger set of information can be exploited. In both cases the calculation is performed starting from the trajectory parameters at the innermost measurement point. In the 2D impact parameter case the estimate can be done analytically since the trajectory is circular in the transverse view. In the three-dimensional case the extrapolation is performed by iteration.

Figure 15-65 shows the main steps of the three-dimensional impact parameter calculation: the point of closest approach of the track to the jet direction, S, is extracted first. This point approximates the decay point of the *B* hadron. The tracks are then linearized at this point and their minimum distance from the primary vertex, V, is computed (i.e. the three-dimensional impact parameter is obtained). The VQ segment in Figure 15-65 is called the decay length and approximates the flight path of the *B* hadron.

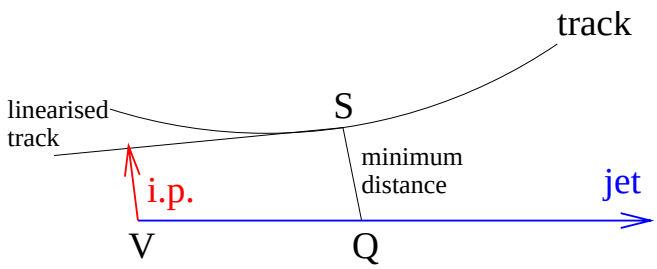


Figure 15-65 Representation (not to scale) of the definition of the track three-dimensional impact parameter.

The impact parameter is signed as positive if Q is upstream of V in the jet direction, and negative otherwise. The tracks from a *B* decay should have a positive impact parameter, while those coming from the primary vertex have an impact parameter comparable to the experimental resolution. The tag makes use of the track impact parameter significance, which is defined as the ratio between the value of the impact parameter and its error. A jet is defined to be tagged as a *b*-jet if there exist a minimum number of tracks exceeding a given threshold on the impact parameter significance.

In order to speed up the reconstruction, only tracks within a jet cone are used. The performance of the tagger depends crucially on the quality of the tracks and the jet direction. Tracks coming from secondary interactions with the material, K^0_S and Λ^0 decays are reduced by a cut on the maximum transverse impact parameter of 2 mm and on the decay length as estimated from the distance between the point of the track closest approach to the jet, which depends on the jet energy and rapidity and varies between 1.5 to 10 cm.

Optimization of the cuts was performed to maximise the b-tag signal efficiency at a fixed mistagging rate of 1 %. More sophisticated tags, such as the one developed by the ALEPH collaboration [15-21], take into account the quality of the tracks and give better performance. However, they are less robust for use at the trigger level, due to the large number of parameters which are needed.

15.6.2 Tagging Region

Tracks are reconstructed in a cone around the Level-1 calorimeter jet. The cone apex is taken as the pixel reconstructed primary vertex. The optimum cone width depends on the reconstructed jet E_T. The number of tracks from b-decays inside the jet cone is a function of the cone size, and also depends weakly on the jet E_T. The fraction of tracks coming from b-decays reaches a plateau value [15-22] at a $\Delta R \approx 0.25$, beyond this point only uninteresting tracks from the hadronization process are added. In the case of light flavour jets the number of tracks increases almost linearly. At high luminosity the proportion of fragmentation tracks to b-tracks increases, requiring a harder P_T cut.

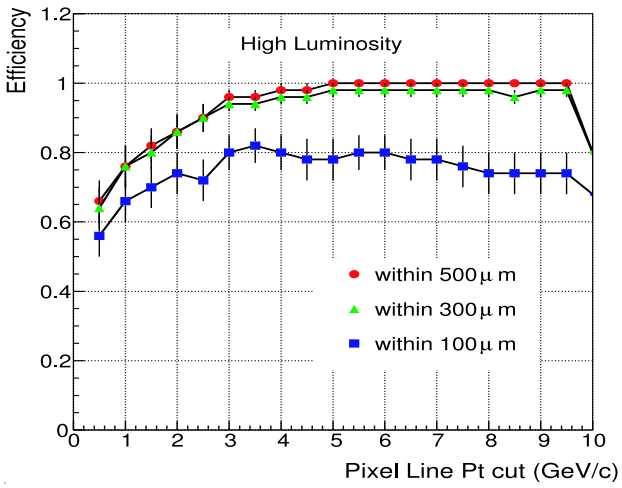


Figure 15-66 Efficiency of the pixel algorithm to correctly determine the primary vertex of the event within 100, 300 and 500 μm, at high luminosity, as a function of the minimum P_T cut on the pixel lines.

The primary vertex location along the beam line is reconstructed using the pixel detector, with the algorithm presented in [15-23]. The primary vertex is taken as the candidate vertex having associated to it pixel lines with the largest summed P_T. At low luminosity the algorithm has high efficiency. At high luminosity the algorithm is modified so that only pixel lines with large P_T are used, this gives increased efficiency with only a small loss of precision. Figure 15-66 shows the efficiency of the pixel algorithm to correctly assign the primary vertex within 100, 300 and 500 μm, at high luminosity, as a function of the minimum P_T cut on the pixel lines.

Primary vertex reconstruction takes about 50 msec on a 1 GHz Pentium-III CPU, for both the low and high luminosity cases, and it is only slightly dependent on the minimum pixel line P_T cut.

15.6.3 Track Reconstruction

Track reconstruction is based on the partial reconstruction of tracks using a regional approach as described in Section 13.4.5: starting from pixel seeds, further hits, compatible with the track P_T, are sought

in a region around the jet axis. The reconstruction is stopped after an adequate number of hits is found along the trajectory.

Two different regional seeding algorithms have been studied. The first one (referred to as “pixel selective seeds”) uses the pixel lines found by the pixel reconstruction which are contained inside a cone of $\Delta R < 0.4$ around the jet direction, and whose extrapolated z-impact point along the beam line is within 1 mm from the primary vertex. The second algorithm (referred to as the “combinatorial seed generator”) uses all combinations of pixel hits which form an angle with respect to the jet direction of $\Delta\phi < 0.2$ and $\Delta\eta < 0.2$, centered in the primary vertex with a tolerance of ± 1 cm. The first method is faster, while the efficiency of the second is comparable with that obtained for offline reconstruction.

15.6.3.1 Refinement of the Jet Direction Using Tracks

Tracks which have been reconstructed within the jet cone are used to refine the jet direction measurement. Due to the coarse granularity of the calorimeter trigger cells, the direction resolution of the Level-1 jets is rather poor. Reduced angular resolution on the jet direction can cause a sign flip of the track impact parameter, deteriorating the performance of the tagger.

The new direction is determined as the P_T-weighted sum of the track directions. Figure 15-67 shows the difference in direction of the jet found at the generator level and of the jet reconstructed at Level-1, HLT and by including the tracks.

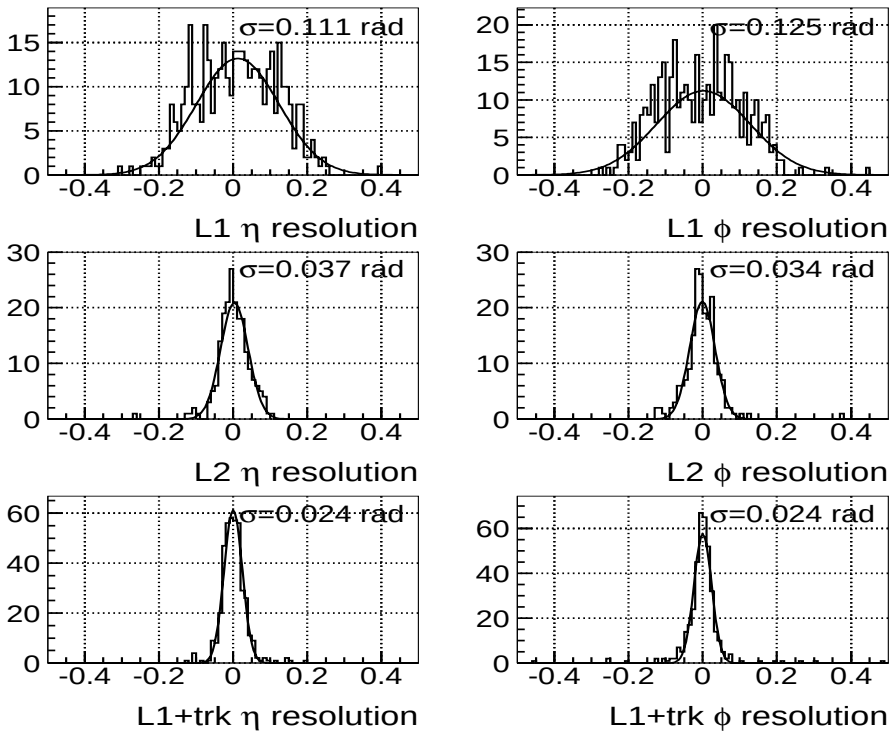


Figure 15-67 Jet angular resolution with respect to the generator information, using Level-1 jet reconstruction, HLT jet reconstruction, and by adding tracking information.

15.6.4 Performance and Timing

In this section HLT selections are given for two samples of events: back-to-back di-jets of different transverse energies and an inclusive QCD sample. The di-jet sample was produced in two different η bins: $|\eta| < 1.4$ and $1.4 < |\eta| < 2.4$, corresponding to the central and forward regions of the tracker. Three bins with $E_T = 50$, 100 and 200 GeV were used: for $E_T = 50$ GeV the track spectrum is softer, with multiple scattering limiting the performance of the tag, while for $E_T = 200$ GeV the performance is limited by the high particle density. In the generation of these events, all the $pp \rightarrow qq$ processes were included, but only events with jets within the η and E_T range in question were selected. The momenta of jets are estimated with the PYCELL routine of PYTHIA, choosing a cell size similar to the one of the CMS calorimeter. For the QCD sample, events generated by PYTHIA 6.152 were retained if $50 < P_T^{hard} < 170$ GeV/c. A total of about 50,000 events in each P_T^{hard} bin were analysed.

The performance of the selection is given in terms of the efficiency to tag *b*-jets against the efficiency for *u*-jets, as well as execution time.

15.6.4.1 Results on Inclusive Jet Samples

Tracks are reconstructed using a regional approach as explained in Section 15.6.3, using both seeding algorithms, with a cut on P_T of 1 GeV/c (2 GeV/c) at low (high) luminosity. The maximum number of hits along the track is 7. The primary vertex is reconstructed using the algorithm described in [15-23], with the only exception being that the default cut on pixel lines is 1 GeV/c (5 GeV/c) for low (high) luminosity.

The *b*-tag algorithm is the Track Counting method described in Section 15.6.1. A jet is tagged as a *b*-jet if it has two tracks exceeding a threshold on impact parameter significance. Tracks are required to have at least three (two) pixel hits for the full (staged) pixel detector configuration.

Figure 15-68 shows the performance of the algorithm for 100 GeV E_T jets, for two different η regions, at low and high luminosity. The difference in performance at low and high luminosity is due to the different cut on the track P_T and the increased pixel readout inefficiency at high luminosity [15-24].

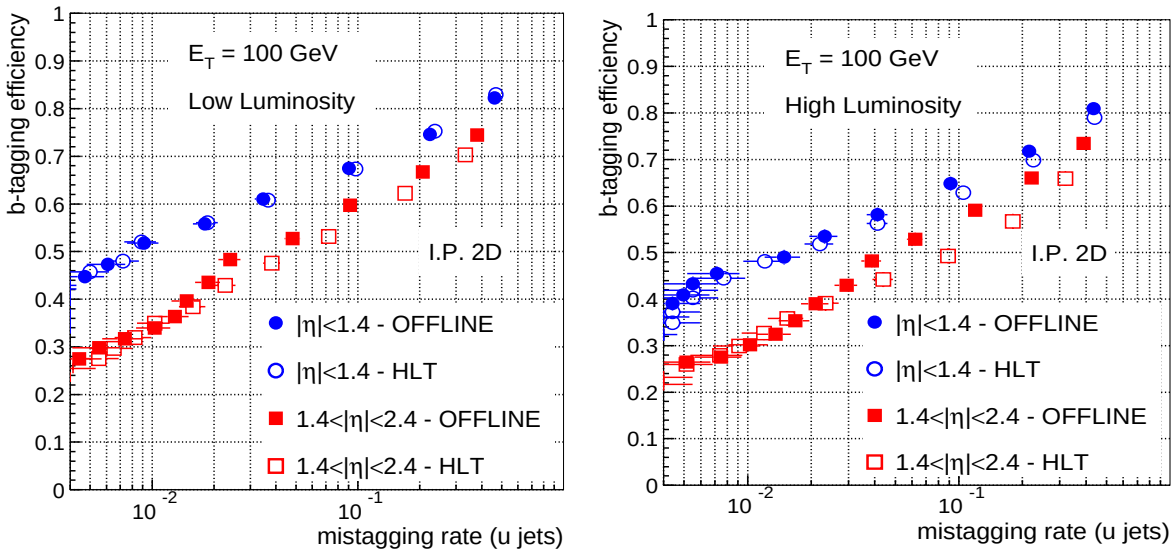


Figure 15-68 Efficiency for the *b*-tag versus mistagging rate for jet with E_T=100 GeV in the low (left) and high (right) luminosity scenarios.

The performance is also compared with the one which could be obtained by using fully reconstructed tracks. No degradation is visible, so the online selection does not result in a reduced performance. A comparison of the performance with the staged and full pixel detector is shown in Figure 15-69.

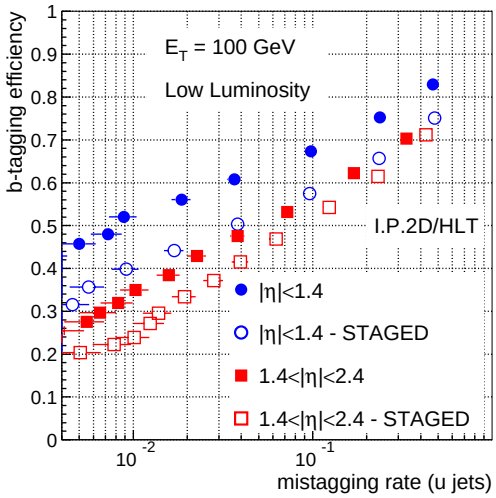


Figure 15-69 *b*-tag performance: comparison of the staged and full pixel detector configurations.

Figure 15-70 (left plot) compares the performance for the different bins of jet E_T. The higher performance of the tagging algorithm for 100 GeV E_T jets is due to the reduction of the multiple scattering component of the impact parameter error. On the othe hand, the 200 GeV jets have higher mistagging rate due to the increased track multiplicity, while those at 50 GeV are limited by multiple scattering. A better performance can be achieved using a three-dimensional impact parameter, as seen by comparing the performance for the 100 GeV E_T jets, also shown in Figure 15-70.

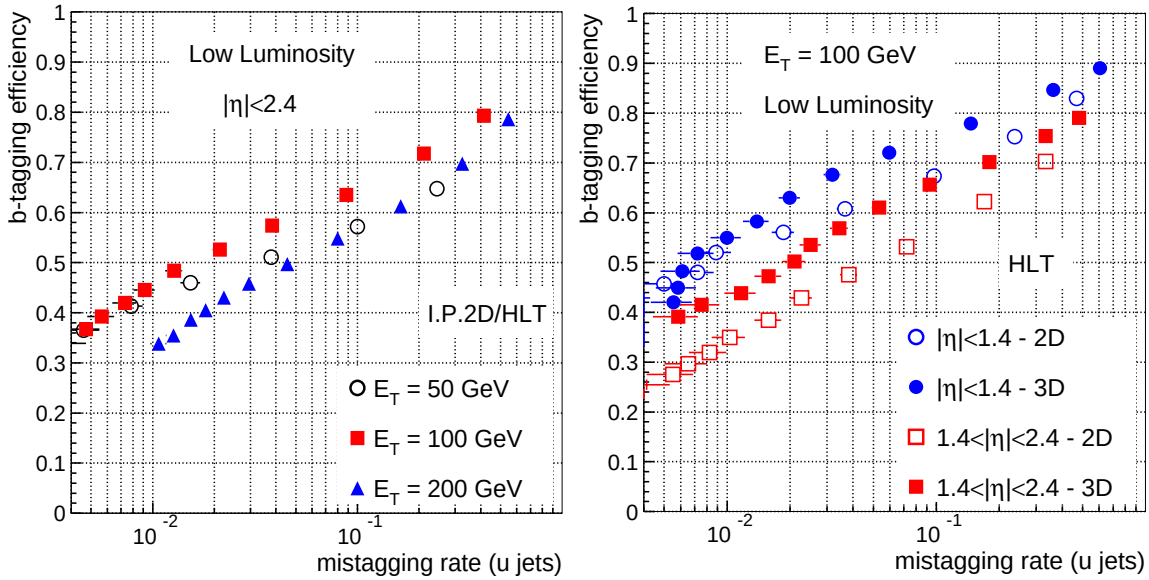


Figure 15-70 Efficiency for *b*-tagging versus the mistagging rate for different jet energies (left) and for 3D vs 2D impact parameters (right).

The impact of this simple *b*-tag algorithm on the inclusive jet rates after Level-1 has been investigated. An inclusive trigger capable of efficiently selecting *b*-jets originating from decays of heavy resonances, such as Higgs bosons, can be implemented by demanding the presence of one or two *b*-tagged jets within the tracker acceptance. We have investigated the performance of a selection based on the leading or the next-to-leading jet which has been tagged as having at least 2 tracks with a transverse impact parameter significance larger than 2. Figure 15-71 shows the rate after this selection as a function of the corrected jet E_T for the first two leading jets in the event.

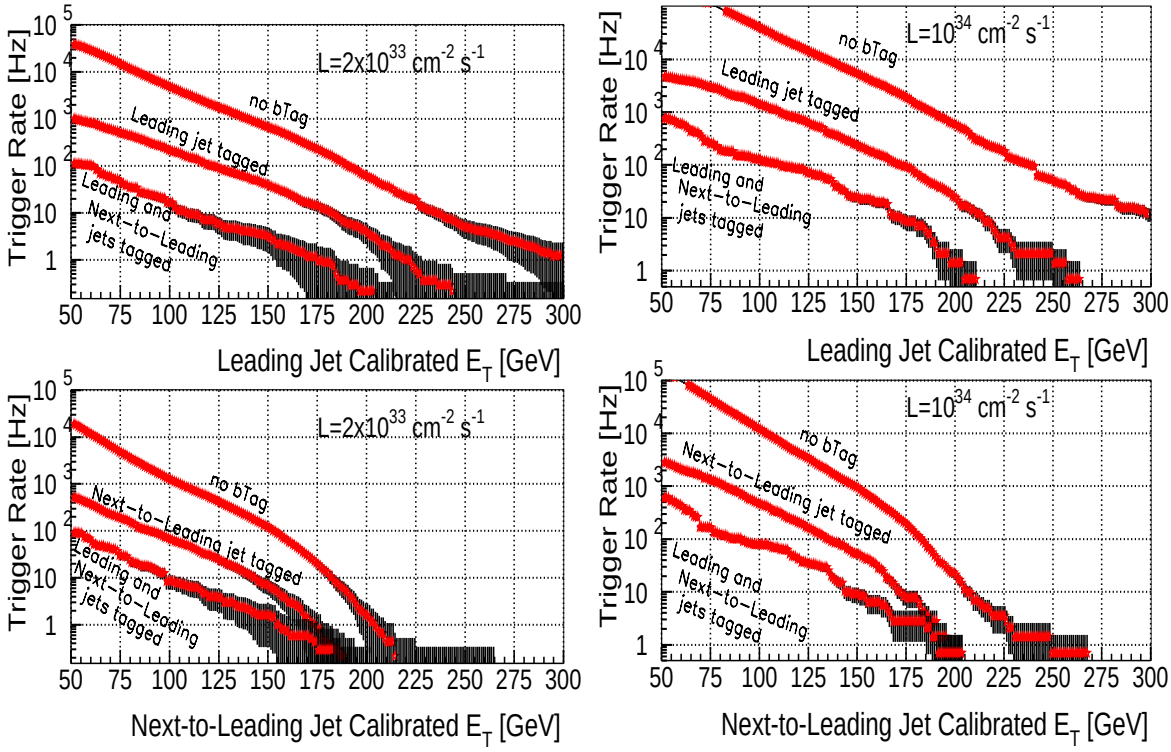


Figure 15-71 Rate after the *b*-tag selections for low (left) and high (right) luminosity for the leading (top) and next-to-leading (below) jets ordered in calibrated E_T, within the tracker acceptance. In each plot the upper curves indicate the trigger rate at Level-1. The middle curves refer to the case when only the leading jet is tagged, while the lower curves refer to the case when both the two leading jets are tagged.

15.6.4.2 Timing

The execution time of the HLT algorithm is measured on a 1 GHz Intel Pentium-III CPU. Figure 15-72 shows the execution time as a function of the number of hits along the track for 100 GeV E_T jets at low luminosity for the pixel selective seeds. The different components are divided in three categories:

- Pixel and primary vertex reconstruction
- Track seeding, building, cleaning and smoothing
- *b*-tag Algorithm.

Most of the time comes from track reconstruction, and scales almost linearly with the number of hits. The algorithmic part is negligible, suggesting that it might be possible to use more sophisticated algorithms for *b*-tagging, with increased performance. At high luminosity, the larger number of seeds increases the execution time drastically. As a comparison, for the same jet transverse energy and pseudorapidity bin,

the number of seeds which need to be considered increases from 7 to about 44 in going from low to high luminosity, even after taking into account the larger P_T cut which is applied at high luminosity. The execution time also depends strongly on the jet energy when using the combinatorial seeding, as shown in Figure 15-73.

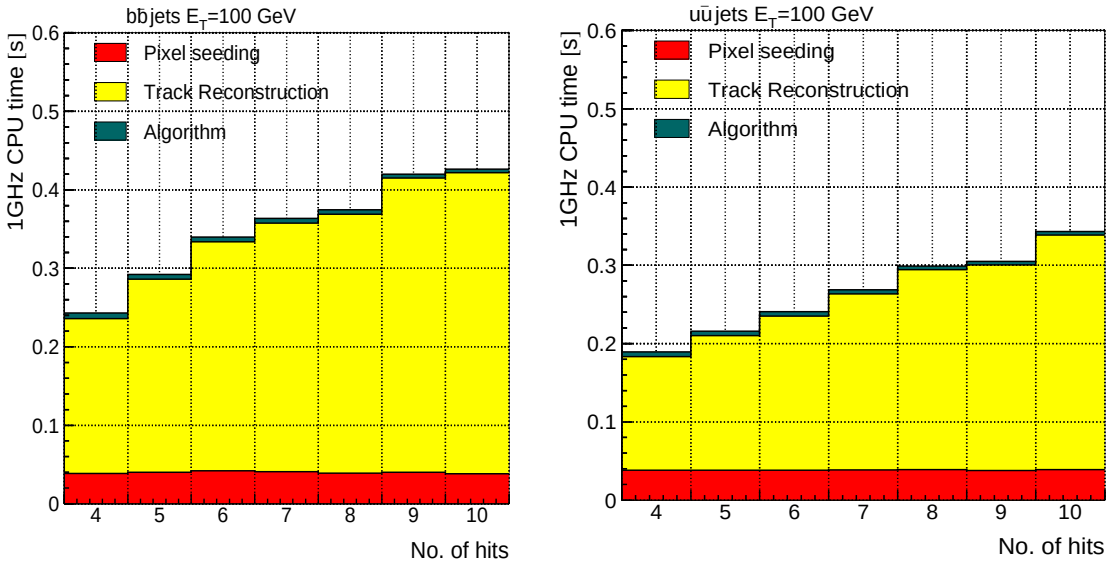


Figure 15-72 Execution time, at low luminosity, as a function of the number of track hits used for 100 GeV di-jet events (left $b\bar{b}$, right $u\bar{u}$). The three components shown with different colours are described in the text.

The execution time is considerably smaller if the pixel selective seeds algorithm is used (Figure 15-74), at the expense of a slightly reduced tagging performance.

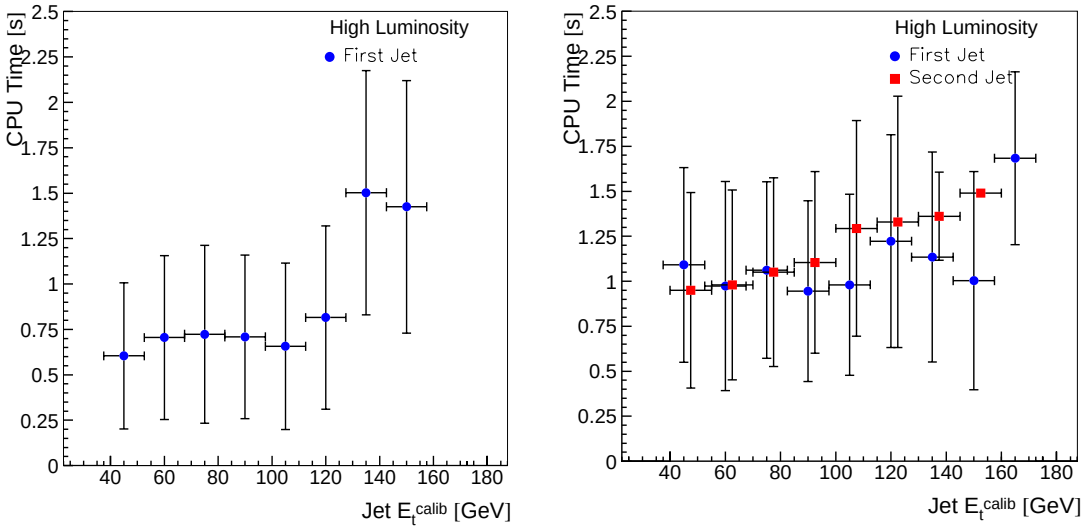


Figure 15-73 Execution time at high luminosity for the regional seeding and combinatorial regional tracking algorithm, for different jet transverse energies. The left plot refers to events with one jet only in the tracker acceptance, while the right plot shows the timing for the leading (first) and the next-to-leading (second) jet in events when two jets are within the tracker acceptance. The error bars represent the spread in the time distribution.

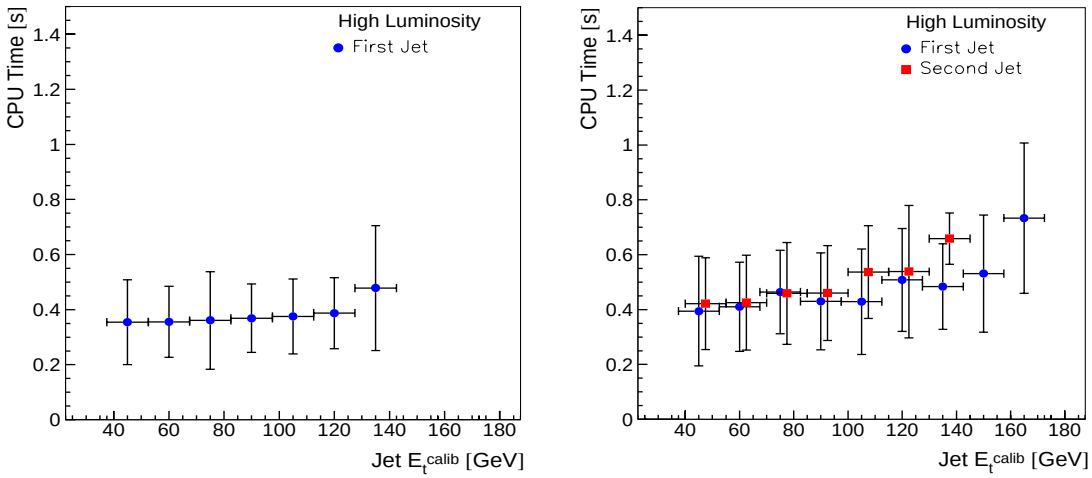


Figure 15-74 Execution time at high luminosity for the pixel lines algorithm, for different jet transverse energies. The first plot refers to events with one jet only in the tracker acceptance, while the plot on the right shows the timing for the leading (first) and the next-to-leading (second) jet in events when two jets are within the tracker acceptance. The error bars represent the spread in the time distribution.

15.6.5 Summary

The High-Level Trigger can benefit from efficient *b*-jet tagging at both low and high luminosity, with execution times which can be sustained in the processor farm running just after Level-1. Comparison with offline tagging performance shows that no information is lost in this online tagging.

A possible dedicated *b*-tagged HLT stream with at least one jet inside the tracker acceptance and a *b*-tag based on 2D impact parameter significance, S, might have the following cut parameters:

- For jets E_T < 80 GeV: 2 tracks with S > 1.5;
- For jets 80 < E_T < 150 GeV: 2 tracks with S > 2.0;
- For jets E_T > 150 GeV: 2 tracks with S > 2.5.

Such a selection would allow to have about a 55% efficiency for *b*-jets and a rejection factor of about 10, almost independent of the jet E_T. It would correspond to a rate of 5 Hz for a cut on the leading jet E_T of about 200 GeV or 160 GeV on the next-to-leading jet.

This selection could be used in the case of SUSY searches, where the number of *b*-jets in the final state can be large. In particular it would help in the HLT selection of point 6 and 6R (Section 15.8.3.2) and also in the cascade decays of $\chi_2 \rightarrow \chi_1 h \rightarrow b\bar{b}$ [15-25].

15.7 Calibration and Monitor Samples

15.7.1 Calibration Methods and Samples

15.7.1.1 Tracker Alignment

The first task after the installation of the CMS tracker will be to precisely align all its components in a single reference frame. The intrinsic detector resolution can reach, for the pixels, a value as low as 10 μm. The alignment procedure should assure a level of precision such that the remaining error can be considered negligible with respect to these figures.

It has been shown, in a simple test beam configuration, that the tracker can “autoalign” to the level of a μm, using charged tracks [15-26] and minimizing the track residuals in an iterative procedure. With this precision the aligned tracker performs as a perfectly aligned tracker. A limitation of this method is that tracking must be possible for the iteration to be started; if no tracks are found, there is no quantity that can be minimized. To investigate this limitation numerous studies have been performed to check the track reconstruction efficiency and the rate of fake tracks for large misalignments.

An ideal tracker alignment is simulated in CMSIM. Misalignment is simulated in the reconstruction, where misalignments can be introduced at four different levels, namely at the sub-detector level, at the layer level, at the ring level and at the detector level. The elements at each of these levels can be rotated or translated independently, and the resulting overall alignment is a combination of the misalignments of the various components. The strategy chosen for track-finding in situations of large misalignment is to enlarge the errors assigned to the reconstructed hits, so that the Kalman filter is still able to match the correct hits to be attached to a track. Naturally, this also results in a larger number of fake tracks being reconstructed, but the redundancy of the tracker layout is enough to maintain this rate at a reasonable level.

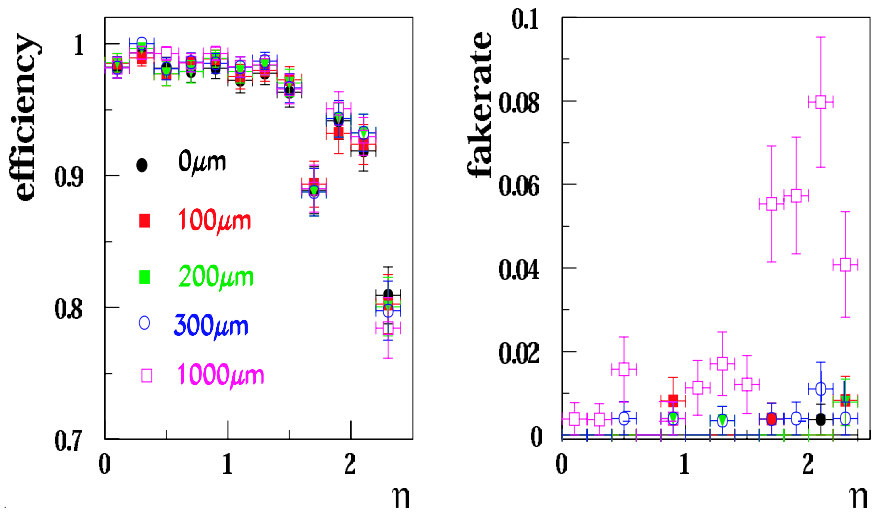


Figure 15-75 Tracking efficiency (left), and fake rate (right) for $W \rightarrow \mu\nu$ events at low luminosity, for misalignments up to 1 mm.

Figure 15-75 shows the tracking efficiency for muons from $W \rightarrow \mu\nu$ as a function of η. The rods and the wedges in the tracker silicon barrel layers are randomly misaligned up to 1 mm. When the reconstruction error is enlarged to take into account the misalignment, no degradation in the efficiency is found. On the

other hand, if the standard reconstruction error alone is used, the tracker efficiency rapidly collapses to almost zero.

The same is also true for random rotations, as shown in Figure 15-76. Both misalignments (translational of 1mm and rotational of 1mrad) are well beyond the tracker construction specifications [15-27].

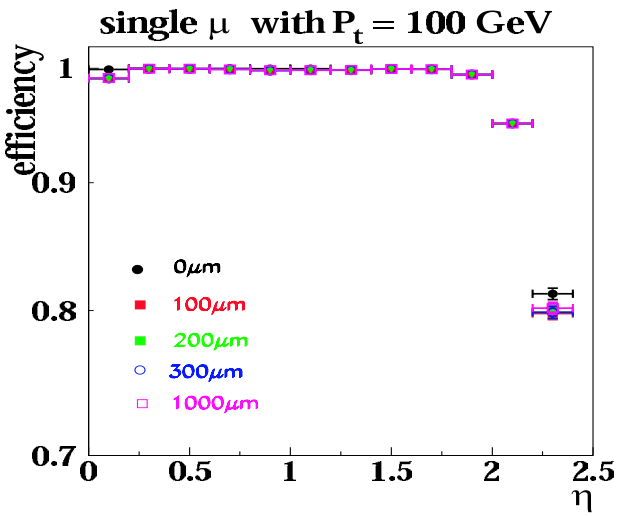


Figure 15-76 Tracking efficiency for single muons with P_T=100GeV/c as a function of η with randomly misaligned rods and wedges up to 1 mrad.

15.7.1.2 ECAL Calibration

Calibration is of key importance for the ECAL. The intercalibration error goes into the energy resolution directly, with very little down-scaling (most of the energy is in a single crystal). What is at stake is intercalibration, not the overall energy scale, since once the channels (crystals) are intercalibrated a handful of $Z \rightarrow ee$ (for example) can be used to set the energy scale.

The standard ECAL calibration will be done using electrons from *W*-decays taken with the single electron trigger. The tracker momentum measurement will be used to intercalibrate the crystals. This should provide a complete intercalibration about every two months. The target is an intercalibration precision of better than 0.5%.

Two issues dominate the $W \rightarrow e\nu$ and $Z \rightarrow ee$ calibration studies:

- Extracting the constants from electron showers whose reconstruction uses ~25 crystals. For this a matrix inversion or iterative method is required. In the case of $Z \rightarrow ee$, where a calibration with m_Z is attempted, the problem is further complicated by the fact that the result (a measured mass) contains no information about the individual energies, but only about their product. However, in both cases, we have candidate iterative methods which perform well in simplified toy/trial setups.
- The trade-off between electron efficiency and the inclusion of electrons with large losses due to bremsstrahlung radiation in the tracker material. The bremsstrahlung loss gives a tail in the E/p distribution, but if all electrons are included there is so much loss that the peak is also shifted. This peak shift is η dependant because of the large variation of tracker material as a function of η. Electrons which radiate significantly can be removed using cuts on the shower shape in the ECAL, or on track parameters, but to get rid of the tails completely very hard cuts are required, with severe consequences for efficiency and hence for the time required to collect adequate statistics for cali-

bration. This leads to the strategy (for $W \rightarrow e\nu$) of intercalibating individual crystals within a small region (in which the variation of tracker material is small) using loose cuts; and then intercalibrating these small regions with much tighter cuts.

Short-term changes in crystal transparency due to radiation damage and recovery will be continuously monitored using the ECAL laser monitoring system. Light transported to the crystals through optical fibres, is injected, and the resulting signal pulses read out, during the 'abort gaps' in the LHC bunch structure. The changes in crystal transparency occur rapidly (~hours) and saturate at a level dependant upon the dose rate. In the absence of radiation there is a recovery which takes place with a time constant of several days. The magnitude of the changes in transparency expected depends on dose rate. In the barrel, at low luminosity, the full extent of the damage should be less than 1%. At $|\eta| = 2.5$ it will be about 5%. At high luminosity the corresponding figures are expected to be 5% and 15%.

The laser system will also be used to transport the test beam 'pre-calibration' of a significant number of super-modules (~10) and one endcap Dee to the experiment. The target is a transported intercalibration precision of better than 2% for those units which are exposed to the beam.

In addition to these standard tools, two others have been studied. Rings of crystals at fixed η can be intercalibrated by imposing ϕ-symmetry on the recorded energy in them. By taking a dedicated stream of special triggers at high rate (~1kHz) and, in the HLT farm, stripping out the handful of crystals in each event having energy above some threshold, an intercalibration at the level of ~2% can be obtained very rapidly. This has been investigated in detail for minimum-bias crossings [15-28], and is now being studied for the case of high E_T single jet triggers. The use of jet triggers would enable the calibration to be done with much larger energy deposits than are used with minimum-bias crossings. But work needs to be done in this case to avoid any trigger bias. The ϕ-symmetry technique may be of particular importance at start-up, in order to rapidly achieve an acceptable intercalibration which yet falls short of the ultimate precision aimed for. The rings in ϕ would have to be intercalibrated using another method. For this purpose, use of *Z*-decays to two electrons is being studied. These events would come through the standard two electron trigger. Preliminary results suggest that the ϕ-rings could be intercalibrated in a few days using *Z*-decays, even without the use of track information.

15.7.1.3 Calibration Samples for the HCAL

The HCAL has five distinct and complementary systems for calibration and monitoring [15-29]:

1. megatile scanner: a collimated gamma source to measure light yield of each tile of every megatile,
2. moving wire source, with gamma source tube(s) installed in every megatile that cross every tile
3. light injection: UV laser and blue-LED light injection into the readout box containing the HPDs,
4. test-beam calibration: four individual wedges with muons, electrons and hadrons are calibrated against the moving wire source for absolute energy calibration scale,
5. physics events in the CMS detector including jets and muons.

The megatile scanner is used to measure the relative light yield of each tile to better than 1% during the construction of the megatiles. The moving wire source will track the changes of each tile to better than 0.7% throughout the life of the HCAL. In the readout box, UV laser and blue LED light will be used for the timing of individual towers and for the measurement of the linearity of the readout electronics. The radioactive source will transfer an absolute calibration, obtained with test beams of single-hadrons, to the calorimeter. Test-beam and physics events will provide the ultimate jet energy calibration.

The source calibration with the megatile scanner and the moving wire source is the primary tool for the calibration of the relative gain of each readout channel in the HCAL. To complement the source calibration, events taken with a minimum-bias trigger will be used to calibrate each readout channel. Such events are taken into the HLT at ~1 kHz and the calorimeter response is stored in the form of histograms, with one histogram for each channel. The histograms will be shipped to a data-logger periodically.

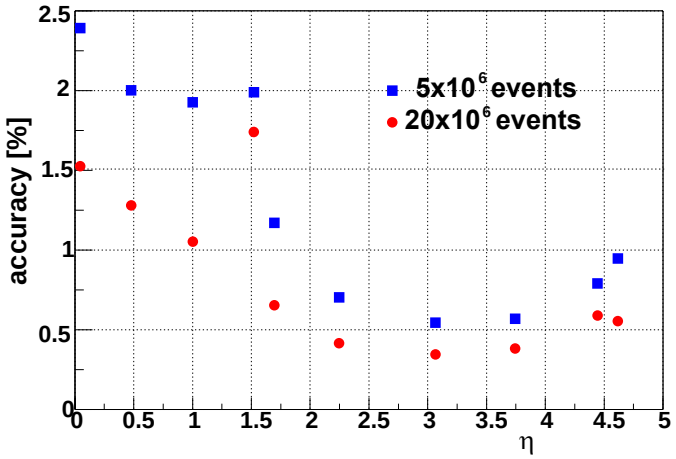


Figure 15-77 Calibration accuracy from online minimum-bias events.

Figure 15-77 shows the accuracy of such calibration when 5 million, and 20 million, minimum-bias events are used, which corresponds to 1.4 hours and 5.6 hours of data taking at low luminosity. The accuracy will be about as good as that of the source calibration. This calibration method also provides additional information that can be used to help correct for non-uniformity of the calorimeter response due dead material before HCAL.

In addition to minimum-bias events, several types of events taken during normal running will be used to calibrate the CMS calorimeter and the jet energy scale [15-29]. Because the CMS calorimeter system is not compensating, its response to hadronic particles is non-linear. Combined with the effects of the strong magnetic field, pile-up from superimposed interactions, noise and threshold effects as well as calorimeter non-uniformity over pseudorapidity, the reconstructed jet energy has a non-linear and eta dependent response to the particle level jet energy. The goal of a jet energy calibration is to determine this response function and to correct the reconstructed jet energy, on average, back to the particle level jet energy. For this purpose it is envisaged to use isolated charged particles, jet balancing events (γ-jet, Z^0-jet, and jet-jet), and the di-jet mass from *W*s in top decays. These samples will be available using the standard thresholds in the Level-1 Trigger and HLT, with the exception of the γ-jet sample which requires a dedicated HLT algorithm as described below.

We select events with a single jet recoiling against a photon by suppressing initial and final state radiation with a cut on the azimuthal angle between the photon and the jet and with a veto on additional jets above some E_T threshold. It has been demonstrated in detailed simulations that in such events the particle level jet transverse energy, E_T(genjet), can be approximated by the reconstructed photon transverse energy, $E_T(\gamma)$, and the jet energy scale can be obtained by taking the maximum of the distribution E_T(recjet)/$E_T(\gamma)$ as a function of the photon (i.e. particle jet) transverse energy and the reconstructed jet pseudorapidity. The major background to direct photon events comes from QCD di-jet events, when one of the jets produces an isolated electromagnetic cluster in the calorimeter. In this case the cluster contains only part of the parton energy and thus these events lead to a systematic shift in the measured energy scale.

Since the CMS single photon HLT selection envisages storing only events with $E_T(\gamma) > 80$ GeV, we need a dedicated calibration selection for lower energies. The production cross section of photon+jet events is large. Making strong requirements at the trigger level on the event selection has a large advantage over simple rate prescaling because the number of signal events contained in the sample eventually usable for the offline analysis is larger. We expect a 60 Hz rate of single isolated electron-photons with $E_T(\gamma)$ >30 GeV from signal events even after strong event selection cuts with low signal efficiency but with very high background rejection.

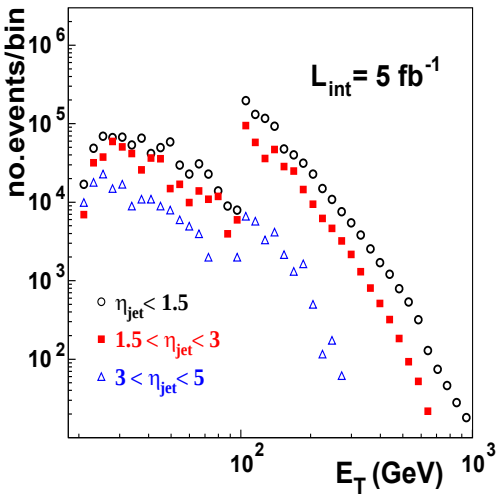


Figure 15-78 Number of calibration events versus photon E_T (the bins have a width approximately equal to 10% of the photon E_T) after 3 month low luminosity data taking for three jet pseudorapidity regions as a function of $E_T(\gamma)$ threshold. A prescaled dedicated calibration trigger below 100 GeV and the single isolated γ trigger above 100 GeV are used.

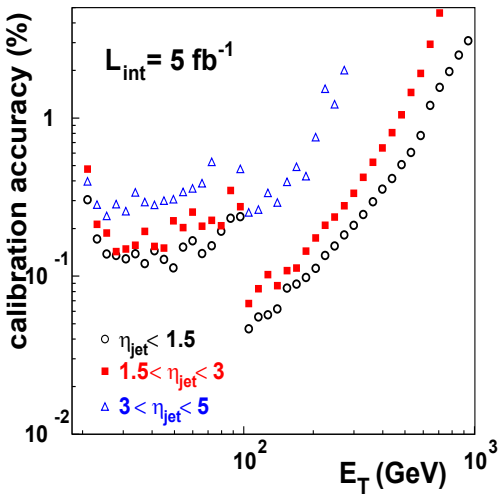


Figure 15-79 Statistical errors on the jet energy scale calibration assuming the fit is done using bins with a width approximately equal to 10% of the jet E_T after 3 month low luminosity data.

The calibration trigger will select isolated HLT photon candidates and will then apply additional isolation cuts. Offline, we will make various selections, based on an azimuthal angle cut and veto on additional jets, as part of our systematic studies. In what follows, we assume the efficiency of the final offline selections is not less than 50%. Our current analysis uses the following photon isolation cuts for the calibration trigger at low luminosity: the transverse energy reconstructed in ECAL between cones 0.07 and 0.5 around the photon candidate passes

$$\sum_{0.07 < \Delta R < 0.5} E_T < 0.005 E_T^{1.5}(\gamma)$$

where only crystals with E_T >0.1 GeV are used, and no pixel line from the signal vertex is allowed in a cone of radius 0.7 around the direction of the photon candidate. The signal vertex is reconstructed in the pixel detector as the vertex with a highest sum of pixel line transverse momenta. With these cuts we still expect total event rate of 25 Hz for $E_T(\gamma)$>20 GeV which will be prescaled to a value consistent with the overall resources for output to storage. Figure 15-78 shows the estimated number of events versus the E_T of the photon after 3 months (2.5×10^6 s) of low luminosity data taking assuming a 1 Hz trigger rate is available for calibration. The energy dependence of the prescale was set so that the statistical error on the calibration was approximately constant over the range 20 GeV < E_T(jet) < 100 GeV. With this prescale, we achieve a statistical error of less than 1% for the jets calibrated on the prescaled trigger. For jets that are calibrated with the unprescaled trigger ($E_T(\gamma)$ > 80 GeV) the uncertainty will be smaller. Although the

unprescaled single photon trigger starts at 80 GeV, for demonstration purposes we assumed a fully efficient trigger only above $E_T(\gamma) > 100$ GeV. The estimated statistical errors on the energy scale measured in 10% wide $E_T(\gamma)$ bins are plotted in Figure 15-79. The calibration trigger can be implemented on top of the isolated photon HLT trigger, as a prescaled extension of the latter down to $E_T(\gamma) = 20$ GeV.

15.7.1.4 Muon Calibration and Alignment

15.7.1.4.1 Calibration

The front-end electronics of the three muon systems (DT, CSC, and RPC) must be pulsed periodically in order to find dead channels, scan and adjust discriminator thresholds, perform t_0 measurements (DT), measure timing windows (RPC), measure charge to ADC-count conversion and linearity (CSC), and measure cross-talk (CSC). Additionally, test patterns will be injected in order to test the Level-1 Trigger logic of all three systems. This pulsing can take place during periods without beams, or during the “abort gaps” in the LHC bunch structure. Additionally, noise measurements will be performed during periods without beam collisions in order to find noisy (high-rate) channels.

The calibrations are not expected to significantly add to the data volume generated during normal running. For example, in pulsing the cathode front-end boards of the CSC system, the data volume per pulse (where a single channel of every front-end board is pulsed in parallel and all 96 channels of every board are read out) is expected to be about 3.5 MB. A full CSC calibration is expected to take 10–20 minutes and generate approximately 300 GB of data.

15.7.1.4.2 Alignment

Misalignment of the muon chambers or the iron disks and wheels they are mounted on can change the measured sagitta of a muon track, thus distorting the momentum measurement. To illustrate this effect, Figure 15-80a shows the effect of alternately rotating the barrel rings MB1–MB4 by ±0.5 mrad in ϕ (simulating chamber positioning errors of 2–4 mm) on the muon Level-2 turn-on efficiency for a nominal P_T threshold of 40 GeV/c (defined at 90% of the plateau efficiency). The turn-on curve becomes noticeably shallower than the curve for perfect alignment. The effect is smaller for lower thresholds, and the increase of the Level-2 trigger rate from such misalignment is less than 5% for thresholds below about 40 GeV/c. The impact on the Level-3 P_T measurement and turn-on efficiency from muon chamber misalignment is negligible, however, as shown in Figure 15-80b, since the momentum measurement is dominated by the tracker measurements.

The alignment of the muon chambers is expected to be determined using an optical alignment system, as discussed in [15-30]. It is possible to perform an *in situ* alignment using high-P_T collision muons [15-31], but uncertainty on the magnetic field in the yoke at the level of only 0.1% implies that it may take several months of data collection at full luminosity to reach a precision of 200 μm, which is the position resolution of a single measurement layer in the DT and CSC muon systems.

In the endcaps, it may be possible to use beam halo muons, travelling roughly parallel to the beam axis, to align the chambers on the iron disks. The Global Level-1 Trigger will implement a dedicated trigger for such muons, which are identified by logic in the Level-1 CSC Track-Finder. Detailed studies still need to be performed to determine the achieveable alignment precision.

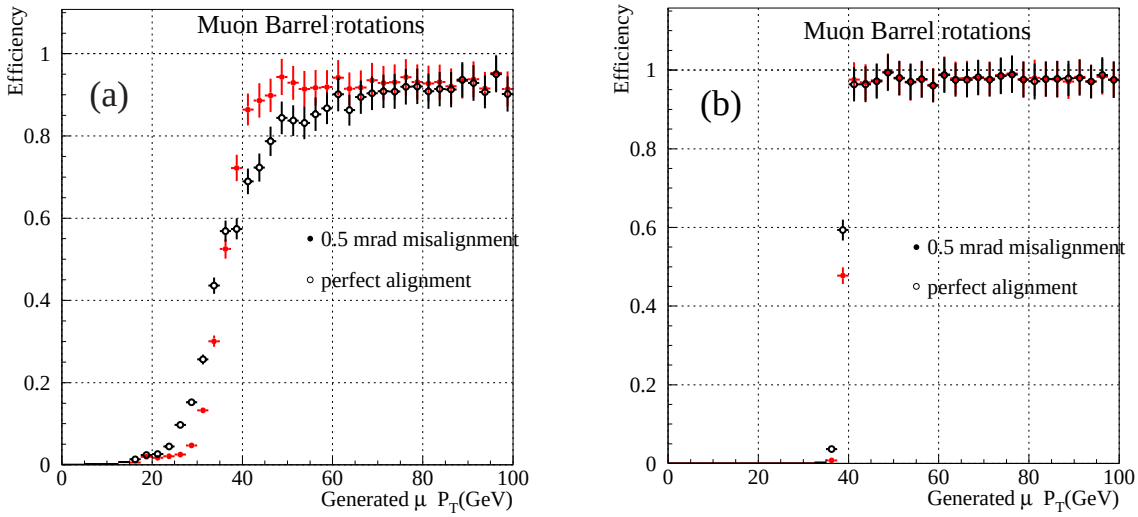


Figure 15-80 Efficiency for single muons to pass (a) the Level-2 and (b) the Level-3 muon trigger as a function of the generated P_T for a P_T threshold of 40 GeV/c, with and without an alternating 0.5 mrad rotation of the barrel rings.

15.7.2 HLT Monitoring

A crucial aspect of the filtering process is the monitoring of the performance of the HLT and its proper functioning. Both tasks can be achieved by analyzing online, and in some cases directing to storage, events that fail the HLT selection. For this purpose at least 1 Hz of events that have failed the HLT will be written to storage. These events will be analysed in near real-time, to identify rapidly any malfunctions in the algorithms or the processors in the Filter Farm.

The online analysis will also be based on statistics accumulated by the HLT process in each Filter Unit. The latter will report key performance parameters, like the fraction of events accepted on each trigger path, at regular intervals. Close monitoring of such quantities, as well as histogramming of select kinematic quantities and topological properties of the objects selected, will be used to monitor the correct functioning of the HLT. More details on monitoring the Filter Farm and the High-Level Trigger can be found in Chapter 16, "Computing Services".

15.8 HLT Performance

This section summarizes the physics object selection, the total estimate for the CPU requirements and the physics performance of the HLT trigger table for the start-up luminosity of 2×10^{33} $cm^{-2}s^{-1}$.

15.8.1 Summary of HLT Selection

The current set of thresholds and rates to storage for each physics object, described in the preceding sections, is listed in Table 15-24.

Table 15-24 High-Level Trigger requirements at low luminosity. The thresholds correspond to the values in E_T or P_T with 95% efficiency (90% efficiency for muons). There is no actual threshold in the HLT selection for τ-jets, so the threshold shown is that of the corresponding Level-1 Trigger requirement.

Trigger	Threshold (GeV or GeV/c)	Rate (Hz)	Cumulative Rate (Hz)
Inclusive electron	29	33	33
Di-electrons	17	1	34
Inclusive photons	80	4	38
Di-photons	40, 25	5	43
Inclusive muon	19	25	68
Di-muons	7	4	72
Inclusive τ-jets	86	3	75
Di-τ-jets	59	1	76
1-jet * E_T^{miss}	180 * 123	5	81
1-jet OR 3-jets OR 4-jets	657, 247, 113	9	89
Electron * Jet	19 * 45	2	90
Inclusive *b*-jets	237	5	95
Calibration and other events (10%)		10	105
TOTAL			**105**

The values of the thresholds indicated in Table 15-24 are indicative of the type of event mixture that an output event rate of $O(10^2)$ Hz would yield. The thresholds are optimized to guarantee high efficiency for given the maximum Level-1 Trigger rate and the are both observed. The latter is the subject of a following section.

15.8.2 CPU Requirement

A key issue for the High-Level Trigger selection is the CPU power required for the execution of the algorithms described thus far. The algorithms were timed, as described in the preceding sections, on a Pentium-III 1 GHz processor, and the current requirements vary from a very fast ~50 ms for jet reconstruction to the longer ~700 ms for muon reconstruction.

The first step is to weight the CPU needs of the algorithms by the frequency of their application, which is the Level-1 Trigger rate of the corresponding channels. This is shown in Table 15-25 and yields a total of 4092 CPU seconds as the total need to cover the 15.1 kHz of events allocated in the Level-1 Trigger table[1]. This translates to a mean of 271 ms per event that passes the Level-1 Trigger.

Table 15-25 Summary of CPU time required for the selection of each physics objects in the HLT. The CPU figures refer to a 1 GHz Intel Pentium-III CPU.

Physics Object	CPU time per Level-1 event (ms)	Level-1 Trigger rate (kHz)	Total CPU time (s)
Electrons/photons	160	4.3	688
Muons	710	3.6	2556
Taus	130	3.0	390
Jets and E_T^{miss}	50	3.4	170
Electron + Jet	165	0.8	132
B-jets	300	0.5	150

The second step is to translate this measurement to a standard measure of CPU power. To connect the current measurement to earlier estimates of the CPU requirements, the SpecInt95 (SI95) benchmarking unit was chosen. The Pentium-III 1 GHz CPU capacity is estimated to be 41 SI95. Using these figures, the total CPU power required to process the maximum Level-1 Trigger rate of 100 kHz by the HLT farm is 1.2×10^6 SI95. This is the figure quoted in Chapter 2, “Overview”.

What is more relevant, however, is the estimate of the CPU power needed to carry out the physics program at the start-up of the LHC, when the luminosity will not have reached its full value. The current scenario for CMS is to have a DAQ system capable of reading a maximum of 50 kHz of events accepted by the Level-1 Trigger. The CPU requirement for this system is thus one half of the total, i.e. 6×10^5 SI95 or, in terms of CPU “units”, 15,000 CPUs as in those available in a standard commercial Personal Computer (PC).

The third and final step is the extrapolation of these figures to the year 2007. In this step a number of variables should be considered, including changes in the architecture of the basic microprocessors and memory, as well as changes in the software itself. No such detailed analysis has been carried out, in the belief that the uncertainties are too large for such considerations to improve the reliability of the estimate. Instead, the basic thesis of Moore’s Law, i.e. that CPU power increases by a factor two every 1.5 years, is used. Given the time period between now and the start of the first physics run, in 2007, there are three such 1.5 year periods, and thus the CPU units of 2007 should be a factor ~8 more powerful than those of today, thus requiring ~40ms per event. This implies that a total of ~2,000 CPUs are needed for the system at the LHC start-up . This is the basis of the 1,000 dual-CPU PCs used in the cost estimates of Chapter 17.

There are, naturally, large uncertainties in the above estimate, listed in order of importance:

- A major uncertainty is related to all the samples that have not been simulated. Inherent in the above estimate is the assumption that all events in the 50 kHz of events accepted by the Level-1 Trigger will require the full 300 ms. However, only 16 kHz of the total of 50 kHz has really been simulat-

1. In the full Level-1 Trigger rate allocation of Table 15-1 there is an additional 0.9 kHz of minimum bias events that will be used for calibration and monitoring. These events are assumed to require the same amount of CPU as the mean of the 15.1 kHz of events for which CPU time estimates are available.

ed, with the rest of the events being included in the “safety factor” in the allocation of the Level-1 Trigger bandwidth. It is clear that much of this safety factor will be used, since basic processes from the beam in the machine, e.g. beam halo, have not been included. It is expected, however, that such processes will require far less CPU to identify and reject.

- The figure of ~300 ms per event accepted by the Level-1 Trigger does not include all the processing needed to go from the actual raw data to the final decision by the HLT. The overhead from the software framework for accessing the DAQ packets of raw data, and the time needed to unpack the raw data in the form needed by the reconstruction algorithm are not included. These times cannot be measured at present, since the raw data formats are not finalized. First estimates suggest that the CPU requirement can be significant, taking up to one quarter of the total reconstruction time.
- The selection of Table 15-24 is only an example of the type of requirements and rates involved in the HLT selection, and the actual trigger table will very likely include additional selections. In particular, it is expected that more selections involving combined objects will be introduced. This wil increase the required CPU time.
- The time needed to process a single combination depends on the actual implementation of the reconstruction algorithm and the underlying libraries, e.g. the mathematical and memory allocation ones. It also depends on the quality of the compiler. In this respect also the current CPU time estimates are an upper limit. The current time estimates are dominated by the muon reconstruction step which is in turn dominated by the GEANE extrapolation step. A significant speed-up is expected when the extrapolation package is upgraded. Morever, the current package used for matrix algebra will be replaced with a better one, while specialized memory allocators wil be used for cases where the memory management overhead has been measured to be significant. Such modifications require little or no change to the reconstruction source code and can be done at a later stage without compromising algorithmic correctness.
- There are uncertainties arising from the actual occupancy in the CMS sub-detectors. The time needed to perform a given pattern recognition and/or reconstruction task depends on the number of elements or combinations that need to be considered in order to reach a given reconstruction efficiency, as well as on the time needed to treat each element. The minimum number of combinations needed to achieve a given efficiency depends on the occupancy of the detectors, and on the accuracy of each measurement. The occupancy, in turn, has uncertainties arising from the event generators description of the hard interaction, the noise level in the detector, including the full readout chain, the neutron background in the detector, and finally from beam backgrounds. The accuracy of each measurement has an uncertainty related to the current understanding of the detector and accuracy of the detector response simulation knowledge of the calibration and alignment at HLT time. All these uncertainties have reasonable estimates or are expected to be sufficiently small to be neglected in the present studies.

The CMS HLT reconstruction algorithms underwent rapid development in the context of the DAQ TDR, and this development will continue after the publication of the present document. The CPU time measurements given in this document reflect the current state of the algorithms. A reliable estimate of the CPU power needed for the first physics run in 2007 is difficult to make. There are significant uncertainties in the estimates, as explained above, and for this reason no attempt is made to estimate the potential speed-ups mentioned. Instead, such CPU savings are considered to be part of the “contingency” on CPU, to be used on all the processing steps that will require additional processing, as described above.

It is expected that the CMS reconstruction software will continue to evolve considerably between now and the first physics run. The HLT filtering step should benefit from all improvements, since the code used in the HLT, with the exception of the input and output mechanisms, is identical to the code to be used offline.

15.8.3 Efficiency of HLT Selection for Major Physics Channels

15.8.3.1 Higgs Physics

The search for the Higgs boson is one of the most important tasks of experiments at the LHC. This section summarizes the studies of the Higgs signals, described in Section 15.2 and in Section 15.3 for the Standard Model (SM) Higgs and in Section 15.5 for some MSSM Higgs channels. Specific channels which involve peculiarities in the final state, e.g. an "invisible Higgs boson", are discussed in other sections below.

The mass range to discover the Standard Model Higgs at the LHC ranges from the present negative search limit of LEP 114.4 GeV/c^2 to approximately 1 TeV/c^2. In the heavy mass range the channels with the best sensitivity for Higgs masses up to 800 GeV/c^2 is $H\rightarrow ZZ$. In particular $H\rightarrow ZZ\rightarrow\mu\mu\mu\mu$ will be the "gold-plated" channel above M_H=180 GeV/c^2. For lower values of the Higgs mass the decay channels $H\rightarrow WW^{(*)}\rightarrow ll\nu\nu$, $qqH\rightarrow qqWW^{(*)}$, $qqH\rightarrow qq\tau\tau$, $t\bar{t}H\rightarrow tt\gamma\gamma$, $t\bar{t}H\rightarrow ttbb$, $WH\rightarrow W\gamma\gamma$ and $H\rightarrow\gamma\gamma$ can also be used to discover and study the Higgs.

For the region around 120 GeV/c^2 the most promising channels to date are the fusion processes or the decays into two photons. The decay $H\rightarrow\gamma\gamma$ will be triggered at the HLT requiring two photons detected in the ECAL and validated by the tracker, using the asymmetric cuts of E_T > 40 and 25 GeV for the photons. This cut is equal to the proposed final offline selection for this channel. It yields a final background rate to tape of 5 Hz. For low-luminosity running the combined Level-1/HLT trigger efficiency for a Higgs with mass of 115 GeV/c^2, amounts to 77% for all those decays where both photons are within the ECAL fiducial volume $|\eta| < 2.5$. With respect to all events where the decay photons have the offline minimum E_T, the efficiency is 83.7%. The geometrical acceptance for the two photons to be within the fiducial volume is 65%. The trigger efficiency for high luminosity running is typically 5% lower.

The channel $H\rightarrow WW^{(*)}\rightarrow\mu\mu\nu\nu$ has been studied for Higgs mass values of 120, 160 and 200 GeV/c^2. The combined low-luminosity HLT trigger consists of a single muon P_T threshold of 19 GeV/c and a symmetric double muon threshold of 7 GeV/c and has a total output event rate of 29 Hz. The total efficiency within the fiducial acceptance is 92% at M_H = 160 GeV/c^2. The efficiency increases with increasing Higgs mass. The efficiency for this trigger reduces by 10-15% at high luminosity, because the thresholds have to be increased to 31 and 10 GeV/c for the single and double muon trigger respectively.

The golden channel mentioned above $H\rightarrow ZZ\rightarrow 4\mu$ has near-full efficiency since the efficiency for e.g. $Z\rightarrow\mu\mu$ is about 92%. These efficiencies are relative to having at least one muon inside the geometric acceptance of the trigger, $|\eta| < 2.1$, and all final state muons (two or four, depending on the channel) inside the full acceptance of the muon system: $|\eta| < 2.4$.

These examples show that working HLT trigger schemes with good efficiency have been identified for SM Higgs particles, while in general, a heavy Standard-Model Higgs particle will be easier to trigger on than a light one.

For MSSM neutral high-mass Higgs particles, *A/H,* and the charged Higgs the "golden" decays to heavy gauge bosons are closed or strongly suppressed. Therefore alternative decay channels have been studied, in particular the decay modes $A/H\rightarrow\tau\tau$ and $H^+\rightarrow\tau^+\nu$. The main background to these channels are QCD jets. The τ-jets are identified as described in Section 15.5. The identification can be performed by calorimeter tau jet candidates supplemented either by a fast track finding algorithm in the pixel detectors or by using regional reconstructed track finding algorithm. For the low luminosity runs the latter algorithm is found to be more efficient, giving an efficiency of about 45% (49%) for M_H = 200 GeV/c^2 (M_H = 500 GeV/c^2), at the same time giving the required background reduction of a factor 1000. Both

τ-jets are required to pass the trigger criteria. The corresponding efficiencies for high luminosity running are about 15% lower.

A similar trigger can be devised for the charged heavy Higgs decay $H^+ \rightarrow \tau^+\nu$, requiring one τ-jet to be identified and missing E_T at the HLT. The efficiencies are 58% (M_H = 200-400 GeV/c^2) for the low-luminosity runs and are about 10% lower at high luminosity.

For the neutral MSSM Higgs the combined electron/τ-jet trigger channels have been studied for the decay $A/H \rightarrow \tau\tau \rightarrow e+\tau$-jet. The combined trigger has a reduced threshold on the electron, compared to the single-electron trigger and has an HLT efficiency of 70% at both low and high luminosity. The overall efficiency of the combined trigger are only about 2-5% higher than those of the single electron trigger, but this is mostly due to the choice of a relatively high mass for the Higgs. For Higgs masses around 120 GeV/c^2, the fusion channel $qqH \rightarrow qq\tau\tau$ benefits from such a trigger. Similar studies were made for the channel $A/H \rightarrow \tau\tau \rightarrow \mu+\tau$-jet and give an efficiency of 54% (46%) at low (high) luminosity.

15.8.3.2 Supersymmetry Searches

One of the main goals of the LHC is to look for evidence for Supersymmetry (SUSY), perhaps the most popular extension of the Standard Model. If SUSY exists, significant-sized samples of sparticle events should be accumulated shortly after LHC turn-on. However, unless SUSY is discovered at the Tevatron before the LHC satrt-up, the signature of SUSY will not be known in advance.

The most popular SUSY models invoke the conservation of R-parity (R_P), which makes the lightest Supersymmetric particle, a neutral gaugino in most models, stable. In these models, squark and gluino events, which are produced strongly and therefore have very large production cross section, would appear in the detector as events with multiple jets and large E_T^{miss}. More challenging from a trigger point of view are models where the R_P is not conserved and superparticles decay predominantly to jets. In some of these models, the lightest neutral gaugino decays to three jets, and the signature of squark and gluino production could be a large number of jets. There can be some moderate E_T^{miss} due to decays of top quarks, *W* and *Z*-bosons, *b*-quarks and τ-leptons produced during the squark/gluino decays.

Supersymmetry models have a very large number of free parameters and thus a wide variety of possible signatures at the LHC. There have been several studies to find points in the SUSY parameter space that will in some way span the range of predictions that apply to the start of the LHC. Reference [15-32] was used to select a few points to study for this TDR. These points all use the mSUGRA parameterization of the SUSY parameter space. Other parameterizations have not been considered, since the purpose of this study is not to provide an exhaustive study of SUSY but only to provide examples of the type of Level-1 and HLT requirements that are needed to provide good efficiency for SUSY events.

At low luminosity, the greatest challenge comes from the points with the lowest sparticle masses, just above the reach of the Tevatron, because the transverse energies of the jets and E_T^{miss} are smallest. At high luminosity, the challenge is to maximize the acceptance for the highest mass points, since they have the smallest cross section.

Table 15-26 lists the parameters[1], the masses of the sparticles and the cross sections for the points we have studied for this TDR. The SUSY mass spectra and branching ratios were calculated using ISAWIG 1.104 which uses ISAJET 7.51. This information was imported into HERWIG 6.301, which was used to

1. In the mSUGRA parameterization of the SUSY parameter space, m_0 is the common scalar mass at the unification scale, $m_{1/2}$ is the common gaugino mass, A_0 is the trilinear scalar coupling constant, β is the ratio of the vacuum expectation value of the two Higgs doublets, and μ is the higgsinos mixing parameter.

Table 15-26 Parameters used for the generation of the SUSY mSUGRA samples used in this TDR. The values $A_0 = 0$, $\tan\beta = 10$, and $\mu > 0$ were also used.

Point	M_0 (GeV/c^2)	$M_{1/2}$ (GeV/c^2)	σ (pb)	$m(\tilde{g})$ (GeV/c^2)	$m(\tilde{u}_L)$ (GeV/c^2)	$m(\tilde{\chi}_1^0)$ (GeV/c^2)	M_h (GeV/c^2)
4	20	190	181	466	410	70	110
5	150	180	213	447	415	66	110
6	300	150	500	349	406	45	106
7	250	1050	.017	2235	1986	445	122
8	900	930	.022	2032	1962	391	121
9	1500	700	.059	1625	1975	293	120

do the actual event generation. The points were chosen to give a variety of potential SUSY signatures. Point 4[1] has enhanced slepton (especially stau) production. Point 5 is a "typical" SUSY point with squarks lighter than gluinos and therefore lots of E_T^{miss}. Point 6 has gluinos lighter than squarks and thus tends to have more jets and less E_T^{miss} than a typical point. Point 7 has enhanced tau and sneutrino production. Point 8 is characterized by $\tilde{\chi}_2 \to h\tilde{\chi}_1$ which yields an enhanced (*b*)-jet yield compared to Point 7, while the E_T^{miss} is similar to that in Point 7. For point 9, the gluinos are lighter than all squarks, except for the lighter stop, $\tilde{t}_1$, thus allowing the decay $\tilde{g} \to t\tilde{t}_1$ which in fact dominates and therefore events have many jets and smaller E_T^{miss}. While these exact points (points 4, 5, and 6 in particular) may be excluded by LEP Higgs searches, Higgs production and the exact mass for the Higgs in the decays does not play an important role in the observability or the characteristics of these events. The events were generated using ISAJET. The same points are used to simulate SUSY with R-parity violation with $\tilde{\chi}_1^0 \to jjj$.

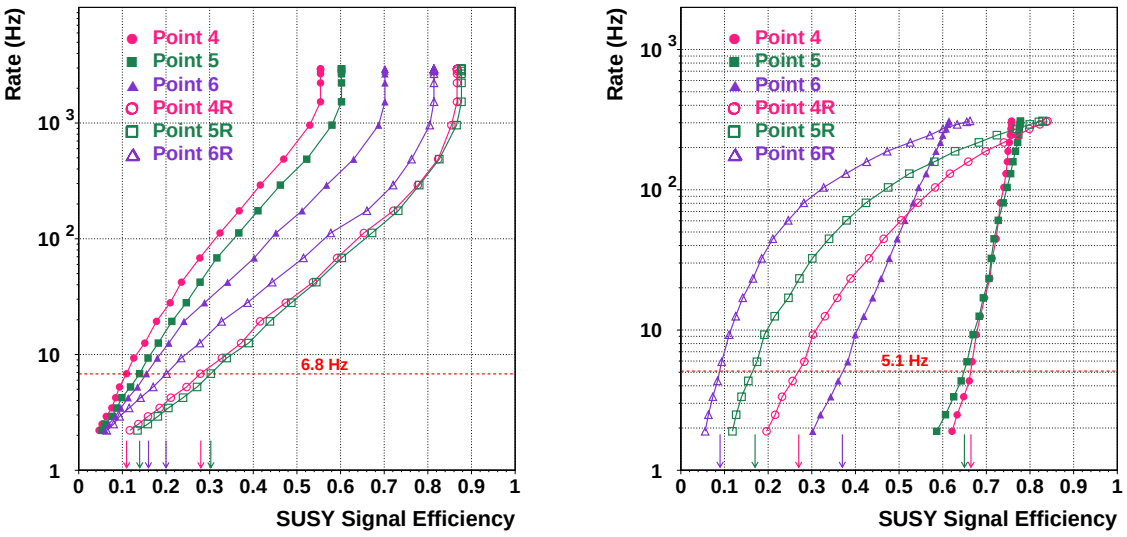


Figure 15-81 HLT rate vs. efficiency for SUSY signals, for events that pass the Level-1 jet+E_T^{miss} Level-1 Trigger. Left: the requirement is for four jets above a threshold which varies and leads to the contours shown, with one contour for each SUSY point. Right: the requirement is for one jet+E_T^{miss} and E_T^{miss} is varied.

1. The labels given to the points (point 4, point 5, etc) are based on a historical convention with no significance here.

For the low-mass points, a few simple triggers with jets and E_T^{miss} were considered. At low luminosity a

Table 15-27 The Level-1 and High-Level Trigger cut values, rates and efficiencies for six Supersymmetry points at low luminosity. The HLT efficiencies are with respect to events that pass the Level-1 Trigger. All thresholds refer to the values with 95% efficiency, with the exception of the Level-1 E_T^{miss} which is the actual cut value. For a definition of the SUSY points, see text.

	Level-1 Trigger		**High-Level Trigger**	
SUSY point	**1 Jet >79 GeV+ E_T^{miss}>46 GeV**	**3 jets, E_T>86 GeV**	**1 Jet >180GeV+ E_T^{miss}>123 GeV**	**4 jets, E_T>113 GeV**
	efficiency (%)	**efficiency (%) (cumulative efficiency)**	**efficiency (%)**	**efficiency (%) (cumulative efficiency)**
4	88	60 (92)	67	11 (69)
5	87	64 (92)	65	14 (68)
6	71	68 (85)	37	16 (44)
4R	67	89 (94)	27	28 (46)
5R	58	90 (93)	17	30 (41)
6R	47	84 (87)	9	20 (26)
Background	**rate (kHz)**	**rate (kHz) (cumulative rate)**	**rate (Hz)**	**rate (Hz) (cumulative rate)**
	2.3	**0.98 (3.1)**	**5.1 Hz**	**6.8 (11.8)**

Table 15-28 The Level-1 and High-Level Trigger cut values, rates and efficiencies for six Supersymmetry points at high luminosity. The HLT efficiencies are with respect to events that pass the Level-1 Trigger. All thresholds refer to the values with 95% efficiency, with the exception of the Level-1 E_T^{miss} which is the actual cut value. For a definition of the SUSY points, see text.

	Level-1 Trigger		**High-Level Trigger**	
SUSY point	**1 Jet >113 GeV+ E_T^{miss}>70 GeV**	**3 jets, E_T>111 GeV**	**E_T^{miss}>239 GeV**	**4 Jets, E_T>185 GeV**
	efficiency (%)	**efficiency (%) (cumulative efficiency)**	**efficiency (%)**	**efficiency (%) (cumulative efficiency)**
7	90	62 (90)	85	18 (85)
8	97	76 (98)	90	28 (92)
9	91	67 (94)	72	28 (76)
7R	91	99 (100)	70	75 (90)
8R	86	100 (100)	58	78 (88)
9R	75	99 (100)	41	52 (64)
Background	**rate (kHz)**	**rate (kHz) (cumulative rate)**	**rate (Hz)**	**rate (Hz) (cumulative rate)**
	4.5	**1.1 (5.4)**	**1.6**	**1.5(3.0)**

3-jet trigger and a jet+E_T^{miss} trigger are considered at Level-1. For the HLT, a jet+E_T^{miss} and a 4-jet channel are considered. Figure 15-81(left) shows the 4-jet trigger rate versus efficiency for events that pass the Level-1 Trigger jet+E_T^{miss} criteria for the six SUSY points as the threshold on the jets is varied. Figure 15-81(right) shows the rate versus efficiency for the jet+E_T^{miss} trigger as the threshold on the E_T^{miss} is varied. The arrows on the plots show the thresholds chosen for the low luminosity trigger table as a compromise between efficiency and bandwidth. The cut values, trigger rates and signal efficiencies are given in Table 15-27. This simple table is sufficient for LHC turn-on. After the first few runs, and once the actual trigger conditions are known, more triggers will be added to increase the efficiency for SUSY

For the high luminosity case, the high mass SUSY points 7, 8, and 9 are considered. It is relatively easy to design highly efficient triggers for these samples. Table 15-28 summarizes the optimized Level-1 and HLT requirements and output rates. The Level-1 Trigger has a single jet+E_T^{miss} and a 3-jet trigger, while the HLT has an E_T^{miss} and a 4-jet trigger. For these high mass points, it is easy to get a very efficient trigger table with only a few trigger terms, even for R-parity violating SUSY.

15.8.3.3 Invisible Higgs

Some extensions of the Standard Model (SM) contain Higgs bosons which can decay into stable neutral weakly interacting particles, therefore giving rise to “invisible” final states. In recent work [15-33][15-34], it was shown that the potential for discovering a Higgs boson decaying invisibly at the LHC can be extended considerably by studying Higgs production via weak boson fusion (WBF). The presence of the two forward-backward tagging jets that accompany Higgs production via weak boson fusion is a powerful tool for separating signal from the very large backgrounds. The large rapidity gap between the two tagging jets is used in the trigger definition. The following offline cuts for an invisible Higgs analysis were proposed in reference [15-33] and checked using a detailed simulation [15-34]:

1. $E_T^j > 40$ GeV, $|\eta_j| < 5.0$, $|\eta_{j1}-\eta_{j2}| > 4.4$, $\eta_{j1}\eta_{j2} < 0$,
2. $E_T^{miss} > 100$ GeV
3. $M_{jj} > 1200$ GeV/c^2
4. $\Delta\phi_{jj} < 1$

The efficiencies of the Level-1 Trigger and the HLT selections discussed below are the efficiencies for events that pass these offline selections.

The HF calorimeters play a crucial role in the selection of the invisible Higgs because of the two forward-backward tagging jets. The relative acceptance in the HF calorimeter ($3.0<|\eta|<5.0$) for these jets after requiring $E_T>30$ GeV for each jet is shown in Table 15-29 before and after the cut on the rapidity gap between the jets. After the rapidity gap constraint, almost 80% of the Higgs events will have at least one tagging jet in the HF.

Table 15-29 Acceptance of the CMS HF calorimeter to tagging jets ($E_T > 30$ GeV) in the $qq \rightarrow qqH$ process.

	no jets in HF	**one jet in HF**	**2 jets in HF**
no cut on $\|\eta_{j1}-\eta_{j2}\|$	49%	45%	6%
$\|\eta_{j1}-\eta_{j2}\|>4.4$	22%	65%	13%

The Level-1 jet trigger covers the entire calorimeter acceptance, including the HF calorimeter. At Level-1 a jet+E_T^{miss} trigger can therefore be used for the invisible Higgs selection. Figure 15-82 shows the transverse energy of the highest E_T jet (left plot) and calorimeter E_T^{miss} (right plot) reconstructed offline and at

Level-1 at L = $10^{34}cm^{-2}s^{-1}$ for Higgs events passing the WBF requirements and for a Higgs mass of 120 GeV/c^2 (which is slightly higher than the current limit of 114.4 GeV/c^2 from LEP-II [15-35]). The Level-1 Trigger was optimized by examining the trigger rate for a single jet plus E_T^{miss} trigger vs. the signal efficiency by changing the E_T^{miss} thresholds for the fixed set of single jet thresholds. Figure 15-83

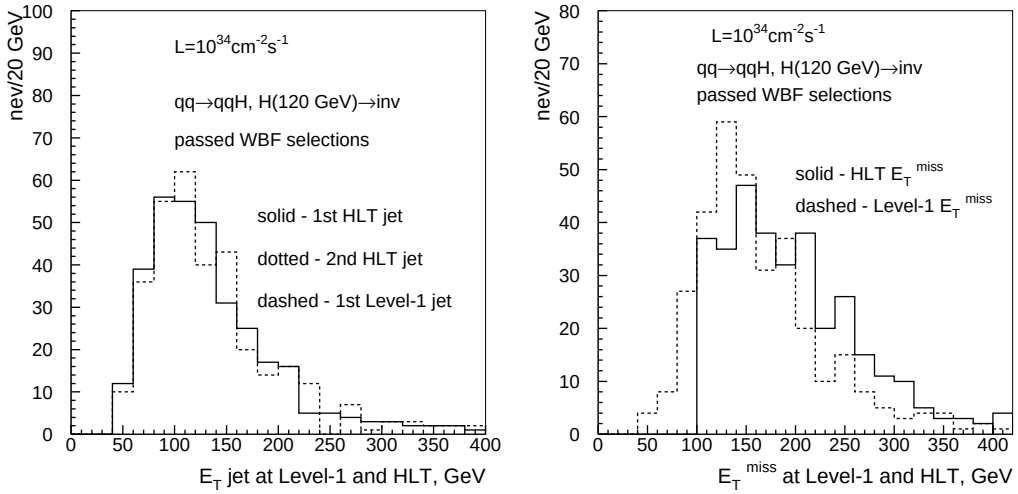


Figure 15-82 Higgs events in the process *qq* → *qqH*, with the Higgs decaying invisibly, M_H = 120 GeV/c^2, and which pass the WBF cuts. Left plot: transverse energy of the highest-E_T jet reconstructed in offline (solid histogram) and at Level-1 (dashed histogram). Right plot: E_T^{miss} reconstructed offline (solid histogram) and at Level-1 (dashed histogram). The luminosity is L=$10^{34}cm^{-2}s^{-1}$.

shows the Level-1 jet+E_T^{miss} trigger rate and Higgs efficiency for single jet thresholds of 70(60), 90(70), 110(80) at high (low) luminosity when the E_T^{miss} threshold is varied.

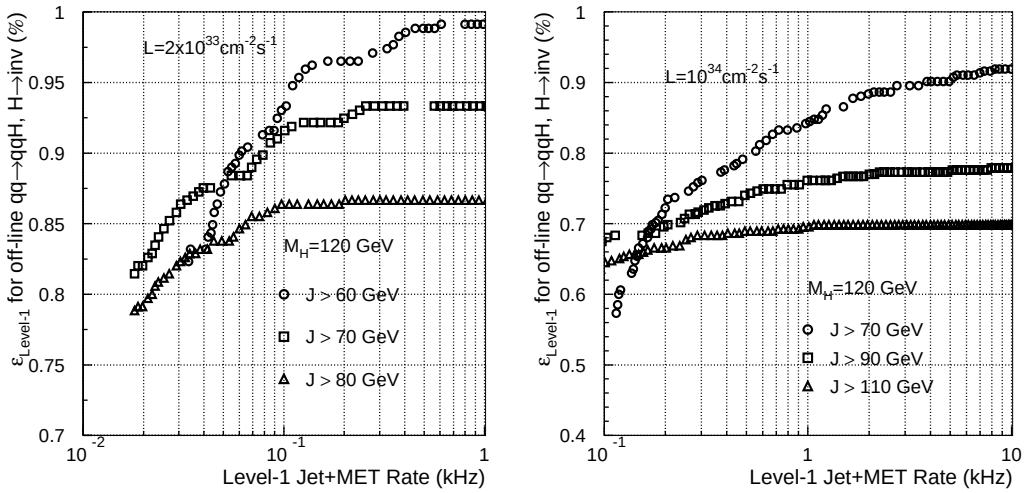


Figure 15-83 QCD 2-jet rate (in kHz) for a jet+E_T^{miss} trigger vs efficiency for *qq* → *qqH*, with the Higgs decaying invisibly, for a Higgs mass of M_H = 120 GeV/c^2, and for events which passed WBF cuts when the E_T^{miss} threshold is varied and with single jet thresholds as labelled. Left (right) plot: low (high) luminosity.

Table 15-30 lists the Level-1 1-jet+E_T^{miss} trigger thresholds and the Higgs efficiency for Level-1 1-jet+E_T^{miss} trigger rates of 0.2, 0.5 and 1.0 kHz at low and high luminosity. A rate of 0.5 kHz gives a Higgs efficiency that is quite high: 98% (80%) for low (high) luminosity.

Table 15-30 Summary of the single-jet plus E_T^{miss} Level-1 Trigger thresholds for a total trigger rate of 0.2, 0.5, and 1.0 kHz at low and high luminosity. Shown are the jet and E_T^{miss} thresholds, the efficiency for $qq \rightarrow qqH$ with the Higgs (with M_H = 120 GeV/c^2) decaying invisibly for events which pass the WBF cuts.

Rate for Level-1 1-jet+E_T^{miss} trigger		0.2 kHz	0.5 kHz	1.0 kHz
Low Luminosity	single-jet threshold, E_T^{miss} threshold (GeV)	60, 73	60, 64	60, 56
	efficiency	0.96	0.98	0.99
High Luminosity	single-jet threshold, E_T^{miss} threshold (GeV)	70, 122	70, 112	70, 103
	efficiency	0.72	0.79	0.84

For the HLT selection, the first of the four sets of offline requirements listed earlier (the cuts on E_T's and η's) are used along with the requirement M_{jj} > 1 TeV/c^2, and a cut on E_T^{miss}. The values for the cuts on E_T and |η| are the same as those for the offline. Figure 15-84 (left plot) displays the rate for QCD multi-jet events as a function of the cutoff on E_T^{miss} with only the first of the four sets of offline requirements applied, and with the first and the M_{jj} cut applied. A rate of 0.1 Hz can be reached at low luminosity for E_T^{miss} >110 GeV without the M_{jj} cut. The signal efficiency in this case is near 100% since the offline cut on E_T^{miss} is 100 GeV (see Figure 15-82). At high luminosity a rate of 0.2 Hz can be reached with the M_{jj} cut and an ~150 GeV threshold on E_T^{miss}. The total Level-1 1-jet+E_T^{miss} and HLT efficiency for Higgs at high luminosity is shown in Figure 15-84 (right plot) as a function of the E_T^{miss} cutoff. For 0.2 Hz rate (E_T^{miss} > 150 GeV and M_{jj} > 1 TeV/c^2) the Higgs efficiency is 0.7.

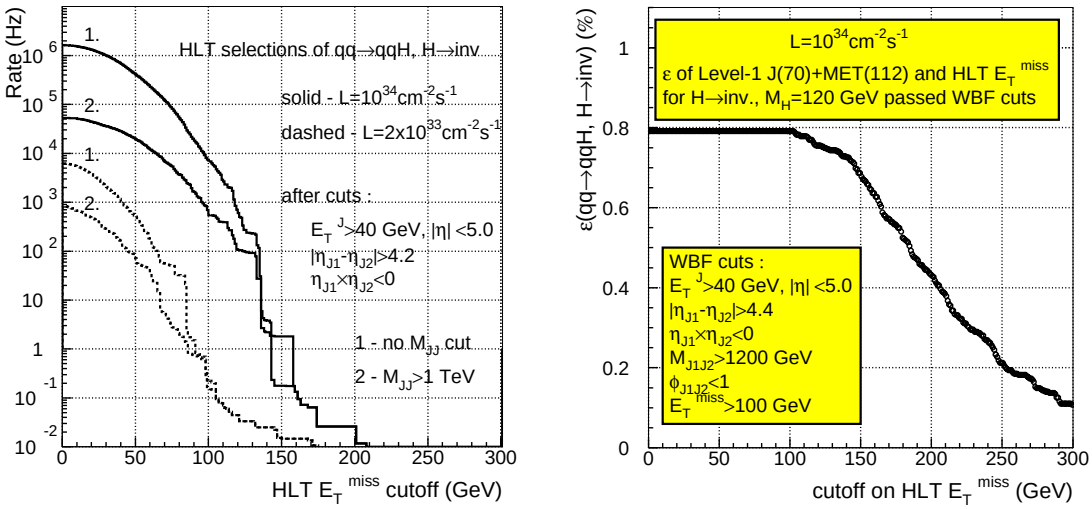


Figure 15-84 Left plot: QCD 2-jet background rate after topological WBF cuts (1) as a function of the E_T^{miss} cutoff for L = 10^{34}cm^{-2}s^{-1} (solid histogram) and L = 10^{33}cm^{-2}s^{-1} (dashed histogram). Right plot: total Level-1 single jet plus E_T^{miss} rate and HLT efficiency for $qq \rightarrow qqH$, with the Higgs decaying invisibly, M_H=120 GeV/c^2, and for events which passed WBF cuts, as a function of E_T^{miss} cutoff at L = 10^{34}cm^{-2}s^{-1}.

15.8.3.4 New Particle Searches: Di-jet Resonances

Many theories of new physics, such as technicolor, “Little Higgs” theories [15-36], and grand unified theories, predict new particles that decay to two jets. Table 15-31 lists the expected limits on various particles at LHC turn-on [15-37].

Table 15-31 Expected mass limits on new particles that decay to di-jets at LHC turn-on [15-37].

Particle	Limit for 2 fb^{-1} (GeV/c^2)	Limit for 100 fb^{-1} (GeV/c^2)
W'	720	920
Z'	720	940
E6 di-quarks	570	780
Axigluons	1160	1300
excited quarks	910	1180

The mass limits from the Tevatron [15-38] were obtained by fitting the di-jet mass spectra to the sum of an ansatz function and a lineshape for the new particle obtained from simulation. Systematic uncertainties include jet energy scale, the background parameterization, the line shape for the new particle, trigger efficiencies, uncertainties on the jet energy resolution, and other effects.

Studies of triggering strategies for these particles were done using a fast simulation for jets written specifically for this study. Generator-level particles are clustered using the same algorithm used for offline reconstruction. The jet energy is then smeared using a resolution function parameterized from the full simulation. The rates from this toy Monte Carlo are found to be in good agreement with the rates from the full simulation.

Because the contribution to the measured width of a di-jet resonance from the calorimeter resolution is large compared to the intrinsic width of most of these new particles, the results for different particles scale simply according to their production cross sections. The search for the Z' is used as an example here. The results for other particles simply scale from these results.

The search for low-mass di-jet resonances at the LHC will be very difficult due to the large backgrounds from Standard Model processes. Significant amounts of data below the resonance will be needed to be fitted to obtain the free parameters in the ansatz. A very approximate estimate for the luminosity needed for a 5σ discovery can be obtained by estimating the number of events due to the signal in a window with a width $\pm 2\sigma$ of the Gaussian part of the resonance and demanding a 5σ excess. This yields a lower limit on the required luminosity because it does not take into account systematic uncertainties and the problem of fitting for the free parameters in the ansatz function.

The results are listed in the second column of Table 15-32. The analysis assumes that data will be needed down to an E_T cutoff of at least at M/4 to use for fitting the ansatz. As an example, the discovery of a Z' with mass 600 GeV/c^2, will require data down to an E_T cutoff of 150 GeV. Table 15-32 lists the rate at $E_T = M/4$, the prescaling factor and resulting rate that would be needed to discover this particle in 1 year of low luminosity running (20 fb^{-1}), and in 5 years of low luminosity running (100 fb^{-1}). Again, this is probably an underestimate of the requirements.

One can also consider the time it takes to discover the Z' as a function of mass for a constant rate to storage. The instantaneous luminosity in LHC is expected to have an exponential decay, and therefore the

Table 15-32 Minimum requirements to discover a Z' decaying to two jets. The first column, "Luminosity", gives the integrated luminosity needed to have a 5σ Z' signal, the second column, "M/4", gives the threshold on the jet trigger E_T used to determine the luminosity, the third column, "rate", gives the rate for the single jet trigger at that threshold, the fourth column, "1 year" gives the rate needed to acquire the events within 1 year (20 fb^{-1}) (and also, in parenthesis, the prescaling factor required and the number of events), the fourth column, "5 years", gives the rate and prescale to acquire that number of events in 5 years. The last three columns give the time required to acquire, under three different prescaling scenarios described in the text, the same number of events.

Z' mass	Luminosity	M/4	rate	rate (prescale) (events) 1 year	rate (prescale) 5 years	fixed rate	fixed prescale	storage rate of 100Hz
(TeV/c^2)	fb^{-1}	GeV	Hz	Hz	Hz	years	years	years
0.6	1.4	150	800	56 (14.2) (40×10^6)	11(71)	1.65	2.8	1.5
0.8	2.3	200	200	23 (8.7) (26×10^6)	4.5 (44)	0.68	1.15	0.63
1.0	4.3	250	55	12 (4.7) (26×10^6)	2.3 (24)	0.35	0.59	0.33
1.2	7.3	300	25	9 (2.7) (33×10^6)	1.9 (13)	0.37	0.46	0.37
1.4	14	350	11	7.8 (1.4) (55×10^6)	1.6 (7)	0.70	0.70	0.70
1.6	20	400	6	6 (1) (60×10^6)	1.2 (5)	1.0	1.0	1.0
1.8	31	450	3.5	-	1.1 (3.2)	1.55	1.55	1.55
2.0	52	500	2	-	1.1 (1.9)	2.6	2.6	2.6

rates at a given threshold also decrease exponentially. Applying this model, and using dynamic prescaling to keep the rate to storage constant, the prescaling factor also decreases exponentially with time.

Table 15-32 also lists the time to discovery for three prescaling scenarios:

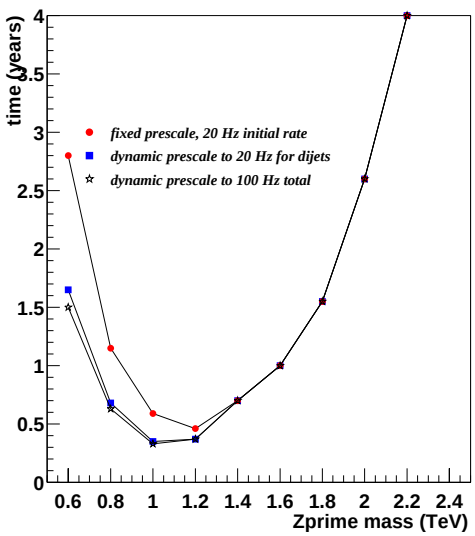


Figure 15-85 Time for the significance for the number of dijet resonance events in a window around the resonance mass to have a 5 σ significance over the background for the 3 prescale schemes described in the text.

- an exponential luminosity decay with time constant of 10 hours, during a beam fill of 10 hours, and a fixed rate to storage for the single jet trigger of 20 Hz;
- a fixed prescale, with a rate to tape for the single jet trigger of 20 Hz at 2×10^{33} $cm^{-2}s^{-1}$;
- an exponential luminosity decay with a 10-hour lifetime, during a beam fill of 10 hours, a fixed rate to storage for all triggers of 100 Hz, taking an initial rate for the single-jet triggers of 5 Hz.

All three scenarios assume 20 fb^{-1} per year for low luminosity running. Figure 15-85 shows the time to discovery for these three options.

15.8.3.5 Standard Model Physics

The measurement of *W* and *Z* boson production properties, and especially their couplings, will be one of the topics which will be studied at the LHC. Deviations may hint at new physics. First manifestations of Supersymmetry may have to be discriminated against a background of *W*+jets and *Z*+jets events. The same holds for heavy flavour physics and, in particular, studies of the top quark, where couplings, rare decay modes, spin characteristics and correlations have to be studied. *W, Z* and top-quark production also provide key tests of QCD.

The main channels for analysis of *W* and *Z*-bosons at LHC will be their leptonic decays. The efficiencies for *W* and *Z*-bosons have been determined in Section 15.2 for electrons and in Section 15.3 for muons. The production rates for $W\rightarrow e\nu$ and $Z\rightarrow ee$ are approximately 20 nb and 2 nb respectively and therefore lead to a rate of events of 40 and 4 Hz respectively at low luminosity.

About 60% of the produced *W*-bosons will have an electron in the fiducial volume of the ECAL $|\eta| < 2.5$. Using the single electron trigger the overall efficiency is 67% (59%) at low (high) luminosity.

Similar numbers are obtained for the muon decay channel. Here the geometrical fiducial acceptance of $|\eta| < 2.1$ is about 50%, and the trigger acceptance using the single muon trigger as defined in section 14-3 is 69% at low luminosity and 42% at high luminosity. For the channel $Z\rightarrow\mu\mu$, with muons in the fiducial volume, the trigger acceptance is larger and amounts to 92 % (86 %) at low (high) luminosity.

The efficiency for top quarks via the decay $t\bar{t}\rightarrow\mu$+X is also rather high, and amounts to 72% at low luminosity. Detailed studies of top production and decay properties will undoubtedly be among the main activities of the first years of running and the ground work against which searches for new phenomena (and potential discoveries) will have to be compared and judged. Top-quark production is often the main background in various searches, foremost in SUSY searches, and for this reason it will have to be understood thoroughly early on in the LHC physics program.

15.8.4 Summary

A draft “trigger table” for the Level-1 Trigger and the High-Level Trigger selection at a “start-up” luminosity of 2×10^{33} $cm^{-2}s^{-1}$ has been shown. The assumption of this table is a total DAQ bandwidth of 50 kHz. It has been shown that high efficiencies for most physics objects are attainable with a selection that remains inclusive and avoids detailed topological or other requirements on the event. The overall CPU requirement of this selection is approximately 300 ms on a an Intel 1 GHz Intel Pentium-III CPU.

Much more sophisticated trigger requirements can, and most likely will, be employed. As an example, at a minimum, as the instantaneous luminosity drops throughout a fill of the LHC bandwidth will be freed from the triggers discussed here. This additional bandwidth can be reallocated to the same triggers by de-

creasing the thresholds, as in the example of the di-jet resonance dynamic prescale factors discussed in this chapter.

The additional bandwidth may also be used in introducing new triggers, e.g. for *B*-physics, a topic that has not been described so far. Introduction of such triggers is then purely an issue of whether there are adequate CPU resources for the selection of the relevant events. The systematic optimization of the track reconstruction code and the extensive use of “regional” and “conditional” track reconstruction allow for the very fast search for and the full reconstruction of *B*-meson decays. Further, the optimization of the tracking code indicates that it can be applied to the full Level-1 event rate at both low and high luminosity. This would extend and complement the current “Level-2” selections described in this chapter. There is ongoing work in the area of optimization of the tracking code and of its application in various parts of the selection. Two examples which include a “Level-2” tracker-based selection and a *b*-tag are described in detail in Appendix G. The same Appendix includes a discussion of selected *B*-physics channels.

The selection presented in this chapter indicates that it is possible to provide the HLT selection of 1:1000 in a single processor farm. Furthermore, the full event record is available, and the software that implements all algorithms can be changed and extended quite easily. The CMS HLT system has great flexibility and provides much room both for improving the selection of the various physics channels, as well as for adjusting to unforeseen circumstances resulting from bad beam conditions, high background levels or new physics channels not previously studied.

15.9 References

15-1 CMS Coll.,“The Trigger and Data Acquisition project, Volume I, The Level-1 Trigger, Technical Design Report”, CERN/LHCC 2000-038, CMS TDR 6.1, 15 December 2000.

15-2 E. Meschi et al., “Electron Reconstruction in the CMS Electromagnetic Calorimeter”, CMS Note 2001/034.

15-3 P. Aspell et al., “Results from the 1999 Beam Test of a Preshower Prototype”,CMS Note 2001/001.

15-4 See, for example, J.P. Peigneux et al., “Results from Tests on Matrices of Lead Tungstate Crystals Using High Energy Beams”, Nucl. Instrum. Methods A378 (1996) 410.

15-5 I. Puljak, Thèse de Doctorat de l’Université Paris VI, 21 Sept 2000.

15-6 CMS Coll.,“The Electromagnetic Calorimeter Project, Technical Design Report”, CERN/LHCC 97-33, CMS TDR 4, 15 December 1997.

15-7 G. Daskalakis and K. Lassila-Perini, “Jet rejection using the pixel matching for the low and the high luminosity”, CMS Note 2002/039.

15-8 V. Innocente, M. Maire, E. Nagy, CERN program library long writeup W5013-E, GEANE.

15-9 G. Bruno et al., “Local reconstruction in the muon detectors”, CMS Note 2002/043.

15-10 N. Amapane, M. Fierro, M. Konecki, “High-Level Trigger Algorithms for Muon Isolation,” CMS Note 2002/040.

15-11 Jet Algorithm Working Group, “Towards standard Jet Algorithms for Run II”, http://niuhep.physics.niu.edu/~blazey/jet_alg/jet_alg.html

15-12 D. Green et al., “Energy Flow Objects and Usage of Tracks for Energy Measurement in CMS”, CMS Note 2002/036.

15-13 R. Kinnunen and A. Nikitenko, “Study of H(SUSY) -> Tautau -> l + Tau Jet + Etmis in CMS”, CMS Note 1997/106.

15-14 D. Denegri and R. Kinnunen, ”Study of H(SUSY) -> tau,tau -> h+ + h- + X in CMS “, CMS Note 1999/037.

15-15 R. Kinnunen, “Study for Heavy Charged Higgs in pp -> tH+ with H+ -> Tau + Neutrino in CMS”, CMS Note 2000/045.

15-16 J. E. Huth et al., Proceeding of Research Directions for the decade, Snowmass 1990.

15-17 S. Eno et al., “A Study of a First and Second Level Tau Trigger”, CMS Note 2000/055.

15-18 D. Kotlinski, A. Nikitenko and R. Kinnunen, “Study of a Level-3 Tau Trigger with the Pixel Detector”, CMS Note 2001/017.

15-19 G. Bagliesi, S. Gennai and G. Sguazzoni, “A L2 Trigger for Tau Hadronic Decays with Tracker Isolation in the Low Luminosity Scenario”, CMS Note 2002/018.

15-20 G. Segneri and F. Palla, “Lifetime-based b-tagging with CMS”, CMS Note 2002/046.

15-21 The ALEPH Coll., Physics letters B 401 (1997) 163-17;
also the DELPHI Coll., E. Phys. J. C10 (1999) 415.

15-22 The LEP Heavy Flavour Group,”Input Parameters for the LEP/SLD Electroweak Heavy Flavour Results for Summer 1998 Conferences”,
LEPHF/98-01,http://www.cern.ch/LEPEWWG/heavy/lephf9801.ps.gz.

15-23 D. Kotlinski and A. Starodumov, “High Level Tracker Triggers for CMS”, Presented at: Vertex 2001, Brunnen Switzerland, September 200,CMS Conference Report 2002/003.

15-24 D. Kotlinski, “CMS Pixel Detector”, to appear in Proceedings Beauty 2002 Conference, Santago de Compostela, Spain, 17th-21st June 2002.

15-25 S. Abdoulline and D. Denegri, “On the Possibility to Observe h -> b-bbar with S/B ~ 1 in SUSY (mSUGRA), and Implications for Tracker and HCAL”, CMS Note 1997/070.

15-26 M. Winkler et al, “Estimation of Alignment Parameters, Using the Kalman Filter with Annealing”, CMS Note 2002/008.

15-27 A. Ostaptchouk et al, “The Alignment system of the CMS Tracker”, CMS Note 2001/053.

15-28 D. Futyan and C. Seez, “Intercalibration of ECAL Crystals in Phi Using Symmetry of Energy Deposition”, CMS Note 2002/031.

15-29 CMS Coll.,“The Hadronic Calorimeter Project, Technical Design Report”, CERN/LHCC 97-31, CMS TDR 2, 20 June 1997.

15-30 CMS Coll.,“The Muon Project, Technical Design Report”, CERN/LHCC 97-32, CMS TDR 3, 15 December 1997.

15-31 F. Matorras et al., “Offline muon station alignment using the inclusive mu LHC production”, CMS Technical Note 1996/005.

15-32 H. Baer et al., Phys. Rev. D 58 (1998) 075008.

15-33 O.J.P. Eboli and D. Zeppenfeld, Phys. Lett. B495 (2000) 147.

15-34 B. DiGirolamo et al., “Experimental Observation of an Invisible Higgs Boson at the LHC”, Workshop on Physics at TeV Colliders, Les Houches, 2001, the Higgs Working Group: summary report, hep-ph/0203056.

15-35 LEP Higgs Working Group, hep-ex/0107032

15-36 M. Schmaltz, “Introducing the Little Higgs”, hep-ph/0210415.

15-37 D. Amidei and R. Brock, “Future Electroweak Physics at the Fermilab Tevatron: Report of the TeV_2000 Study Group”, FERMILAB-PUB-96/08.

15-38 F. Abe et al. (CDF Collaboration), Phys. Rev. Lett. 74 (1995) 3538.

16 Computing Services

Following execution of High-Level Trigger (HLT) algorithms and other processing in the Filter Systems, data are transmitted to the online and offline Computing Services. These data include events selected by the HLT for physics analysis, events selected for calibration, some fraction of events rejected by the triggers, collected calibration information, and a wide variety of data for online monitoring. Some data will be processed online and some will be transmitted to offline systems for event reconstruction, selection, and other offline processing.

This chapter describes the online Computing Services as well as the interface between online and offline computing. Many of the tasks executing in the online Computing Services, such as monitoring and calibration, have been described elsewhere in this Technical Design Report (TDR) or in sub-detector TDRs. This chapter focuses on the general applications running on the online Computing Services and the organization of those Computing Services. The full description of the Tier-0 offline Computing Services will be given in the forthcoming Computing TDR. In this chapter we focus on the interface between online and offline and on the requirements for prompt initial offline processing of the data.

16.1 Introduction

The online Computing Services must be highly reliable yet low cost. In particular, the costs of management, maintenance and operations must be kept to a minimum. This clearly implies that the online Computing Services should be one system managed for use by all the calibration, monitoring, and other processes. In particular, all subdetector processes should run on a uniform Computing Services system. The system must be upgraded frequently as computing hardware and software improves. Again, this calls for a uniform system comprising all online Computing Services.

According to the design of the event builder, all events and calibration data collections are first sent to one of the Filter Units. There, the HLT is used to select events for storage and analysis. In addition, the Filter Units may be used to accumulate and pass on calibration or monitoring information to processes running on the Computing Services. It is expected that there will be a very significant stream of calibration data requiring significant computing resources. It is expected that monitoring to ensure data quality will be comprehensive and therefore a large number of monitoring processes will also require computing resources. Several other online processes, some of which are described in this chapter, will also execute on the online Computing Services.

The interface between online Computing Services and the offline processing systems is designed to provide for maximum data collection efficiency and maximum data quality. This chapter states some of the requirements on that interface, necessary to achieve that efficiency and quality.

Full reconstruction of events selected by the HLT will take place in the offline “Tier-0” computer falicities. Some of the calibration and monitoring tasks will depend on this initial reconstruction step. Therefore, the timely offline reconstruction of selected events is a requirement of the online system.

Some general aspects of planned computing systems, as they are understood today, are also described below. Certainly there will be major improvements in computing hardware, software, and operating systems by the time the LHC starts running. There will also likely be qualitative changes in the optimal way to design a computing system. Nevertheless, a general description of Computing Services as would be designed today is worthwhile.

16.2 Architecture

The Filter Farm, Computing Services, and the connection to the offline Tier-0 centre are displayed in Figure 16-1. Accepted events and other data from the Filter Units (FU) are passed through the Computing Services Network to the appropriate compute Server Unit (SU). Processes on the SU then complete the online processing of the data.

Processes on the FU collect accepted events, divide them into streams, and forward them for offline processing. In normal operation, the accepted events are immediately sent on to the offline Tier-0 centre for the initial offline processing step. The data are saved in the Raw Data Store until the result of the production has been fully validated. Normally events will be promptly forwarded to a Processing Unit (PU). The result of the reconstruction will be saved along with the raw data in a large Object Database.

Most monitoring and calibration processes are completed online in the SU. For example, a monitoring process executing on one of the SU subscribes to a particular type of data, collects it, processes it, then stores the result for display, validation and use in other analyses. An online calibration task works in a similar way.

A possible network configuration, shown below, features a dedicated network for the initial event processing. The benefit of this is independence from other offline processes.

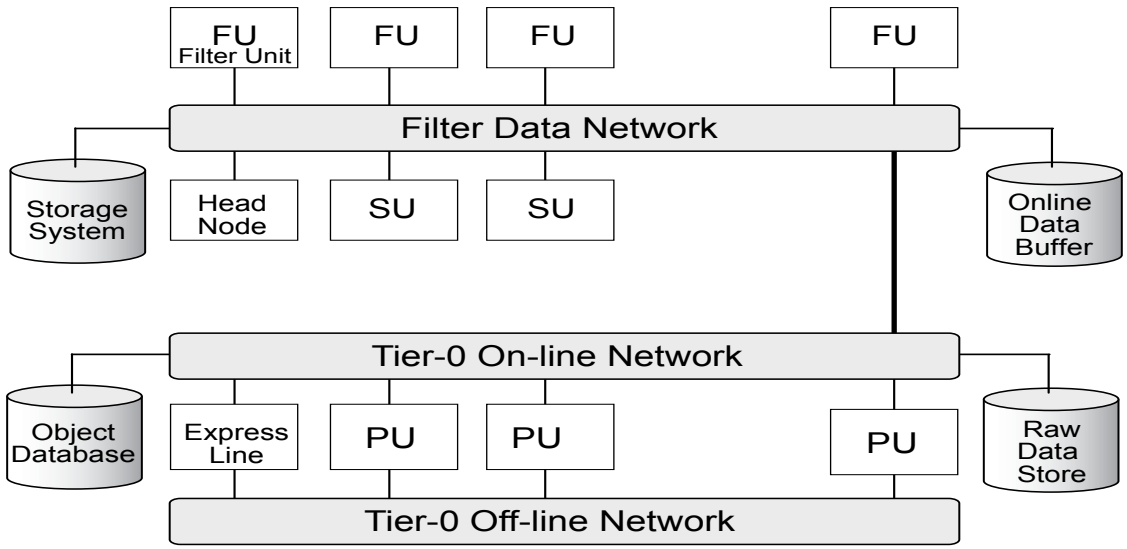


Figure 16-1 Architecture of the Computing Services.

16.2.1 Computing Services Network

Data from the Filter Systems are transmitted to Computing Services on a high-speed network the “Computing Services Network”. It is possible to transmit data from any part of the Filter System to any part of the Computing Services. The Computing Services Network is also used to transmit data for offline storage and processing.

The output event rate from one FU node is very small, well under one Hz. Even the aggregate Filter Farm output event rate is quite small. The data rate due to calibration and monitoring will likely be higher than the data rate due to accepted events; however, it will still be a small fraction of the input data rate to the

Filter System. Therefore, the Computer Services Network will be inexpensive compared to the Filter Data Network. It will likely be built from multiple Ethernet switches connected by trunks. It is even possible that hardware components of the Filter Data Network can be used in the implementation of the Computer Services Network.

16.2.2 Computing Services Processor Cluster

Each unit of the Computing Services cluster of processors is labelled a SU in the figure. The cluster contains of order 100 dual-processor nodes. The processes running on the SU are CPU limited in general, as are those on the FU and PU. Some of the nodes have a very high input data rate, like the FU. Therefore, the requirements for the SU nodes are very much like those for the FU.

The SU should be interchangeable from the point of view of the online computing task such as monitoring and calibration. The system can be made more reliable by having a pool of unused nodes that can be used to replace any node that fails. The compute nodes will not be used for permanent storage of data so that processes can be easily migrated to other nodes.

The Computing Services cluster is managed by one or two head nodes depending on whether sub-clusters are called for. The head nodes manage the tasks on the cluster. Some tasks are resident on particular nodes of the cluster while others run in batch queues on the head node and are subsequently assigned to available SUs. The configuration of resident tasks can be easily modified since the tasks themselves do not depend on which node they run.

The online Computing Services will use a single operating system that is supported by both online and offline software and is highly functional, inexpensive, and supports the most cost-effective computing hardware. The baseline operating system of today is Linux.

16.2.3 Computing Service Storage Systems

All data related to the calibration and monitoring processes running on the online Computing Services will be stored on reliable storage systems like RAID arrays. This will require a high-bandwidth, low-latency connection between processors and storage systems. The storage systems should be backed-up frequently without loss of data taking efficiency.

16.2.4 Online Data Buffer

If the connection to the offline Tier-0 centre is temporarily lost, data collection should proceed uninterrupted. A reasonable data buffer of 100 TB is sufficient to survive a one-week interruption of data transmission from the experiment. It also will allow useful local operation under many conditions. This should be a reliable (RAID) disk buffer. In the event of a problem with the Data Buffer, data can be sent unbuffered to the offline Computing Services.

16.2.5 Estimate of Computing System Size

It is difficult to make an accurate, bottom-up estimate of the required size of the online Computing Services. A sensible estimate based on resource allocation is ~10% of the size of the farm or at least 100 proc-

essors at start-up. Calibration or monitoring tasks that require more resources should be carefully reconsidered.

The storage needs will be dominated by the online Database. Monitoring, calibration, and other tasks will likely make use of some additional storage; on the order of 200 Terabytes at start-up, in addition to the online data buffer.

16.2.6 Network Connection to Offline Computing Services

While the Filter Farm and online Computing Services will be at the experimental site, it is expected that the offline Tier-0 centre will be located on the Meyrin CERN site. An adequate network connection between the Filter Farm and remote Computing Services is required. A 5 Gb/s network link devoted to moving data will be sufficient even under adverse conditions when a backlog of data must be transmitted. Reliable transmission using TCP/IP over Ethernet would be the most straightforward choice today.

16.2.7 Tier-0 Networks

Subject to technology changes, the Tier-0 networks are expected to be based on Gigabit Ethernet with the possible use of 10-Gigabit Ethernet in some areas. The capability exists to have an “online” network dedicated to the initial processing step. This minimizes possible interference from other computing tasks. The dedicated network could be physical or virtual.

16.2.8 Tier-0 Raw Data Store

It is important to minimize the loss of raw data due to processing failures; therefore, the data coming from the detector will be stored until the result production is fully validated. A large data store could be used for this purpose and as a data buffer in the event that the processor system cannot keep up with the data rate from the detector. A disk buffer of 50 TB is appropriate. If more buffer space is needed, the time needed to process 50 TB of data is such that mass storage may be used in the event this buffer becomes full.

16.2.9 Tier-0 Processing Farm

The offline Tier-0 processing farm is currently estimated to contain 500 commodity dual-processor PCs, each of which will have approximately eight times the performance of the high end systems available in late 2002. These nodes are called Processor Units (PU). The processors will be used for the initial processing step, for reprocessing of data, for physics analysis, and for various other tasks. For management purposes, the farm may be arranged in sub-farms to ensure appropriate scaling.

For ease of management and operation, the PU will not be used for storage. The local disk contains the operating system and any temporary files needed for processing. In this way, PUs can be easily added or removed. The operating system can be upgraded at will.

The performance of the PU on the CMS reconstruction task is the most important parameter of the system. Experience has shown that the reconstruction task makes efficient use of the CPU; approximately 85% utilization is a reasonable estimate. A 10% loss of efficiency for hardware or software maintenance can be expected.

16.2.10 Object Database

The Object Database is a major component of the CERN Tier-0 centre. It will store raw data, reconstructed objects and calibration constants, as well as production and analysis status. Ultimately, it will contain many petabytes of data.

The experience of modern High Energy Physics collaborations is that the Database is one of the most complex systems in the experiment. Optimised use of a database can result in significant resource savings albeit it at a price to be paid. The onsite database will likely require a good deal of care and maintenance. Corruption of the data must be avoided. Some downtime will probably be introduced to do the maintenance, leading to inefficiency in the system. It is also important that physics discoveries not be delayed due to problems with the database so discovery channels will be directed to an Express Line analysis stream in addition to a standard stream using the large database. The details of the Database are beyond the scope of this document.

16.3 Computing Services Tasks

This section details how the primary Computing Services tasks will execute on the online Computing Services farm.

16.3.1 Computing for Online Monitors

Every aspect of the detector performance, rates, accelerator performance, readout electronics, Level-1 Trigger, Event Builder, Filter System, HLT software, communications networking, computer systems, calibration, Run Control, offline reconstruction, offline selection, and physics performance will be monitored online. The sources of monitoring data are spread over the online and offline systems. Monitoring data typically pass through some manager node. It is then transmitted to client nodes in Computing Services. Monitoring processes subscribe to particular data categories and subsequently receive those data.

For the smooth operation of the experiment, both the monitoring software and the computing hardware on which it executes will be standardized to provide ease of long-term maintenance and operation of the monitors.

16.3.2 Computing for Online Calibrations

Detector systems rely on calibrations that will be performed both online and offline. Because the LHC event rate is so large, highly accurate calibrations can be made in a realatively short time. The Tracker and Muon systems rely on good alignment. While the systems are designed to be as stable as possible, the experience of many experiments is that alignments are not always stable. The electromagnetic calorimeter has very good energy resolution but suffers from changes in light output with temperature and radiation. These are examples of detectors that may require that up-to-date calibrations be available for initial processing. Online calibrations will be completed on a time scale such that they may be used in the initial processing step.

Most of the calibration will take place in the online Computing Services. This includes the use of data taken purely for calibration purposes and the calibration of electronics. It is also planned that some aspects of detector alignment will take place online. Again, these tasks will be executed on processors in the Computing Services farm.

Calibration data may be in the form of special calibration blocks, full or partial events selected for calibration, or full events selected for physics analysis. In general, all these data are assembled and sent to a Filter Unit for processing. The FU transmits calibration data to the appropriate calibration process on the Computing Services system for further analysis, accumulation of data, and computation of a calibration result. The result is sent to the database for use both offline and online.

16.3.3 Other Online Processes

Many other online applications run on the Computing Services. Examples include event displays, histogram displays, electronic logbooks, validation tasks, expert systems, and calculations of optimal trigger thresholds and prescale factors.

16.4 Other Online Services

16.4.1 Control Room Workstations

Workstations with large displays are provided in the control room to interact with the various tasks running on Computing Services. For example: monitoring and calibration processes must be controlled and checked; event displays must be controlled, calibrations must be validated; entries must be made in electronic logbooks. These workstations can also be used for other tasks.

About ten workstations should be sufficient for interaction with Computing Services tasks. These do not include the RCMS and DCS systems.

16.4.2 Control Room Displays

Large, high-resolution displays in the control room are used to make monitoring and event displays visible over a wide area of the room. Technology advances are expected to make very large displays cost effective. Special workstations will be needed to drive these displays. At this time, plasma displays or projected displays would be the technologies of choice.

16.4.3 Event Display Streams

Online event displays are useful for many purposes. To make use of them, streams of event types for display are selected in the Filter System. These include various types of physics streams, calibration streams, error streams, and clearly, events selected for express line processing.

16.4.4 Local Resources for General Computing

In addition to workstations located in the control room, a pool of about 20 additional workstations should be provided for online use. These may support alternative operating systems and software as well as the standard OS and CMS software. Wireless connections and convenient power for laptops should also be supported all around point 5. This will help provide access to computing during peak activity periods.

16.4.5 Support for Remote Control Rooms

The online Computing Services will provide for remote access to the control room so that detector experts and shift-takers can work remotely. This means that all the displays can be shared efficiently over computer networks; that monitoring and calibration can be controlled remotely; and that audio and video connections are available.

Remote controlled video cameras with pan and zoom facility will help enable physicists outside of point 5 to inspect the condition of many systems including readout electronics, gas systems, control systems, as well as all information available in the control room. Video conferencing facilities in the control room will enable communication between local and remote shift personnel.

16.4.6 Filter Farm Use During Shutdown

The FU nodes and the PU nodes of the Computing Services could be essentially the same. They are both optimised to perform well running reconstruction software. During long shutdowns of the data taking, the Filter Farm should be used to augment the capacity of the Tier-0 centre to the extent that this does not interfere with maintenance and development of the online systems. The primary use is likely to be reprocessing of data. The additional requirements that this use will place on the online Computing Services are not severe given good networking between point 5 and the Tier-0 centre.

16.5 Interface to Offline Computing

The interaction of offline processing, calibration, and monitoring with the online system places requirements on both the online and offline systems. The primary requirement on the offline system is for prompt, reliable processing.

The initial reconstruction, selection, and calibration processing for the data require that very significant offline computing resources be highly available during data taking. The HLT, which has access to the full event information, has been shown to select a very pure data sample. Nearly all the events passing the HLT requirements will be used in a physics analysis, background measurement, or calibration process.

For many reasons, all the data must be processed promptly. Calibrations of detector components will be performed then used in the reconstruction process. These calibrations may also be used for online monitoring, HLT algorithms, and detector performance assurance. Prompt processing also minimizes disk buffer sizes and access to mass storage.

The initial processing task is similar to that in other High Energy experiments. A large number of independent events must be reconstructed or used for calibration. These events can be easily processed in parallel so large farms of inexpensive computers can be used.

Since computing resources will be scarce, processors should not be left idle. During periods of data-taking, a large fraction of the on-site processor resources will be devoted to the initial processing of data from the detector and, during periods when the experiment is not collecting data, resources will be shifted to other tasks. The ability to allocate computing resources, including Filter Farm nodes, to tasks like reprocessing will help satisfy the large on-site computing needs of CMS. Reliability of the processing system is also of great importance. The less organized computing activities, like physics analysis, must not make the system used for initial processing unstable. It should be possible to segregate the nodes allocat-

ed to initial processing from the rest of computing services, yet allow allocation to various tasks depending on running conditions.

16.5.1 Requirements

The requirements for the initial processing step can be easily understood, however, some important numbers are estimates that are uncertain to a factor of two. For example the processing power needed to reconstruct an event can only be estimated from Monte Carlo. It will clearly depend on running conditions and detector performance. Projections of the price performance of future computing systems are also clearly uncertain by at least a factor of two.

The expected rate of events satisfying the HLT is about 100 Hz, at least at a luminosity of 2×10^{33} $cm^{-2}s^{-1}$. There will certainly be periods during which the trigger rate is above this average value due to running conditions. A value as high as 200 Hz can perhaps be expected. At full luminosity, the average event size is conservatively estimated to be 1 MB while at low luminosity the event size will be reduced to about 300 kB due to the reduction in pile-up.

The current estimated processor time needed to reconstruct an event is 3000 SI95-seconds. With 10^9 events per year, the average processing power needed for the initial processing step is 100 kSI95. If the HLT rate is 200 Hz, the instantaneous need for processing power is 600 kSI95 to keep pace with incoming data. Ideally 600 kSI95 should be available, however, 300 kSI95 would be sufficient if processing of some data streams is delayed until a break in data collection. This translates into about 1000 high performance dual processors in 2007.

Since the instantaneous need for processors is larger than the average rate, it must be possible to allocate idle processors to reprocessing or physics analysis routinely when data taking stops. It should also be possible to allocate Filter Farm nodes to reprocessing during long breaks in data taking.

16.5.2 Offline Monitors

There are three types of information to be monitored offline. First, information about the status of the data processing must be regularly updated. Examples include the latency of events in each stage of the system, the processing time as a function of trigger type, the number of recoverable exceptions, the utilization and status of the processing nodes, and the network utilization will be monitored. Second, the calibration results, detector performance, rates into physics channels, luminosity, beam conditions, beam location and beam size will be monitored. It would be beneficial if these could be fed back to the online system within an hour. Third, a comparison of the offline selection algorithms with the HLT results will be monitored.

16.5.3 Initial Processing of the Data

Processing involves reconstruction of events, the addition of persistent reconstruction information to the raw data, the use of physics events to extract calibration and detector performance information, the tagging of physics events with overall event parameters, the selection of physics events as candidates for different physics processes and analyses, the separation of events into streams to optimise the data access for further analysis, and the updating the Object Database.

Ideally, the initial processing of the data coming from the detector should take place on a time scale of about 30 minutes, although this number is quite uncertain. To cover the possibility that calibration or de-

tector performance information may be fed back to the online system, the offline Computing Services will be designed to work on this time scale.

16.5.4 Rolling Calibration

Although, as previously stated, most of the calibration will take place in the online Computing Services, some calibration and alignment will naturally come out of the full reconstruction of physics events. Calibration will certainly be an important part of the initial event processing. The calibration streams may even be processed first so that the results can be used in the reconstruction of physics events.

A “Rolling Calibration”, in which the calibration results derived from one short time period are used in the next short time period to reconstruct, cannot be ruled out a priori. Modern HEP collaborations like BaBar have made this work successfully. The Computing Services will allow for a Rolling Calibration, implying prompt initial processing of data.

16.5.5 Express Line

Some types of events, for example “golden” Higgs candidates, spectacular multi-lepton events, or events with huge missing transverse energy are so interesting that they should be analysed immediately with the result fed back to the online crew and the physics crew on shift. There are many possibilities for backlogs and analysis delays in the standard processing stream so a separate “Express Line” makes sense. An event picked in the HLT for Express Line analysis would be sent immediately to a processor with no backlog. A separate output stream to disk will assure that the results of the Express Line are available. Express Line events should be so rare that duplication of the analysis in the standard stream will not be a significant overhead.

The Express Line will give feedback to the online crew in a matter of minutes. To minimize software overhead, the Express Line can use the same software as used in the standard production stream. Express Line processing could be located in the Filter Farm or in the central Computing Services, depending on performance of the system. Migration from one location to the other should be transparent.

16.5.6 Offline Software

Effective development projects are under way for all aspects of the CMS software. By using prototype software for this TDR a large base of users has been achieved. The same software will be used online and offline to minimize delays in migrating physics selection to the HLT. A discussion of the CMS software is clearly beyond the scope of this document, however, the software does impact our design and the scope of the Computing Services. The time required to reconstruct events, the memory requirements, and the need for network I/O are all based on projections from the current performance of the software prototypes. Significant Monte Carlo production, event reconstruction and analysis have been done, with the prototype software, on large clusters of Linux dual processor nodes, for this TDR.

16.5.7 Quality Assurance, Validation, and Processing Control

The online shift crew will be responsible for Quality Assurance of the data and detector. This may include prompt feedback from the offline processing. Another major QA task is the assurance that the re-

construction and calibration processing is functioning well. Intervention from a processing shift crew or from experts is likely to be needed on many occasions.

There will be professional operators of the CERN computing services as well as detector experts and a shift crew from the experiment to run the data processing. As with other shift crews, remote participation will be very valuable. It increases the pool of experts and allows more participation. It even offers relief from night shifts. A controlled method of remote access for experts and shift crew should be available. For the less expert shift crew, web based tools that are easy to learn and use will be beneficial.

Quality Assurance will continue to be automated over time but will often require the shift crew to look at data summaries and graphs.

A key aspect of the processing chain will be its validation. Again, automatic tools are already in use but human intervention will likely be needed from time to time.

16.5.8 Online and Offline Computing Services Transparency

Most of the tasks described above should be executable both online and offline. This will be useful in the case of emergencies and will allow for rapid testing. In particular, Express Line processing should be possible either offline or online. Vital offline calibrations may need to migrate online in the event of network or offline problems. In general, it should be possible to move calibration and monitoring tasks to optimise system performance and reliability.

16.6 Cluster Computing Systems

A very similar cluster architecture could be used to implement the Filter System, the online Computer Services, and offline Tier-0 system used for initial processing. Recent developments have made clusters of PCs running Linux relatively easy to install and manage. Some properties of this type of computing cluster are discussed in this section.

Computing Systems will evolve over the lifetime of the experiment. This certainly implies hardware that changes rapidly and sometimes significantly with time. New hardware must be integrated quickly. This often implies rapid evolution of the Operating System version to support new hardware.

Computing hardware research and development relevant to the CMS processing problem has been carried out at many HEP labs and universities. In particular, CERN, the LHC computing grid project, other grid projects, and the worldwide CMS computing effort have been very active. One of the most advanced tests of large clusters was the CMS production and reconstruction of a large sample of events aimed primarily at this TDR. The largest and most successful production was carried out in the spring of 2002.

Outside HEP, there is also a very significant move toward large clusters of PCs. Many of these are aimed at applications that require a much tighter coupling between the processors than does the HEP application. Nevertheless, the R&D done outside HEP is also useful for CMS.

There seems to be a consensus that (rack mounted) PCs running Linux and connected over Ethernet are a very effective solution to the HEP problem. Both Pentium CPUs from Intel or similar processors from AMD can be used. There is a competitive market in CPUs, motherboards, memory, disk drives, interfaces, power supplies, and cases. Industry standards allow a good deal of choice among individual components.

There has been steady progress from industry in delivering rapidly improving computer hardware at ever more affordable prices. It is very likely that this progress will continue for some time. Projecting the rate of progress for HEP applications into the future can be very complex. Even measuring past progress is difficult for many reasons. For example, the applications we are interested in were never tested on past computers and actually won't run on them; computers of the more distant past were relatively expensive and the measurement depends on just how expensive a computer is used; and finally details of the hardware are constantly changing and applications need to change to take advantage of new features (like hyperthreading on the P4).

It can be argued that the price performance has doubled every 18 months over the past several years. It can also be argued that the performance has doubled every 2 years and that the price will not continue to drop. It is pointless to get into the details of this argument here since our requirements are also uncertain. Taking the somewhat conservative position that performance will double every 2 years (at constant price), we project a factor of 4 to 8 improvement for the average PC ready at the time of LHC start-up. It can also be projected that memory speeds will be sufficient for HEP processes to take advantage of increased CPU performance.

The 2001 International Technical Roadmap for Semiconductors published by the Semiconductor Industry Association[16-1] lays out predictions for technology directions for the next decade, including transistor density, chip size and clock speed. Their predictions indicate that steady progress can be made for some time. The silicon feature size will continue to decrease allowing more gates to be put on a single chip, clock speeds to increase, and keeping power reasonable. Intel has announced a 90 nm feature size for next year, ahead of predictions.

There is evidence from the P4 performance that an increased number of gates does not benefit HEP applications as much as some others. Hardware improvements are likely to cause changes in the optimal system. There may ultimately be a move to more CPUs on a single chip. The one clear lesson from the past is that change is a certainty.

One cautionary note is in order. Market forces are hard to predict. It can be argued that there is no great need for commodity computers to continue to improve in CPU performance and that manufacturers are not reaping great profits by making hardware improvements. It is probably true that the greatest uncertainty in future computer performance is due to market uncertainties, not technical hurdles.

16.6.1 Cluster Management

The management of large numbers of functionally similar nodes has progressed significantly over the last several years. For example the NPACI-developed system, Rocks[16-2], allows for the management of large numbers of nodes with interaction with individual nodes only at the time of installation. The operating system on a large cluster can be changed in a few minutes allowing rapid reconfiguration of the cluster for various tasks. In the Rocks system, the system partition of the local disk is considered completely expendable and is wiped each time the system is restarted.

Within this type of cluster management system, there are "head nodes" or manager nodes for each major task. Processor nodes can be reassigned to tasks very quickly. For the Computing Services application, it may be useful to dedicate networks to some of the task, like the initial processing task.

In the end, developments for the Grid will play an important role in our choice of cluster management software and batch systems, since the onsite CERN Computing Services for CMS should be integrated into the Grid. Nevertheless, the initial processing step should probably be isolated from other Grid activities.

Technologies for implementing system management tasks are presently under investigation. Several commercial and public domain products exist, with the more recent ones being based on the DMTF standards [16-3] where the CIM (Common Interface Model) [16-4] model is used to describe the components of the system. This technology has been integrated with Web technologies such as XML and HTTP in the form of Web Based Enterprise Management (WBEM) [16-5].

The tools that the LCG project [16-6] will propose to manage the PC farms distributed over the future LHC Grid network will be followed closely, since it is very likely that these tools will be the reference for all LHC experiments and could be easily adopted to also manage the online systems.

16.6.2 Current Computing Hardware

It is useful to take a snapshot of current computer hardware in September of 2002 as a point from which to project. Systems currently being purchased are 1U rack-mounted dual 2.5 GHz Pentium 4. They give a performance on CMS applications equivalent to 120 SI95 for the two CPUs running a total of four processes to take advantage of hyperthreading. Memory speed has been measured to be 1.5 GB/s for a copy. The systems have two onboard Gigabit Ethernet interfaces. A test of one interface showed a 930 Mb/s transfer rate in TCP/IP. The key number is the CPU performance. The others are more than adequate for the CMS farm needs. The cost of such a system is about $1900.

16.6.3 Future Computing Hardware

Assuming minimal changes in available computer architecture of the next 4 years, we can imagine the dual processor system of 2006. The dual processors will have about 2 GB of memory. Given the expected growth of memory capacity, this should be very standard. The computational nodes purchased in late 2002 have high-performance dual Gigabit Ethernet interfaces on the motherboard. This networking would be sufficient for the Farm of 2006, but low latency 10 Gigabit Ethernet will probably be standard. The nodes should have small reliable disks. Neither disk speed nor disk size is of much importance and it is possible that no disk at all will be needed.

The processor systems should have near optimal overall price performance. That is, the cost of hardware per SI95 should be optimised without seriously impacting management and operation costs. Since a cluster node can be added or removed from the system without serious impact, the procurement and maintenance of these commodity computers will be optimised for price performance. If the Moore's Law improvement continues, a processor node loses much of its value each year. After about four years, the rack space becomes more valuable than the node and the node should be replaced. Since replacement with a higher performance system is a very viable maintenance option, the cost of warranties and maintenance of the processor nodes can be minimized.

Linux is the choice for the operating system. Linux is inexpensive, supports the hardware with the best price performance, and is the primary development and production system for CMS software. It is likely that the choice of operating system will change over the lifetime of the experiment.

16.7 References

16-1 Semiconductor Industry Association, http://www.semichip.org/

16-2 NPACI Rocks, http://www.rocksclusters.org/rocks-documentation/2.3/

16-3 Distributed Management Task Force (DMTF), http://www.dmtf.org/

16-4 Common Interface Model (CIM), http://www.dmtf.org/standards/cim_schema_v27.php

16-5 Web Based Enterprise Management, http://www.dmtf.org/standards/standard_wbem.php

16-6 The LHC Computing Grid (LCG) Project, http://lcg.web.cern.ch/LCG/

Part 5
Project Organization, Costs and Responsibilities

17 Organization and Responsibilities

17.1 Project Organization

The CMS Trigger/DAQ (TriDAS) project, with S. Cittolin as the Project manager, is divided into two projects, the Trigger project and the Data Acquisition project. Both the Trigger Project Manager (W. Smith) and the DAQ Project Manager (S. Cittolin) are assisted by a Technical Coordinator (J. Varela for the Trigger and A. Racz for the DAQ) and a Resource Coordinator (J. Varela for both projects).

The DAQ project is further divided into the following subsystem projects:

1. The Detector Control project includes the coordination of the detector control systems, the technical support to the detector DCSs, the communication with external systems and the integration of the central DCS.
2. The Data to Surface project comprises the FED and FRL systems, the Trigger-DAQ column integration and the Local DAQ systems.
3. DAQ Integration includes the FED Builders and the RU Builders, the Event Manager and control networks, the Online Software framework, the Run Control and Monitor system and the Data archives.
4. The Event Filter project includes the Filter software framework, the Higher Level Trigger monitoring and the farm management.

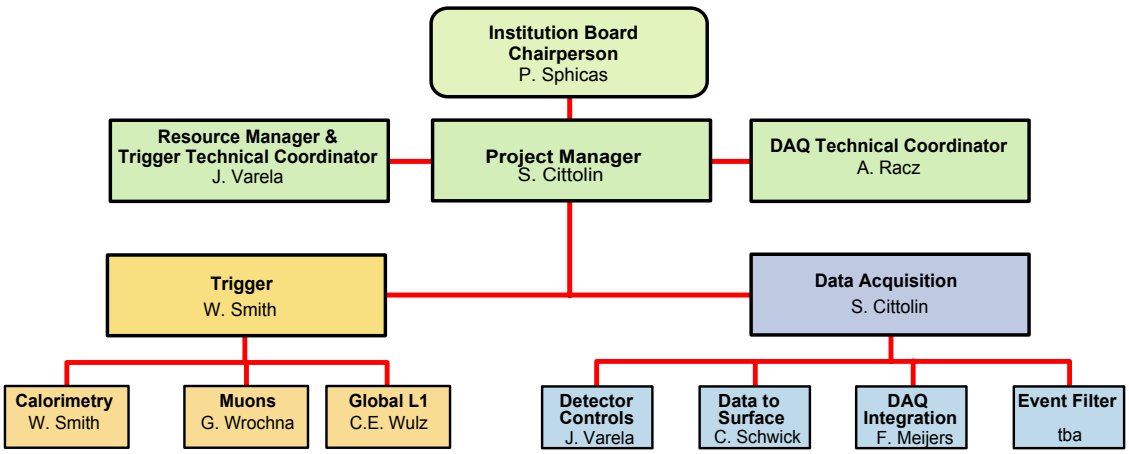


Figure 17-1 Organization of the Trigger and Data Acquisition Project

The above sub-projects and the corresponding coordinators, along with the corresponding sub-projects and coordinators for the Trigger sub-project of TriDAS, are shown in Figure 17-1 and in Table 17-1 which also lists the countries participating in each sub-project. The Event Filter subproject includes tasks which involve people in the combined CPT (Computing, Physics and Trigger/DAQ) project of CMS. The coordinator for this subproject will be identified in the future.

The organization will evolve at commissioning and operation time in order to include the maintenance and operation of the control rooms, the readout and the filter farms.

Table 17-1 Subprojects, participating countries and coordinators in the DAQ project.

Subsystem	Countries	Coordinator
Detector Control System	CERN	J. Varela
Data to Surface	CERN, Italy, Switzerland, UK, US	C. Schwick
DAQ Integration	Belgium, CERN, Greece, Italy, Switzerland, US	F. Meijers
Event Filter	Austria, CERN, France, Hungary, Switzerland, US	TBA

17.2 Overall Schedule

The schedule is divided into six tasks:

1. R&D program,
2. Preseries production,
3. Online software,
4. Mass Production and Procurement,
5. Installation and Integration
6. Commisioning.

Figure 17-2 shows the detailed time lines within each task.

The R&D program is expected to be completed by the middle of 2003. Preseries production will commence at the begining of 2003 and mass “production” in early 2004. It is expected that installation in USC55 will commence in April 2005 while integration will commence in July 2005. CMS is expected to be ready to take data at the end of March 2007.

The current plan calls for only partial completion of the DAQ system at that point in time. In particular, the full system necessary for the readout of the CMS subdetectors and the transportation of the signals to the surface counting room will be in place. The first stage of the Event Builder will thus be completely installed.

The design of the second stage of the Event Builder comprising eight Readout Unit Builders and their associated Filter Subfarms allows a staged deployment. It is planned to install a single Readout Unit Builder by the end of 2005 and up to four by the end of 2006. This would allow a maximum Level-1 rate of 50 kHz at the start of the LHC in 2007. The system will be completed, for a 100 kHz event rate capability, by the end of 2008.

Progress towards the schedule is monitored internally by the Trigger/DAQ group every six months. Progress is also officially reported once per year through the CMS Annual Review (AR) and at more frequent intervals through reporting of the CMS L2 and L1 DAQ milestones to the LHCC.

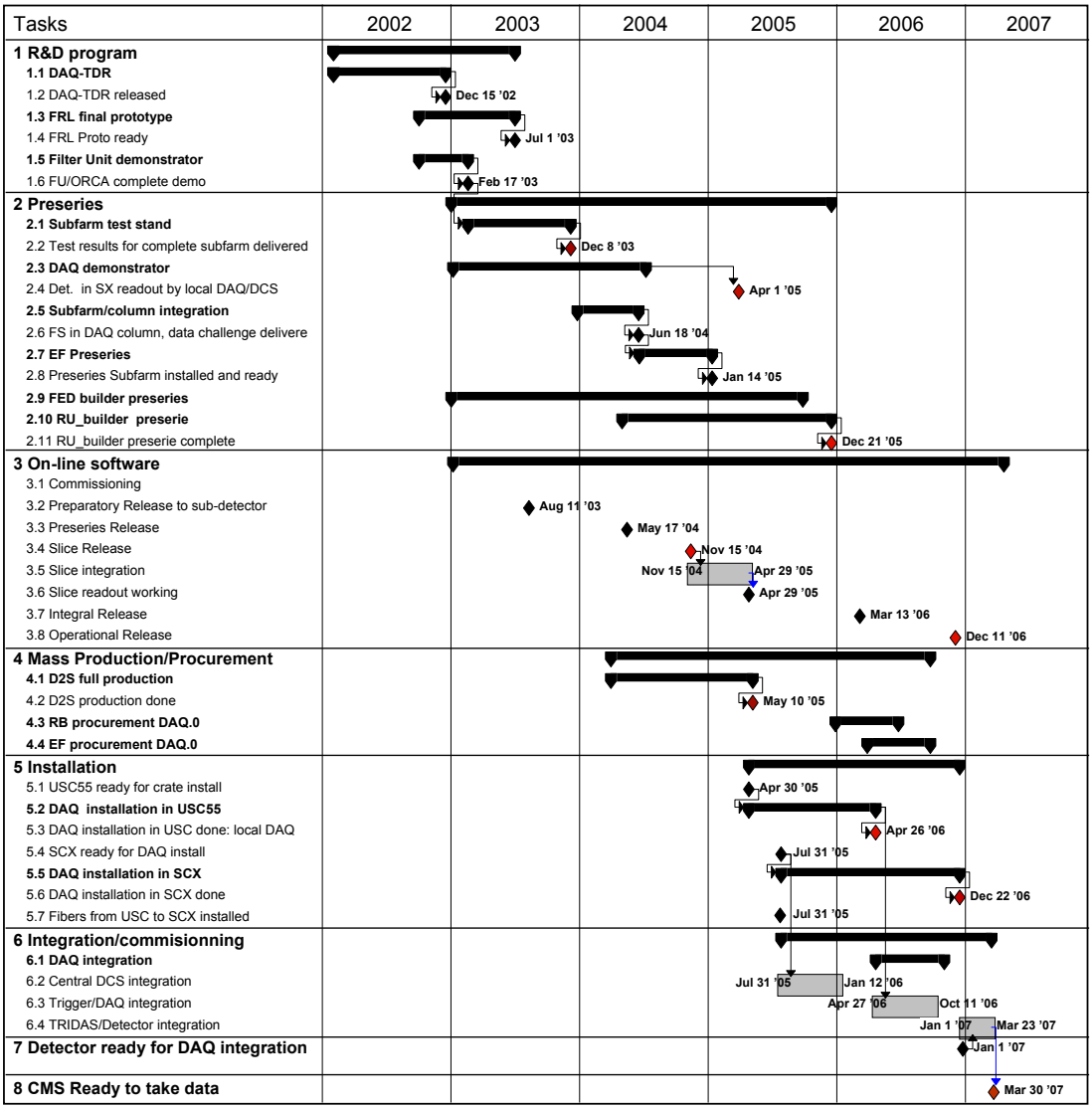


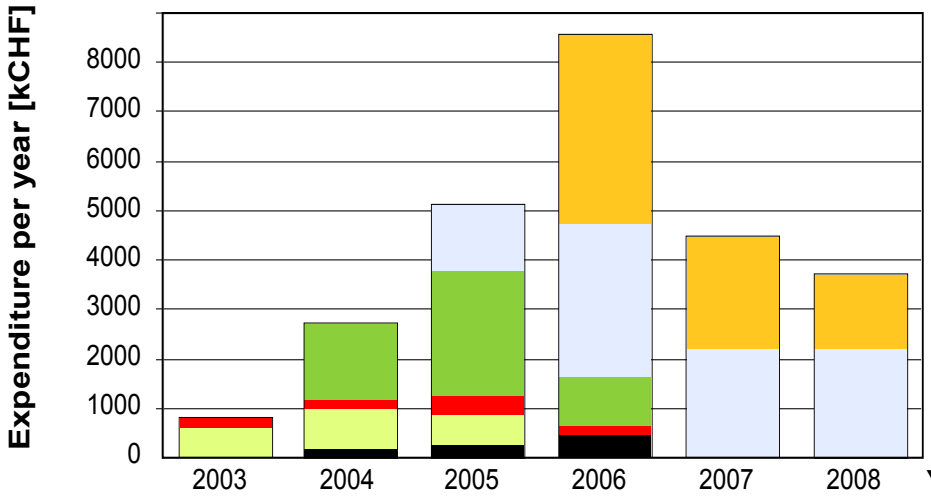
Figure 17-2 Schedule of the Data Acquisition project.

17.3 Costs and Resources

The cost estimate for the CMS DAQ is based on a detailed breakdown of the components of each subsystem. The price of custom-designed items is estimated from the costs of the final prototypes of the R&D program extrapolated to the preseries production.

A large fraction of the total costs are from off-the-shelf items such as cables, fibers, PCs, network interfaces, switches and data storage devices. The cost of these items is estimated from prices known today versus performance trends extrapolated to the expected purchase date. For example a one year delay in purchase of PCs should result in a 25% drop in cost for the equivalent CPU power.

Figure 17-3 displays the expected costs for each subsystem in the period 2003 through 2008. It is assumed that the deployment of the Event builder is completed by 2008.



Expenditures	2003	2004	2005	2006	2007	2008	TOTALs
Event Filter	0	0	0	4372	2623	1749	**8744**
Readout Builder	0	0	1184	2763	1973	1973	**7893**
Data to Surface	0	1511	2519	1008	0	0	**5038**
Detector Controls	92	276	368	184	0	0	**920**
Preseries	376	939	564	0	0	0	**1879**
DAQ Integration	0	180	269	449	0	0	**898**
			Level-1 maximum rates		50kHz	100kHz	**25372**

Figure 17-3 Cost estimate (in kCHF) for the Data Acquisition project.

The total project cost in the years 2006 through 2008 as a function of the Level-1 readout rate capability with such a deployment strategy is displayed in Figure 17-4. In the figure, the entry “DAQ infrastructure” refers to all the DAQ elements necessary to equip the detector readout (i.e. the “Data to Surface”, the Detector Control System, and the DAQ Integration).

Total(MCHF)/Rate(kHz)	2006	2007	2008
DAQ infrastructure 100%	**8.7**		
Level-1 50.0 kHz		**17.1**	
Level-1 100.0 kHz			**25.4**

Figure 17-4 DAQ project cost projection, in MCHF, as a function of time.

17.4 References

17-1 CMS Milestones, CERN/LHCC 2002-035/G_015 30 Sep 2002. Available as edms document 357999 (http://edms.cern.ch/document/357999/1.)

Appendix

A Overview of Switching Techniques and Technologies

A.1 Queuing Architectures

A generic switch can consist of a combination of basic switching elements (e.g. crossbars) and buffering elements that are used for the instantaneous absorption of traffic bursts. By definition, any such buffer, in a switch with 2N ports, must be able to handle a traffic N times faster than the link speed if the switch is not going to block on some outputs. Given a basic switching element, there are three types of buffering, referred to as “Input Queuing”, “Output Queuing” and “Combined Input Output Queuing”. The three queuing types work according to what their names would imply:

- Input Queuing (IQ): here the switch has FIFOs on each switch input port. Requests, from the Network Interface Cards (NICs) connected to each port result in the immediate storage, into these input FIFOs, of the data from the NICs.
- Output Queuing (OQ): here the switch has FIFOs on each switch output port. Packets/cells arriving from other parts of the switch, and destined to a NIC connected to an “output” switch port are buffered in these FIFOs.
- Combined Input Output Queuing (CIOQ): here the switch employs both input and output queue switches. The situation is shown schematically in Figure A-1. This figure, in one, clearly illustrates the IQ and OQ cases independently (so, for example, an IQ switch simply does not have the output FIFOs).

As expected, the four queuing architectures have been studied extensively. In what follows we summarize some key features and conclusions, relevant to switch efficiency, of all the R&D in this field.

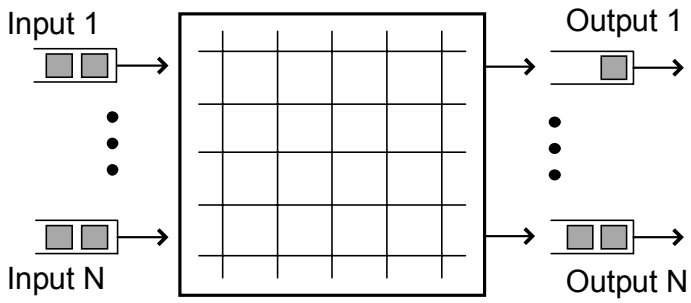


Figure A-1 Schematic representation of a switching fabric with both Input and Output Queuing.

Foremost, is the proof that IQ switches employing only a single FIFO per input port suffer from the “head-of-line” blocking effect, and as a result, the effective switch bandwidth can be limited to just 59% of the total available one [A-1]. Head-of-line blocking is a condition arising from the FIFO nature of the queued requests (on input). Three different cases of head-of-line blocking are displayed in Figure A-2.

Head-of-line blocking is, by definition, not present in OQ switches. In fact, fully non-blocking switches in practice utilize OQ FIFOs. The problem with OQ switching is that for a non-blocking architecture, the bandwidth of the memory used for the output FIFOs quickly becomes prohibitively large. Take a switch with N input and N output ports. For this switch to appear like a genuine cross-bar, i.e. a non-blocking

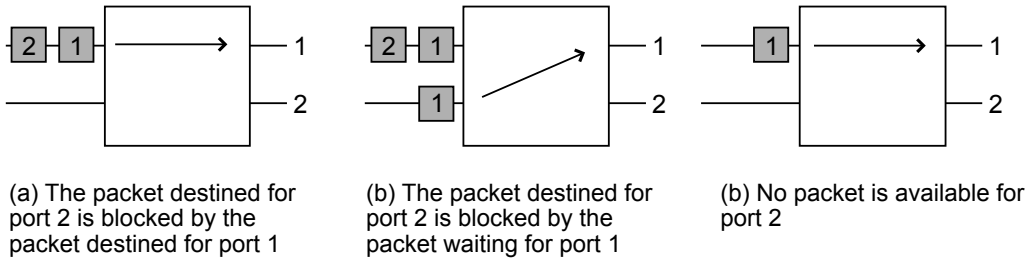


Figure A-2 Three different sources of the head-of-line blocking effect. In all three cases port 2 is idle, thus lowering the switch utilization efficiency.

switch, the output FIFOs must be capable of receiving data from all N ports “at the same time”, which in practice means that the write-access to these FIFOs is N times faster than the input link speeds. This quickly leads to technologically impossible (or, once possible, very expensive) demands on these FIFOs. Take, as an example, the case of a 1,000 port switch with Gb/s links, like the one required by the Event Builder. A OQ switch with these features would have to employ memories functioning at a throughput of 500x1Gb/s = 500 Gb/s. Even with the assumption of a data path to these memories which is as wide as 512 bits, this would imply 1ns access times. This situation should be contrasted by the equivalent need for a IQ switch, where the (input) FIFOs simply have to run at the link speed of 1Gb/s, which for the same ultra-wide path of 512 bits assumed before would translate to a 0.5 μs access time (in practice, of course, 50 ns memories with 16/32 bit wide paths are used).

In fact, OQ switches for even smaller configurations (like 64×64) are a challenge today. The conclusion is that, while OQ-based switches can indeed provide close to 100% efficiency, such switches are impossible, or very difficult, to manufacture (especially with a large number of ports).

Recently, the networking and computing industries have been devoting significant resources to studying the combination of the IQ and OQ schemes into the Combined scheme (CIOQ). One major result, is that a CIOQ switch can be made to behave identically to a (hypothetical) OQ switch with the same number of ports and link speed if it employs, internally, only a factor 4 speed-up (as opposed to the factor of N ended by the OQ switch). This is a very encouraging result for potential consumers of cross-bar-like switches with large numbers of ports (e.g. like the Event Builder).

A.2 Myrinet Technology and Products

A.2.1 Network Technology

Myrinet is a packet switched network. A packet consists of a routing header, followed by an arbitrary length payload and terminated by a trailer that includes a CRC byte computed on the entire packet (see Figure A-3). The routing header is used by the switches, which strip these header bytes as they steer the packet through the network. Hence, the full route is determined at the sender, known as source-routing.

A Myrinet link is full duplex with a speed of 2 Gbps in each direction. Communication is reliable with very low bit error rates (below 10^{-15}). There are three physical media available;

- byte-wide parallel, called SAN,
- serial HSSDC (High-Speed Serial Data Connector) [A-2] and

- fibre (50/125 multimode).

The parallel SAN is actually 10 bits wide (it includes a data/control and backpressure bit) and the serialising/deserialising process includes a 8/10 bit conversion. Hence, the baud rate on the serial medium is actually 2.5 Gbps. The serial copper and fibre media are the same physical media as Infiniband 1x.

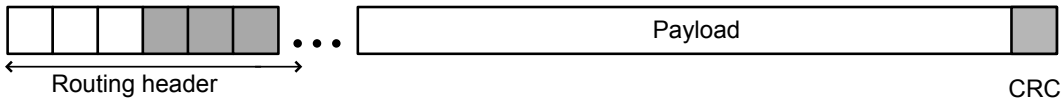


Figure A-3 Myrinet packet structure.

The building block of a Myrinet network is a 16-port switching chip, called Xbar16. It can be used to construct an 16 port switch or to interconnect them to build networks of various topologies of varying sizes. The most common topologies are Clos networks.

The crossbar switches employ wormhole routing. The routing decision is made as soon as the packet header arrives, the stripped header is then sent to the chosen output link and the rest of the packet follows without being internally buffered. The packet header creates a temporary circuit (wormhole), which closes as the trailer passes through each device. Thus, a worm can stretch across many nodes and links at any time. Wormhole routing minimises latency and buffering requirements compared to switches using store and forward techniques. It also has the advantage that it allows arbitrary length packets.

When the desired output port is not available, the worm is stalled and this information is propagated upstream using back-pressure flow control. This flow control is basically Xon/Xoff and utilises a small 'slack' buffer at the receive port of the link to absorb the reaction delay. The back pressure signal is transmitted over the link in the opposite direction.

The network link level flow control guarantees the delivery of packets at the expense of an increased potential of mutual blocking in the switches and, hence, a reduced maximum utilisation of the network.

Myrinet does not support multicast or broadcast in hardware, which is problematic in wormhole-routed networks due to the difficulty of performing flow control over a multicast tree. Thus, multicasting has to be done in software by multicopy unicasting.

A.2.2 The Network Interface Card

A simplified schematic of the network interface card is shown in Figure A-4. A Myrinet host adapter contains a LANai chip with a RISC processor core, an interface to the external bus and the entire interface to the Myrinet link integrated in one VLSI chip. From LANai version 10 onwards, the LANai integrates more than one Myrinet link port.

In addition to the LANai there are fast local SRAM and a PCI interface on the adapter card. The SRAM serves as staging memory for buffering packets and also as the code and data memory of the LANai processor.

There are a number of DMA engines on the NIC: Each Myrinet port interface has a separate send and receive DMA engine that are used between the network FIFOs and the staging memory on the card. In addition, a separate chip implements a PCI DMA engine that is used for moving data between the SRAM and the host main memory over the PCI-bus via the LANai external bus. It can act as a bus master to gath-

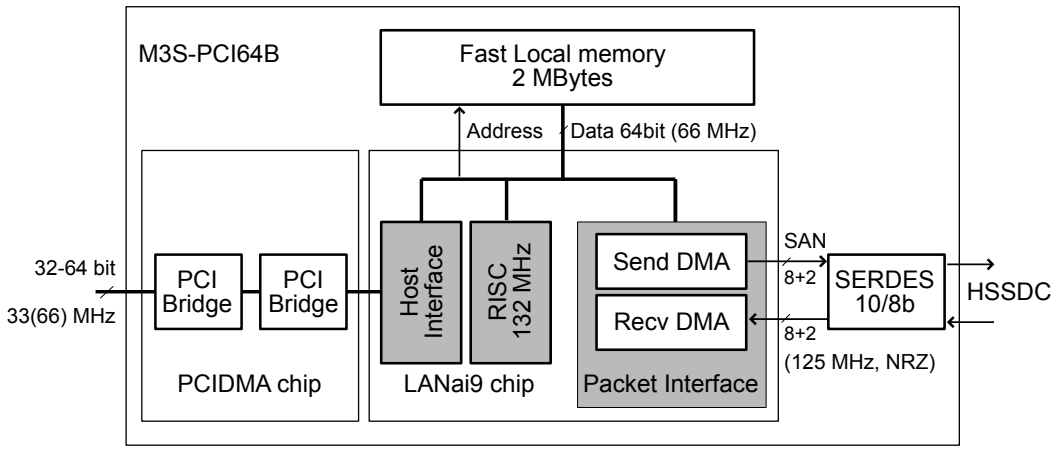


Figure A-4 Block diagram of the Myrinet NIC. Shown is the LANai9 based M3S-PCI64B. The NIC based on LANai10 has multiple packet interfaces and associated SerDes, integrated PCI-X interface, higher bandwidth internal bus and faster RISC.

er data blocks from or scatter to the host memory over the PCI bus. At the same time, the DMA engine computes a datagram checksum of the data it transfers. All DMA engines can work in parallel which allows pipelined operation.

The LANai processor executes a customizable Myrinet control program (MCP). A typical MCP supervises the host and link DMA engines, interacts with the host and implements a communication protocol.

More detailed characteristics of the NIC are given in Table A-1.

Table A-1 Myrinet NIC features.

product	**M3-PCI64B**	**to be announced**
host interface	PCIbus 64bit/66 MHz	PCI-X
	LANai9	LANai10
RISC clock	133 MHz	300 MHz
local bus	64 bit / 132 MHz	64 bit / 300 MHz
local memory size	2 MByte	2 MBytes
local memory bandwidth	1056 MByte/s	2400 MByte/s
effective Myrinet link speed	2 Gbps	2 Gbps
number of Myrinet ports	1	2
port interface	Myrinet	Chameleon (Myrinet,Infiniband,Ethernet)
PCI DMA engine	chained blocks	chained blocks
link SDMA, RDMA engine	single block	multiple blocks

A.2.3 Products

Myricom introduced in 2000 the third generation Myrinet products, called Myrinet-2000. These have an effective link speed of 2 Gbps. Current crossbar chips have 16 bidirectional ports. Information on these products can be found at [A-3].

A.3 References

A-1 M.G. Hluchyj and M.J. Karol,"Queueing in High-Performance Packet Switching",IEEE Journal on selected areas in communications, 6 (1988).

A-2 HSSDC, (High Speed Serial Data Connector) is defined in the FibreChannel FC-PH2 standard. (http://www.fibrechannel.com)

A-3 Myrinet products from Myricom, Inc.,Arcadia, CA, USA, see http://www.myri.com

B Event Builder Prototypes and Simulation

B.1 EVB Performance with Myrinet Switches

This section reports on simulation results of the EVB performance of switching fabrics based on Myrinet crossbars. The aim is to determine the efficiency of the utilisation of the network bisection bandwidth for event building traffic without the use of traffic shaping techniques.

In order to estimate the effect of blocking at the maximum network utilisation, a number of NxN square networks are considered, all composed of 8×8 (i.e. 16 port) crossbars with N sources sending to N destinations. Hence, the traffic is strictly unidirectional. (ignoring back-pressure)

- 8×8 crossbar
- 64×64 delta network. This network has a single path between any source-destination pair. The sources and destinations are on opposite sides of the network. Note that mixing sources and destinations on the same crossbar would lead to a potential deadlock.
- 64×64 three stage Clos network. Here multiple paths are possible between any source-destination pair. When operating with a fixed path assignment (for example choosing the intermediate crossbar number as the output port number of the destination crossbar), this network is equivalent to the two-stage delta network, except that it is dead-lock free. Alternatively, one can choose a random path and the results presented below are with this algorithm.
- 64×64 folded Clos network. This network is the same as the previous, except that here the sources and destinations are mixed on the same crossbar. Hence, the links between the outer layers and intermediate layer are utilised in both directions.

The maximum utilisation of the network is defined as the total saturation throughput divided by the bisection bandwidth and has been determined by simulation. The simulation models a network of Myrinet crossbars with circulating token arbitration (see Appendix C). No overhead is assumed. Two cases are considered, random traffic and event building without traffic shaping. Simulation results for random traffic provide a sanity check while the EVB traffic pattern delivers performance estimates of the EVB system. The results are presented in Table B-1.

Table B-1 Maximum network utilisation.

	8×8	**64×64**	**64×64**	**64×64**
	crossbar	delta network	Clos network	folded Clos
number of stages	1	2	3	3
switches per stage	1	8	8	8
paths for each source-destination pair	1	1	8	8
random traffic	62%	45%	35%	47%
EVB traffic - fixed size	100%	multiple states		
EVB traffic - variable size				
average=2k, rms=1k	70%	47%	34%	48%
average=2k, rms=2k	55%	36%	25%	38%

In the case of random traffic each source sends independently to a destination chosen according to a uniform distribution. The packets are of fixed size. The maximum utilisation for a single stage is estimated to be 61.9%, which is in agreement with the analytical value of 61.8% for a 8×8 crossbar with an input queue. The maximum utilisation for multiple stages is reduced to 45% and 35%, for two and three stages, respectively. This is expected because of the increasing potential for blocking at the additional layers. The wormhole routing implies that a blocked packet will also block other packets along it's path back to the source, since there is essentially no buffering in the switches and the path stays allocated until the trailer is through. The folded Clos performs considerably better than the unfolded one.

In the case of event building, each event has to be assigned to a destination. With trivial assignment, event number n is sent to destination d (d=0,..,N-1) according to d=mod(n,N). For the multistage networks an alternative assignment which reflects the structure of the underlying network is d = 8*mod(n,4) + n/8. Each event fragment is sent as a single packet, which is possible with Myrinet as packets can be of arbitrary size and the maximum length is not limited.

For fixed size fragments the utilisation reaches 100% for the 8×8 crossbar after the first 8 events. The networks composed of crossbars exhibit more states than the 100% efficient one, and depending on the initial condition and the destination assignment algorithm, the traffic can lock into one of them.

For variable size fragments, the coherent event building pattern is disturbed. Fragment sizes are generated independently for all sources and according to a log-normal distribution. The parameters are set to an average of 2 kB and RMS equal to 1 kB and 2 kB. It can be seen that for these conditions the maximum utilisation follows the same trend as for the case of random traffic and that the performance is reduced with increasing fluctuations of the event sizes.

B.2 Gigabit Ethernet Layer-2 Frame Test-benches

The measurements on the two test-benches described here were made before the two-staged event builder design was introduced. The fragment size was then expected to be around 2 kB, so throughput measurements were made with fragment sizes up to 4 kB only. An operating system with lower overhead was selected in order to achieve good performance also with small fragment sizes.

B.2.1 Single-chassis Configuration for 15 x 15 Event Builder

The 15×15 Gigabit Ethernet Event Builder test-bench comprises 30 PCs emulating the RUs and BUs. In addition, one PC serves as the event manager BM. The Gigabit Ethernet NIC of each host is connected to a FastIron-8000 switch from Foundry Networks [B-1]. A fully populated FastIron switch comprises 8 modules with 8 Gigabit Ethernet ports each, connected to a crosspoint backplane. Each module contains 2 MB of shared memory to buffer packets. The bandwidth of the memory system and backplane is dimensioned such that the switch is fully non-blocking. In the present configuration the switch is half populated (see Figure B-1). Details on the host configuration can be found in Table B-2.

The application software running on the PCs implements the event building protocol shown in Figure B-2, using Layer-2 frames to transmit data. To reduce congestion at the output ports, each BU requests event fragments from each RU sequentially. This is similar to the destination based traffic shaping. However, several events are built concurrently in order to use the links efficiently. The protocol has been augmented with a sequence number check, time out and retry mechanism to recover from occasional packet loss.

Table B-2 Configuration of hosts used in EVB Gigabit Ethernet prototype.

Configuration element	Equipment type
Host	Pentium-III PCs with i840 chipset
PCIbus	64b/66MHz
Gigabit Ethernet NIC	SysKonnect SK9821[B-2]
Operating System	vxWorks real-time OS

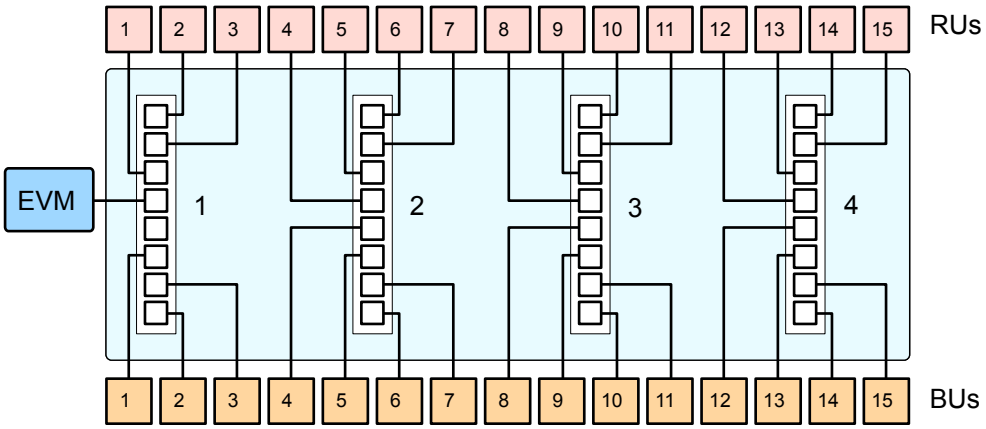


Figure B-1 15×15 Gigabit Ethernet configuration.

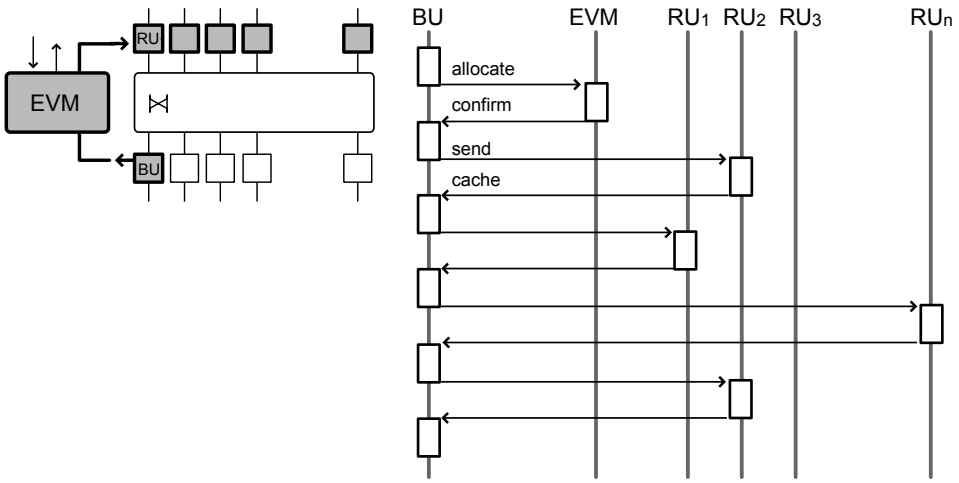


Figure B-2 Event Building protocol used for Ethernet Prototypes.

The throughput per node as a function of the fragment size is shown in Figure B-3, together with the calculated maximum performance taking header sizes into account. It reaches 116 MB/s, which corresponds to nearly 100 % utilization of the available bandwidth. The saw-tooth structure is due to the fact that no aggregation of event fragments into packets is performed. For a fragment size of 2 kB the achieved throughput per node is 105 MB/s. The size of the Event Builder is varied from 1×1 to 15×15 in Figure B-4 for a number of fragment sizes. It can be seen that the throughput scales with the size of the configuration. The measurements were carried out with a maximum fragment size of 4 kB. The 16 kB

fragment size now expected in the RU Builder will lead to a slightly better utilization of the bandwidth, assuming no packet losst.

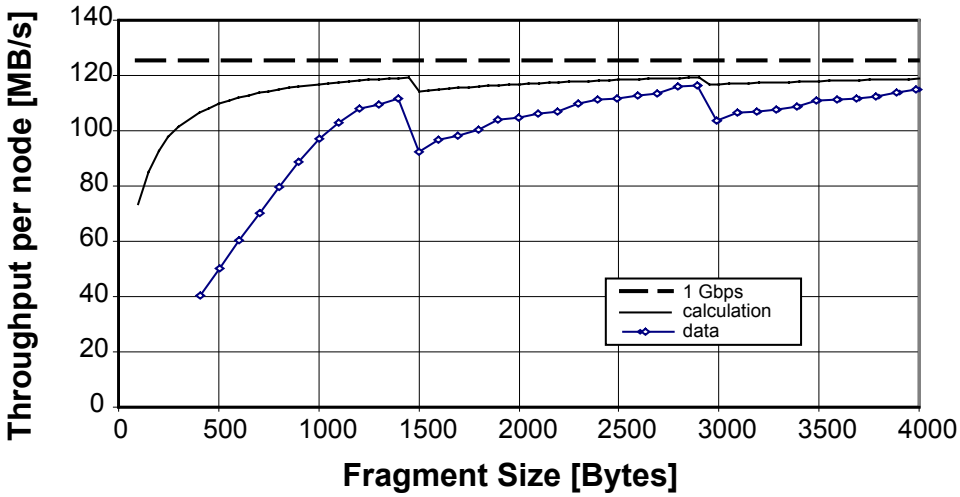


Figure B-3 Throughput per node vs. size for the 15×15 Gigabit Ethernet EVB.

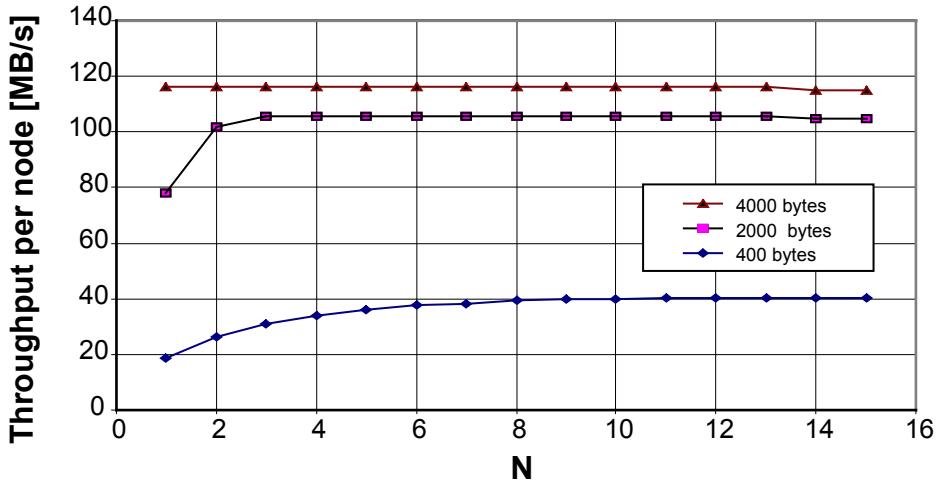


Figure B-4 Throughput per node vs. N×N for a Gigabit Ethernet EVB for different fragment sizes.

B.2.2 A Multi-chassis Switching Network for a 48×48 Event Builder

This study was made to get experience in constructing a large switch through cascading several smaller ones, so that this option can be followed if economically attractive at time of purchase. One issue is that this requires multiple paths between the switches, which is normally not allowed in Ethernet. Another issue is to check how efficiently a multi-chassis switching network could operate with event building traffic.The studies were done with Fast Ethernet (100 Mbps) in order reduce the cost of the test setup.

The configuration tested is a 48×48 event builder in a folded-Clos topology (see Figure B-5). In a folded-Clos topology, the number of ports in the intermediate layer is a factor two less than in a Clos topology. This optimization is possible by mixing RUs and BUs on the same switch and hence utilising the full-

duplex inter-switch links in both directions for event data traffic. The host nodes are connected to three BATM Titan T5 [B-3] 48 port Fast Ethernet switches. The intermediate switching layer is provided by 36 ports on two Fast Ethernet modules of the FastIron switch.

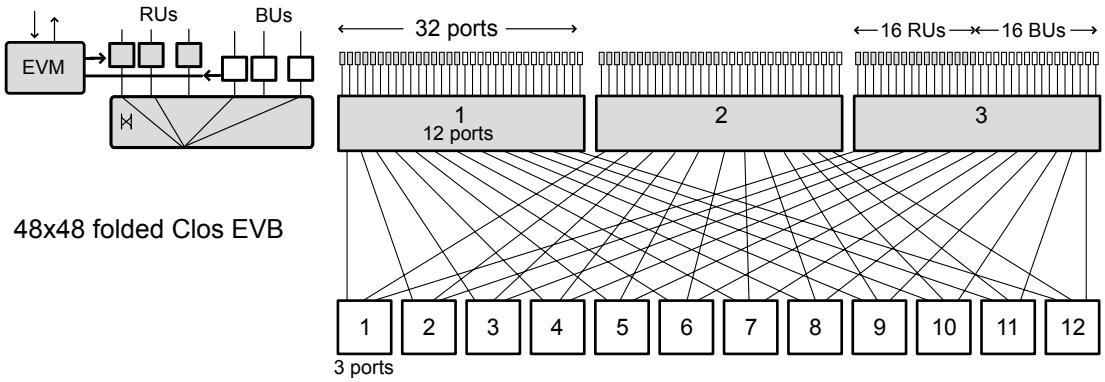


Figure B-5 The 48×48 folded-Clos Event Builder network.

The intermediate layer introduces multiple paths between RUs and BUs, which are not allowed with layer 2 switching. However, using VLANs (IEEE 802.1q), the 36 ports of the intermediate layer are divided into 12 independent virtual switches of 3 ports each. The BATM switches are connected to all 12 VLANs, each one associated to one of the inter-switch links. In this way, every node can choose the intermediate switch by setting the VLAN-Id in the Ethernet packet header. For the tests, every packet is sent to a randomly chosen intermediate switch.

The performance for this event builder is shown in Figure B-6. The throughput per node is measured for configurations ranging from 3×3 to 48×48 for a number of fixed size event fragments. It can be seen that the throughput per node for a fragment size of 2 kB is 11.5 MB/s, which is 90% of the wire speed. The performance for variable fragments sizes (generated according to a log normal distribution with an average of 2 kB and an rms of 2 kB) is slightly reduced to 11.0 MB/s. Furthermore, the performance scales with the size of the configuration. The rate of retries required due to a lost packet was low (in the order of 1 Hz).

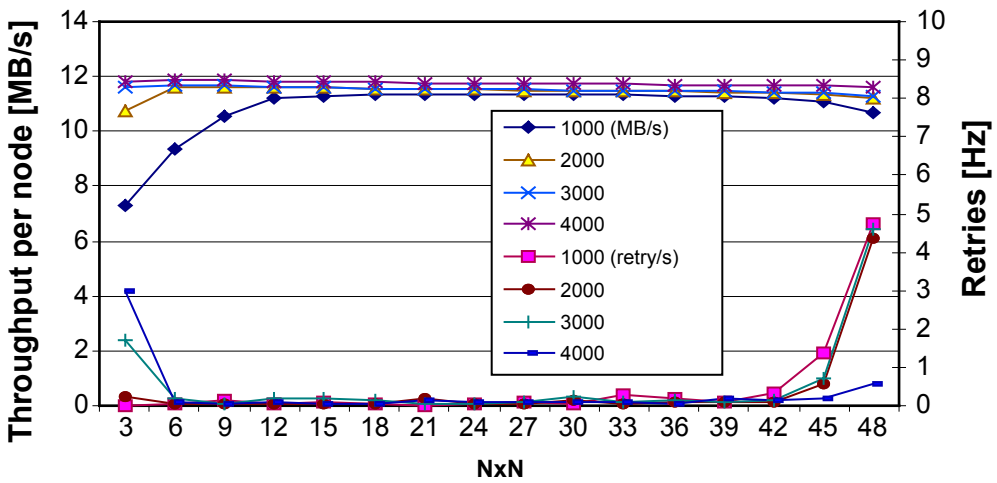


Figure B-6 Throughput per node vs. N×N for the 48×48 folded Clos EVB.

The conclusion is that it is possible to construct composite switching networks for Layer-2 frame switching out of smaller switches if they support VLANs. It also seems possible to achieve high efficiency, when using such networks for event building. More studies are however needed to verify that this holds also for Gigabit Ethernet, rather than FastEthernet.

B.3 Gigabit Ethernet TCP/IP Test-benches

This section presents the results of event building using TCP/IP over Gigabit Ethernet. Using TCP/IP rather than Layer-2 frames has a number of advantages. Foremost, it provides a reliable transport service that removes the need for the event building application to detect and deal with lost packets. TCP/IP also provides flow control and congestion control. Flow control tries to avoid buffer overruns at end points, whereas congestion control tries to prevent packet loss in intermediate nodes (switches or routers). Besides these technical merits, TCP/IP also provides a solution based on standard software. However TCP/IP is more demanding on host resources and the scaling properties for event builder traffic at high load are still unknown.

B.3.1 Point-to-point Streaming Tests

TCP/IP performance is highly dependent on NIC and host CPU performance and on the TCP/IP protocol stack implementation of the host-OS. The TCP/IP performance has been measured for a number of different NICs and hosts. One-way streaming tests were performed on host PCs running Linux 2.4. One host repeatedly sent fixed sized messages whilst another received them.

The PCs are based on the Supermicro 370DLE motherboard with a 64b/66MHz PCI-bus, equipped with a 1 GHz Pentium-III CPU. The Ethernet frames are of the standard size (MTU=1500). The standard TCP options were used, except for the sizes of the send and receive socket buffers which were set to 256 kB. Note, that by default TCP uses the Nagle algorithm which can combine several TCP segments into a packet, to increase efficiency but at the cost of latency. The one-way streaming throughput is shown in Figure B-7 as a function of the message size. It can be seen that there is a significant variation in the performance between the NICs. The AceNIC[B-4] reaches a maximum throughput of about 88 MB/s for sizes above 800 B (70% of the link bandwidth), whereas the RM674TX[1] [B-5] NIC reaches about 115 MB/s for sizes above 2 kB (92% of the link bandwidth).

The tests were repeated with the NICs configured to use Jumbo frames (MTU=9000). These results are compared with those obtained with standard frame sizes in Figure B-8. A significant improvement can be observed.

Further point-to-point tests were performed with a two-rail system based on two AceNIC NICs per PC. The 1 GHz Pentium-III based PC used in the tests above was found not to be powerful enough to drive the two links close to the link bandwidth[2]. However, with a 2 GHz Pentium-IV Xeon based PC, an aggregate throughput of about 247 MB/s was reached using jumbo frames. Measurements with this platform are presented in Figure B-9. Here the two-rail configuration is compared with the one-rail, both for standard and Jumbo frames. Similar performance has been obtained with NICs based on the family of Intel 8254xx Ethernet controllers.

1. The RM674TX is based on the Intel 82544 Ethernet controller.
2. This corresponds to the empirical rule that 1 Hz of CPU is required per bps transmitted.

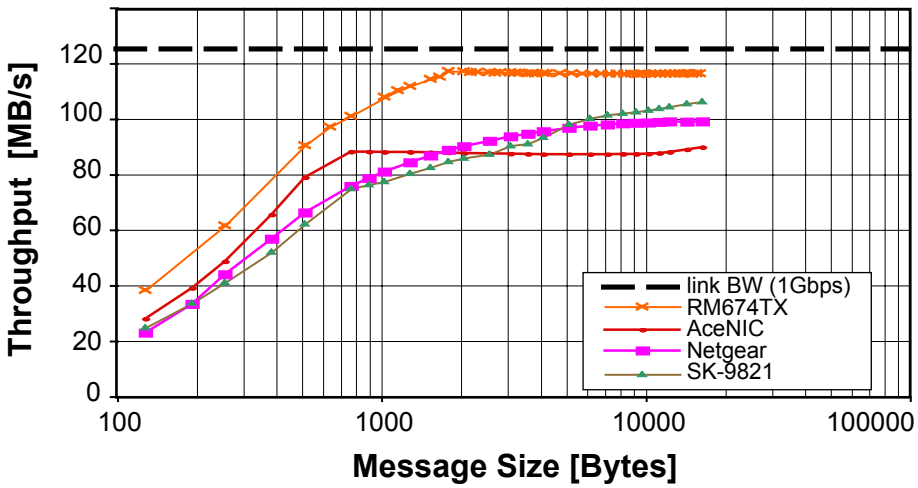


Figure B-7 Streaming TCP/IP performance for various NICs (AceNIC[B-4], RM674TX[B-5], SK9821[B-2], Netgear[B-6]). Hosts are used based on 1 GHz Intel Pentium-III CPUs and Ethernet frames of standard size (MTU=1500).

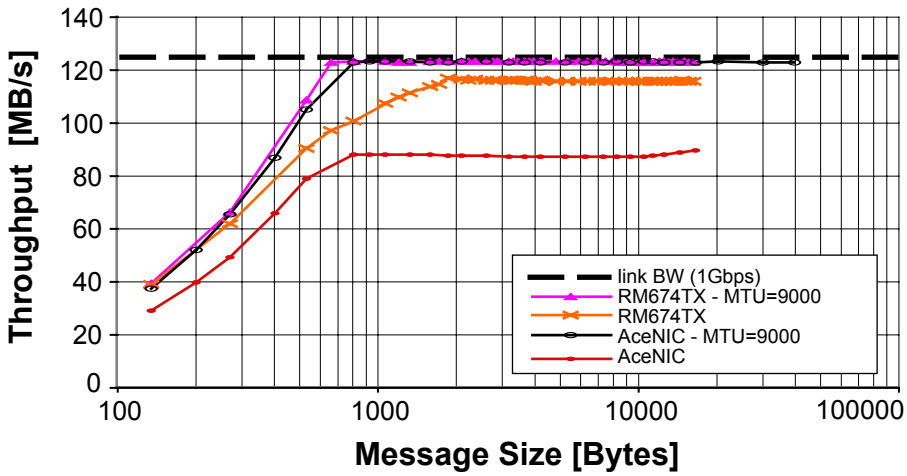


Figure B-8 Streaming TCP/IP performance for Jumbo frames compared to standard frames. Hosts with 1 GHz Intel Pentium-III CPUs are used.

B.3.2 Event Building Tests

The TCP/IP option has been evaluated using the same test-bench (see Section 6.5.2.3, "Test-bench Results for Layer-2 Frames") as was used for the studies of the event building with Ethernet Layer-2 frames. Briefly, it comprises 64 Pentium-III based PCs (750 MHz or 1 GHz) with AceNIC NICs connected to a FastIron-8000 switch. The switch model used does not support Jumbo frames. Therefore, the present studies are restricted to standard frames. The trigger is not emulated and the throughput measured corresponds to the saturation limit.

The application software running on the PCs implements the event building protocol discussed in Section 5.3.2.1, "Event Flow Protocols", using TCP socket connections to transmit data. Each instance of the RU, BU or BM application runs as a single process on its host. The select mechanism is used for the

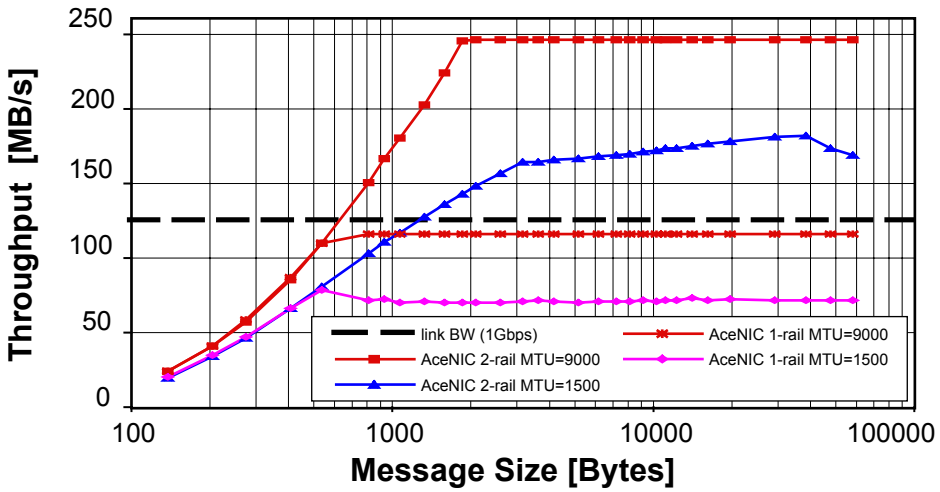


Figure B-9 Streaming TCP/IP performance for a two-rail compared to one-rail configuration with standard and Jumbo frames. The NIC used is AceNIC. Hosts with 2 GHz Pentium-IV Xeon CPUs are used.

synchronous I/O multiplexing to send and receive data via the socket connections to all remote applications participating in the event building.

The control messages are bundled for efficiency reasons. The request messages from BU to RU and the allocation or clear messages between BU and BM aggregate 8 and 16 logical messages, respectively. Furthermore, each BU builds 64 events concurrently.

B.3.2.1 1×1 EVB

As a first step, tests have been done with a simpler configuration based on just two nodes; one RU and one BU. These are connected directly, i.e. without a switch between them. This 1×1 Event Builder does not include a BM. The function of the BM to allocate events to BUs is anyhow not needed, as there is only a single BU.

The throughput per node versus fragment size is shown in Figure B-10, and is compared to the one-way streaming throughput discussed in the previous section. Results are given, both for normal and Jumbo frames. It can be seen that the event builder throughput is significantly lower than to the one-way streaming throughput. This holds in particular for smaller fragment sizes. It should be noted that the event building application is more complex, because communication is full duplex and the receiving of messages has to be done in at least two steps[1]. The throughput per node at a size of 16 kB is about 76 MB/s for standard frames and 95 MB/s for Jumbo frames.

B.3.2.2 31×31 EVB with BM

Studies were done with the 31×31 EVB configuration, including a BM. The throughput per node as a function of the fragment size is shown in Figure B-11. For a size of 16 kB the achieved throughput per node is about 75 MB/s. This throughput corresponds to 85% of the measured one-way streaming throughput (88 MB/s). It should be noted that due to this limitation of the host and/or NIC, the switch load is only about 60%.

1. the header including the size of the message must be read before the variable sized message body

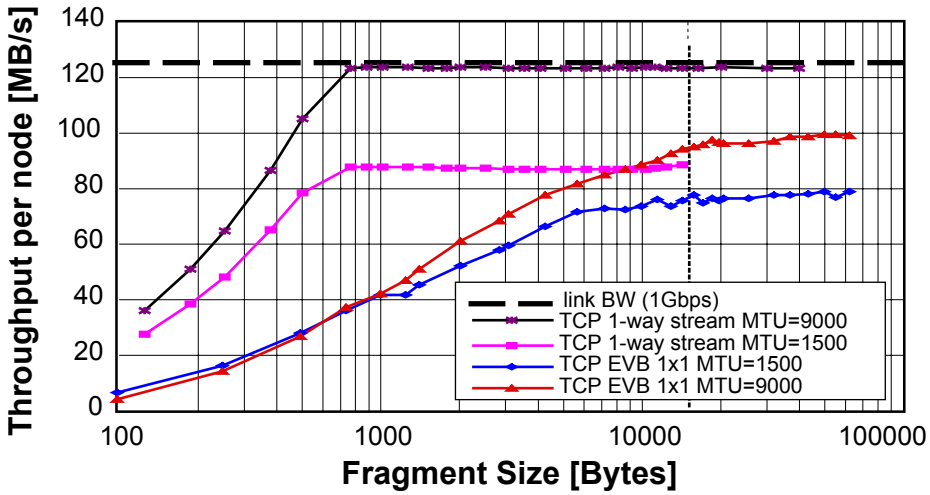


Figure B-10 Throughput per node vs. fragment size for a TCP/IP 1×1 EVB. The vertical dotted line indicates a fragment size of 16 kB.

The size of the Event Builder is varied from 3x3 to 31x31 in Figure B-12 for the nominal case of variable size fragments with an average of 16 kB and an rms of 8 kB. It can be seen that the throughput per node scales approximately with the size of the configuration and a value of about 75 MB/s is achieved.

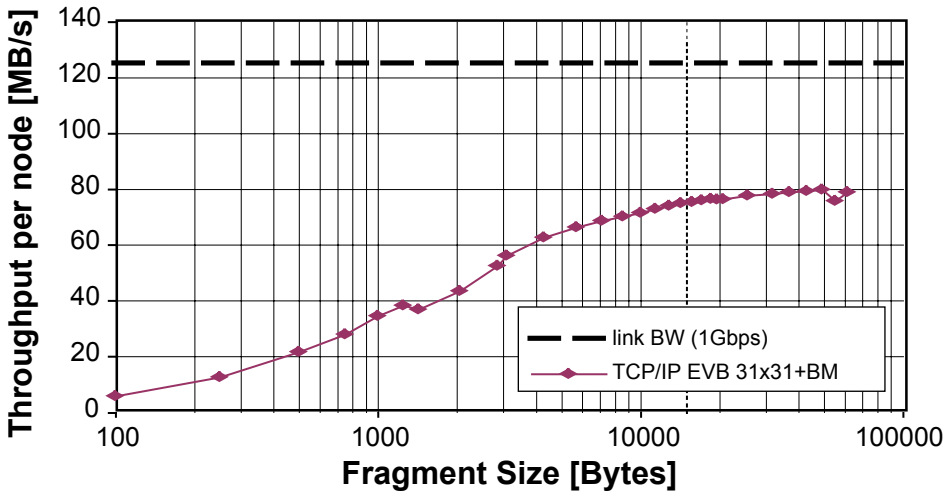


Figure B-11 Throughput per node vs. fragment size for a TCP/IP 31×31 EVB with BM.

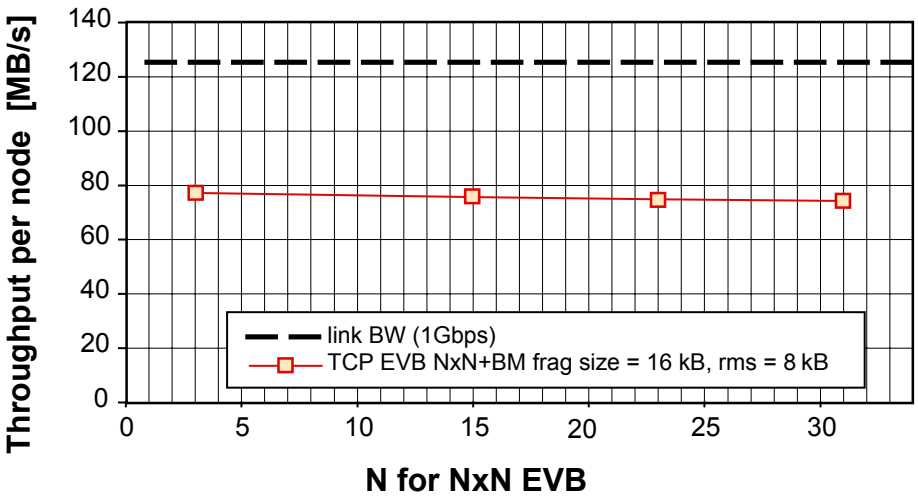


Figure B-12 Throughput per node vs. N×N for a TCP/IP EVB. The fragment sizes are generated according to a log-normal distribution with an average of 16 kB and an rms of 8 kB.

B.4 Simulation of Myrinet Event Builder

A successful simulation of the EVB is one that aids the understanding of the system's functions and also predicts convincingly the behaviour of the system in different contexts and with different parameters. Given that a full prototype of the final system cannot be built due to the resources currently available an important element provided by simulation is the expected behaviour of the full final system once all the components are combined.

In what follows, after a short introduction to the simulation package, an overview of the modelling of single EVB components and EVB networks is given. Data and simulation results are compared and a forecast of the EVB performance with LANai10 based hardware is made. A forecast for a full sized RU Builder and FED Builder is presented in Section 6.7.1.

B.4.1 Simulation Package

The Ptolemy package [B-7], developed at the University of California at Berkeley, provides a framework for the simulation of complex and heterogeneous systems. For the implementation of the EVB simulation the discrete event (DE) domain was used. The DE domain in Ptolemy is designed for time-oriented simulations of systems such as communication networks. Actions are described as the exchange of "Particles" between objects. These objects, called "stars" in the Ptolemy terminology, can receive Particles over one or many ports and trigger the emission of new Particles. Stars represent functional units like the Readout Unit or the Builder Unit.

The Ptolemy environment provides a hierarchical organization of all its objects. As an example, stars can be grouped together into "galaxies", and these galaxies can communicate with other galaxies or with stars. Proceeding further, one can define a universe as a combination of stars and galaxies with no external ports. A universe represents a complete simulation model. In the case of the Event Builder model described later, it is made of a single galaxy by interconnecting ports from different stars.

Ptolemy has been developed in C++ using object-oriented software which supports extensibility and modularity. Extensions to existing stars and creation of new stars can thus be accomplished easily with the aid of various objects and classes provided by Ptolemy. The analysis package ROOT [B-8] has been added for data analysis and for automatic job submission. It allows also to resolve the EVB traffic on the single packet scale. To reduce processing times, a front-end is being developed to utilize a computer farm and run multiple simulations in parallel.

In the EVB simulation within the DE domain, a Particle represents an event that corresponds to a change of the system state. Events can be network messages, DMA transfers, busy states etc. The DE scheduler processes events in chronological order. Each particle has an associated time stamp which is generated by the birth-giving star based on the input event that triggered the generation and the latency of the action.

In the simulation, the EVB is resolved into functional units implemented as stars which will be described in detail in the following sections. The EVB communication protocol of the different units is implemented as it is used in the real EVB system, as described in Chapter 4.

B.4.2 Simulation of EVB Components

An overview of the EVB components that have been implemented is given in Table B-3. Either the FED Builder or the RU Builder can currently be simulated. FED and RU Builder combined in a single system have not been simulated. Since the RU will have sufficient memory to be decoupled from statistical fluctuations of the FED Builder the full system's performance is governed by the least performant component of the FED Builder / RU Builder system. Figure B-13 shows a schematic of the structure of the simulation with its components and the relation between them.

Table B-3 Overview of data sources and sinks in the simulation.

Schema	Source	Sink	Other
FED Builder	FRL	RUI	Trigger
RU Builder	RU	BU	EVM

B.4.2.1 Data Sources

The data chain begins with the RU or FRL respectively. Components in the readout chain prior to the RU/FRL are not implemented. The data source is modelled with a random generator of fragment sizes, the probability distribution can be freely chosen and different for each RU/FRL. Predefined analytical functions (boxcar, normal, log-normal as defined in Table B-4) or histograms can be used to define the probability density function (pdf).

The flexible design of the data source allows to study various scenarios of data conditions. It is also possible to vary the correlation of fragment sizes between data sources from 0% to 100%. The correlation is defined as the linear correlation coefficient *r*:

$r = S_{xy} / \sqrt{S_{xx} S_{yy}}$, with $S_{xy} = \sum (x - \bar{x})(y - \bar{y})$, x and y being fragment sizes of a pair of data sources.

Besides correlation, other aspects of the study of data conditions were unbalanced fragment sizes and pathologically large fragments. The imbalance of input was parametrized with the imbalance ratio R defined as $R = \bar{L}_{max} / \bar{L}_{min}$, with L_{max} the maximum average fragment size and L_{min} the minimum average fragment size of an ensemble of data inputs.

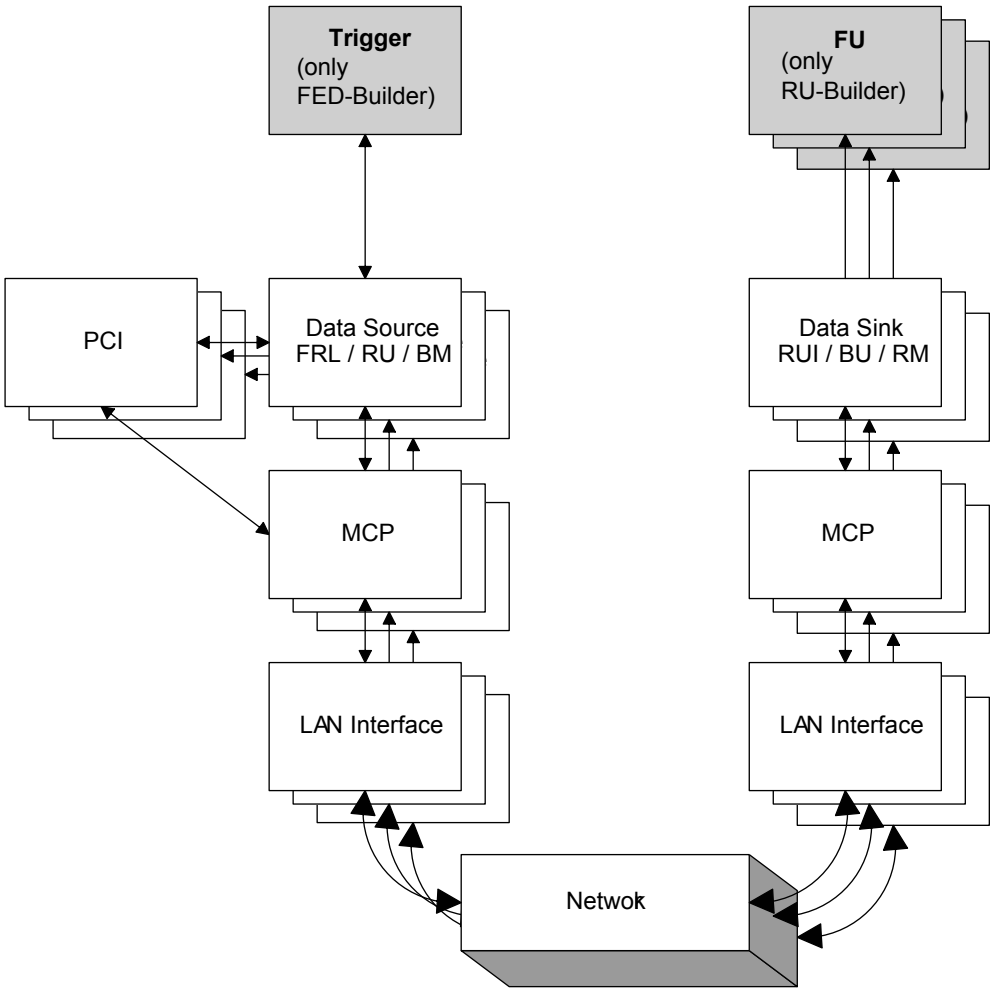


Figure B-13 Schematic of RU/FED Builder simulation.

Table B-4 Probability density functions and their definitions.

pdf	Definition
Boxcar	$y = c[H(x-a)-H(x-b)]$ with $H(x) = \begin{cases} 0;x<0 \\ \frac{1}{2};x=0 \\ 1;(x>0) \end{cases}$
Normal	$P(x)dx = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/(2\sigma^2)} dx$
Log-normal	$P(x)dx = \frac{1}{Sx\sqrt{2\pi}} e^{-(\ln x - M)^2/(2S^2)}$, with $\mu = e^{(M+S^2/2)}$ $\sigma^2 = (e^{S^2}-1)\cdot e^{(S^2+2M)}$

Pathologically large fragments, or “jumbo fragments”, where simulated by injecting a single large fragment, with a size of 200 kB, at a defined time into one data source and observe the behaviour of the system.

B.4.2.2 FRL Model (FED Builder)

The FRL represents the first element in the data flow of the FED Builder simulation. The FED level is implicit in the data source and the merging of FEDs can be emulated by providing adequate fragment size distributions. The PCI bus is modelled with priority queues and arbitration for different access modes, and comprises detailed timings for set-ups and transmission of data. The interface to the network is done with the NIC model described in Appendix B.4.2.9. No network traffic shaping is done in the FED Builder case, but maximal transfer units (MTU) can be employed.

The FRL is connected to a trigger (described in Appendix B.4.2.4) and can throttle in case it is not capable of keeping up with the trigger rate. Analogous to the real system the memory is limited and is settable with a parameter.

B.4.2.3 RUI Model (FED Builder)

The RUI is collecting the data fragments created by the FRLs. It assembles super-fragments and forwards them to the RUs. The collecting of fragments and the assembling of the super-fragments are modelled in the simulation. The RUI is implemented without a detailed modelling of the PCI bus and it is assumed that forwarding of the super-fragments to the RUs is negligible compared to the time needed to collect fragments. The RUI model is used to study the performance of super-fragment building and the memory usage.

B.4.2.4 Trigger Model (FED Builder)

The trigger model in the simulation is a simple object which emits trigger signals with a Poisson time distribution with adjustable mean. The trigger can be throttled by external objects and monitors the efficiency. The trigger efficiency is defined as $\varepsilon = N_{acp}/N_{tot}$ with N_{acp} the number of issued trigger signals in the unthrottled state and N_{tot} the total number of trigger events.

The trigger is a global module which connects to all FED Builders in case of multiple FED Builders.

B.4.2.5 RU Model (RU Builder)

The RU model features similar components as the FRL. It includes the PCI model and the NIC model, but it does not connect to a trigger. The data source is modelled in the same way as in the FRL. Since super-fragments are injected independently of the trigger, it is assumed that the FED Builder, i.e. RUI, saturates the RU. Limitations in the performance of the RU Builder are thus not caused by the FED Builder, but by genuine RU properties and represent the saturation limit.

The forwarding of super-fragments to the network is done via the NIC. The same protocol as in the real hardware has been implemented. The packing of super-fragments into MTUs and the Barrel Shifter protocol are driven by the NIC.

The event building protocol has been implemented. The RU can operate with and without EVM[1]. Studies without EVM have been useful to verify the simulation with data obtained from the test-bench.

B.4.2.6 BU Model (RU Builder)

Similar to the RUI model the PCI bus is not included in the simulation of the BU. It is assumed that the transmission time of super-fragments from the RU to the BU will be long compared to the transmission of super-fragments from the receiving NIC to the BU and that the PCI will not severely limit the transmission of control messages.

The BU can be operated with and without EVM. When the EVM is absent, the BU will act as a pure sink. As for the RU the full event building protocol has been implemented. Optionally the assembly and shipping of complete events to the FUs can be invoked. The resources and the packing factor of control messages in the BU are configurable.

B.4.2.7 EVM Model (RU Builder)

The EVM model consists of an RM, BM, trigger and an RCN. The RM associates Event_IDs with trigger events and distributes the information to the RUs. The RM maps each trigger to an available Event_ID and sends the information with minimal latency to all RUs. The BM serves newly allocated Event IDs to BUs according to their requests. An allocated Event ID becomes available again as soon as it is released by the BU. The RCN can be implemented as a separate network, but by default the Myrinet RU Builder switch is used. In a configuration with EVM the RM is connected to the BU side and the BM to the RU side.

The EVM is configurable. Parameters include the event table size, packing factor of control messages, RCN bandwidth and trigger rates.

B.4.2.8 FU Model (RU Builder)

The FUs are modelled as a simple extension to the BU in case an EVM is present. The FUs are connected to the BU over an idealized network. Each FU has settable processing time distributions and memory size. Typically eight FUs were connected to one BU.

B.4.2.9 NIC Model

Before data can be send to or received from the network they have to be copied from the host memory into the memory of the NIC or vice versa. Thus, the NIC model is included in all models which are connected to the Myrinet network: FRL, RUI, RU, BU and EVM.

The copy process uses both DMA and programmed I/O modes to transfer data via the PCI bus. The latency of the transfer process is modelled in detail taking into account the finite memory bandwidth of the PCI bus, its usage, and processing times. In case of the RUI and the BU models, the communication between NIC and RUI/BU is processed directly, bypassing the DMA and Programmed I/O modelling.

The NIC is modelled with two stars, the MCP star and the Network Interface (NI) star. The physical transport and first protocol layer are restricted to Myrinet networks in the simulation. The MCP star models the custom firmware running on the RISC processor in the NIC and the NI star simulates the actual hardware interfacing to the network. The latencies introduced by the NIC are given by the processing times for distinct tasks. These processing times are given as parameters to the NIC star and have been deter-

1. In this case the Event ID allocation is done by the BU itself, using a static algorithm.

mined from test-bench set-ups. Thus the simulation requires no “free” parameters, beyond those obtainable from point-to-point measurements.

The NIC model can simulate LANai9 and LANai10 hardware, the difference being that LANai9 has one 2 Gb/s port whereas LANai10 will provide two 2 Gb/s ports and will allow higher processing speeds relative to LANai9 hardware due to faster memory, a faster processor and an improved DMA architecture. Since the LANai10 is not yet available, the anticipated performance has been estimated by reducing the processing times by a factor two and adding a second port in the simulation.

B.4.2.10 Single Switch Elements

Implementations of single crossbar switch elements with 8 ports and 16 ports have been made by others [B-9], corresponding to Myrinet Xbar8 and Xbar16 switches. A round robin token arbitration scheme has been added. The implementation follows closely the Myrinet network technology using wormhole routing, back-pressure flow control and small slack buffers as described in detail in Appendix A.2.1.

B.4.3 FED Builder Simulation

The FED Builder is the first stage of the event building process. Fragments are supplied by one or two FEDs to one FRL and the data of eight FRLs are assembled into a super-fragment in a RUI. The simulation integrates the data source in the FRL-model, which allows to study the dependency of the performance on various data conditions.

The interconnect between FRLs and RUIs is formed by one or two 16 ports crossbar switches for LANai9 and LANai10 hardware, respectively. Hence in the LANai10 case, two independent rails are available. The FRL and RUI models connect via the NIC model to the network. The trigger model is employed to provide trigger signals to the FRL model. The FRL can throttle the trigger if the rate is exceeding its capacity.

The FED Builder operates in a pure push manner, incoming fragments on the FRL side are pushed to their destination on the RUI side and no traffic shaping as in the case of the RU Builder is employed. The switch utilization can be improved dividing fragments into packets of a maximum size (MTU), so large fragments get divided into multiple smaller packets. This reduces the effect of output port blocking by large fragments, since other senders can deliver their packets interleaved and continue with the next destination.

The throughput of a simulated LANai9 FED Builder without MTUs for different distribution types is compared to test-bench data in Figure B-14. The distributions differ in width, while the mean is fixed to 2 kB and the pdf is a log-normal. In general a good agreement between data and simulation is observed.

Starting from the good description of the current test-bench hardware different scenarios where simulated to investigate the influence of data conditions, the performance of LANai10 hardware, and the usage of MTUs. The fundamental “figure-of-merit” is the maximal trigger rate the FED Builder can accept without throttling the trigger.

The trigger efficiency was scanned for different trigger rates to establish, that the maximal trigger rate is determined by the saturation throughput divided by the average fragment size. The result is shown in Figure B-15 for LANai9 and LANai10 hardware with and without the usage of MTU and a log-normal fragment size distribution with a mean of 2 kB and an rms of 2 kB. The trigger efficiency starts to degrade

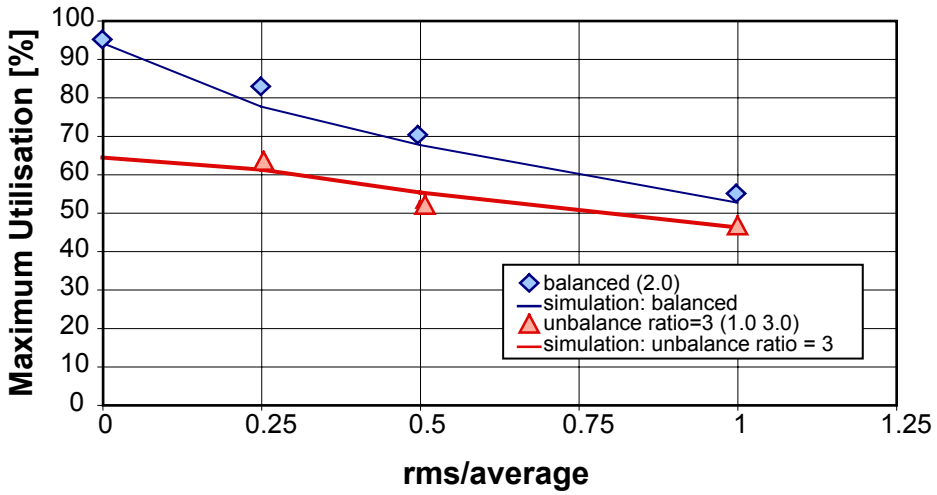


Figure B-14 Throughput for different distribution types for data and simulation.

at a trigger frequency which coincides well with the expected frequency from the simulation of the maximum saturation throughput (discussed below and also shown in the plot).

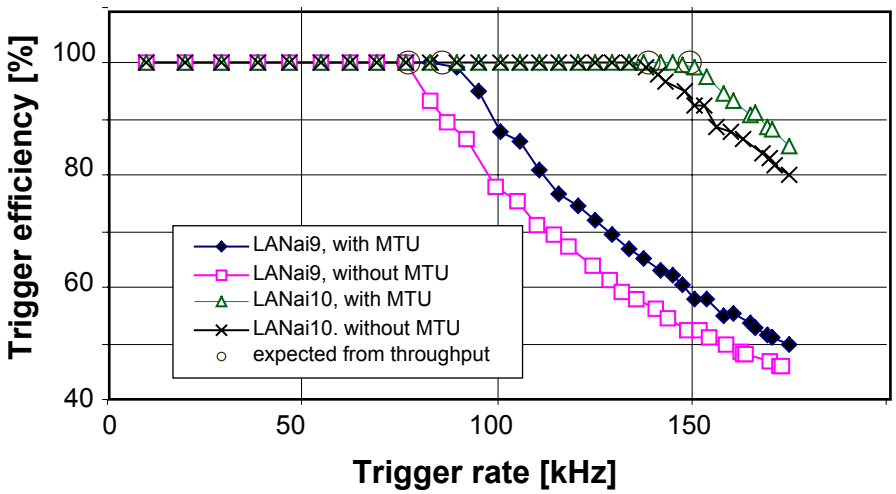


Figure B-15 Trigger efficiency vs trigger frequency for different hardware configurations. The open circles mark the expected maximal trigger rate from the saturation throughput.

The influence of having multiple FED Builders in the system has been studied with a system of 64 single 8×8 FED Builders corresponding to the full system. In Figure B-16 the trigger efficiency obtained with this configuration is plotted as a function of the trigger frequency for LANai10 hardware. It is observed that the system is fully efficient up to a trigger rate of about 150 kHz. This shows that the maximum trigger rate is not very sensitive to the number of FED Builders in the system. The trigger efficiency of a system of multiple FED Builders is not simply the product of the single FED Builder efficiencies, because the throttling of the trigger affects all FED Builders amd hence the systems are coupled.

The dependence of the throughput on the mean and rms of the log-normal distribution is shown in Figure B-17 for LANai9 and LANai10 hardware with and without the usage of MTUs. In this scenario, all inputs have the same fragment size distribution and are uncorrelated. The plot shows the clear per-

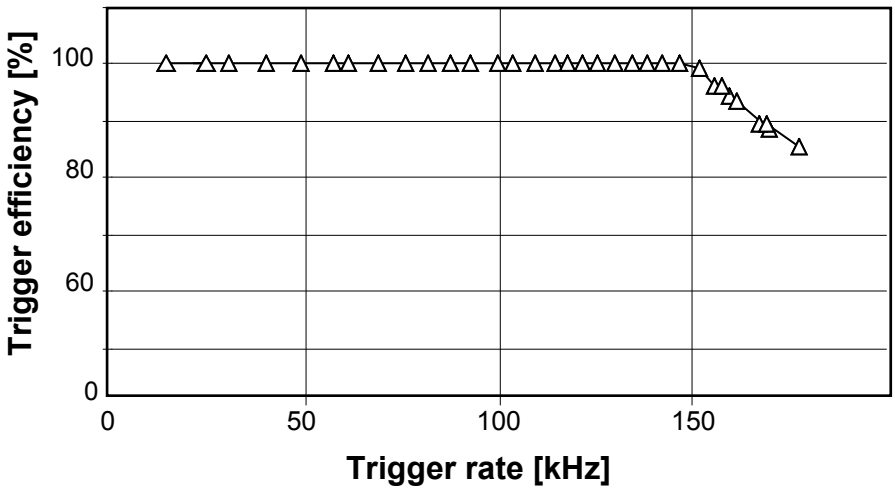


Figure B-16 Trigger efficiency vs. trigger rate for 64 8×8 FED Builders and a log-normal fragment size distribution with a mean of 2 kB and a rms of 2 kB.

formance gain when the two-rail LANai10 hardware is employed, and also demonstrates the beneficial effect of using MTUs.

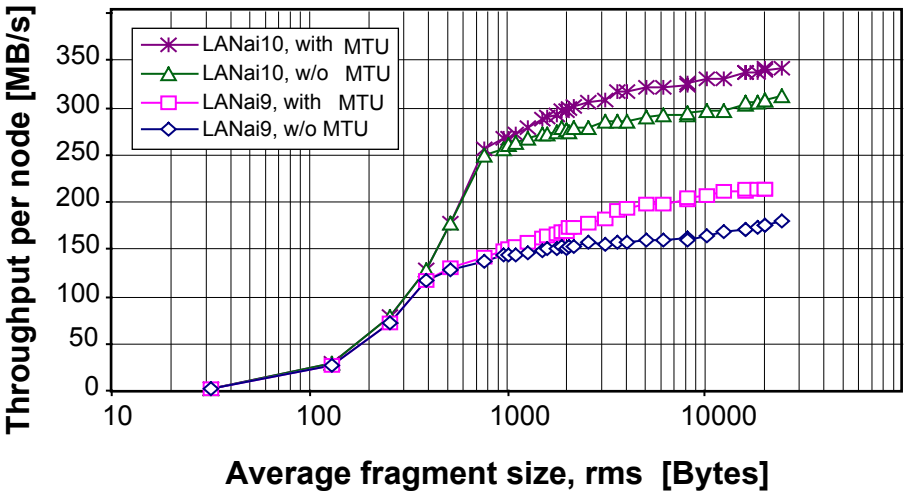


Figure B-17 Dependence of the throughput per node on the mean and RMS of the input distribution to the FRLs. The rms is is set equal to the mean of the log-normal distribution.

The performance of the FED Builder is sensitive to the fragment size distributions of the FRL. The case where a set of FRLs send more data on average than the rest of the FRLs was studied by varying the imbalance parameter for a configuration with 4 FRLs above and 4 below the average (4-4), and another configuration with 1 above and 7 below the average (1-7). The result for LANai9 and LANai10 hardware with and without MTU is shown in Figure B-18.

For LANai10 hardware, the target throughput of 200 MB/s is sustained for the “4-4” configuration up to R values of 3.5, while for the “1-7” configuration the limit is reached at R=1.9. Although the “1-7” configuration seems to be more sensitive to R, it should be noted that in terms of the actual maximum aver-

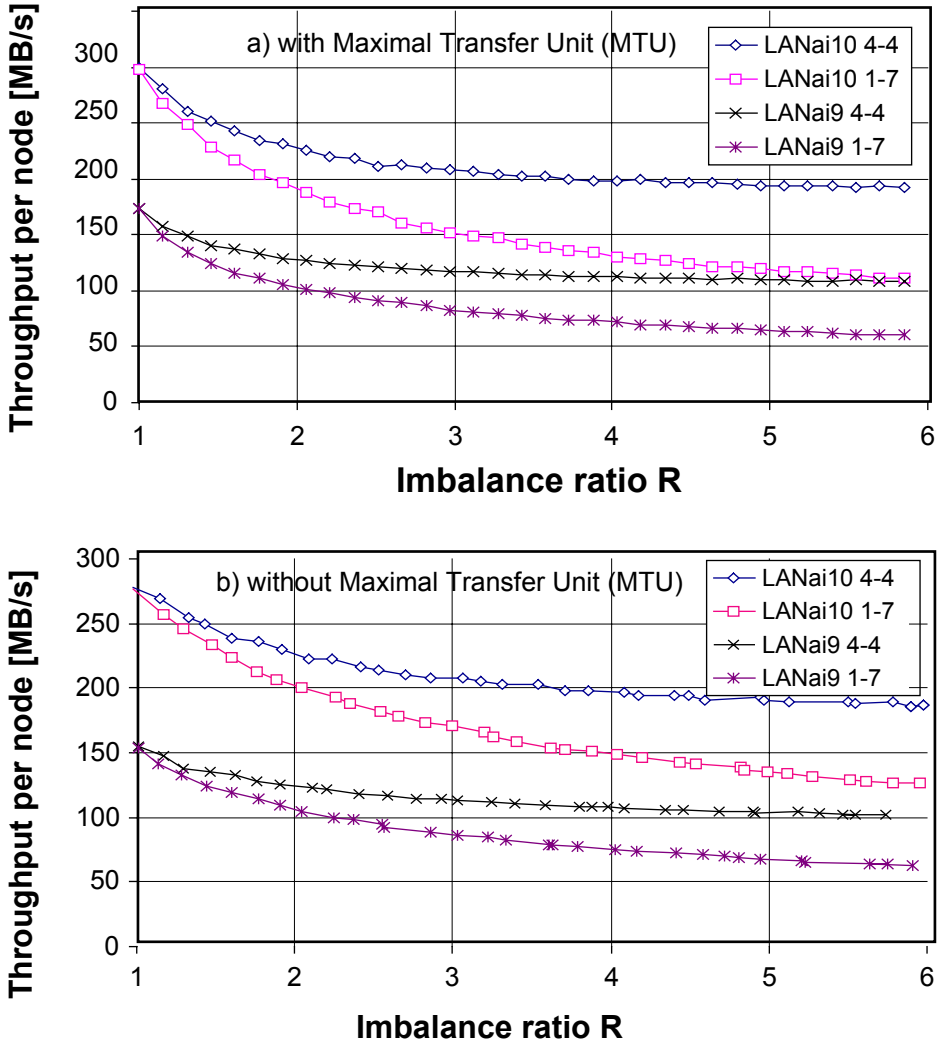


Figure B-18 Throughput vs. the imbalance parameter R, shown for LANai9 and LANai10 hardware with and without usage of MTUs for both 4-4 and 1-7 scenarios.

age fragment size the values do not differ by much, namely 3.1 kB for the “4-4” configuration and 3.4 kB for the “1-7” configuration.

The correlation of fragment sizes between different FRLs is very likely to occur, since the data volume of many detector components is determined by the multiplicity of the underlying minimum bias events. The sensitivity of the FED Builder performance has been studied by varying the linear correlation coefficient between the FRL data sources for a log-normal distribution with an average of 2 kB and an rms of 2 kB. The results for LANai9 and LANai10 hardware with and without MTU is shown in Figure B-19. The correlation of fragment sizes has a negligible influence on the performance over the full range of the correlation coefficient *r*.

The behaviour of the FED Builder has been studied for the special case that one FRL sends a very large fragment, so called “jumbo fragment”. The FRL will have enough memory (1 MB) to buffer and send this “jumbo fragment”. However, the input port of the RUI will not be accessible by the other FRLs during the transmission of the jumbo fragment and thus a serious degradation of performance might occur if the other FRLs are stalled and have to wait for the input to the specific RUI to become available. The “jumbo fragment” scenario was studied for a one rail LANai9 and a two rail LANai10 network with and without

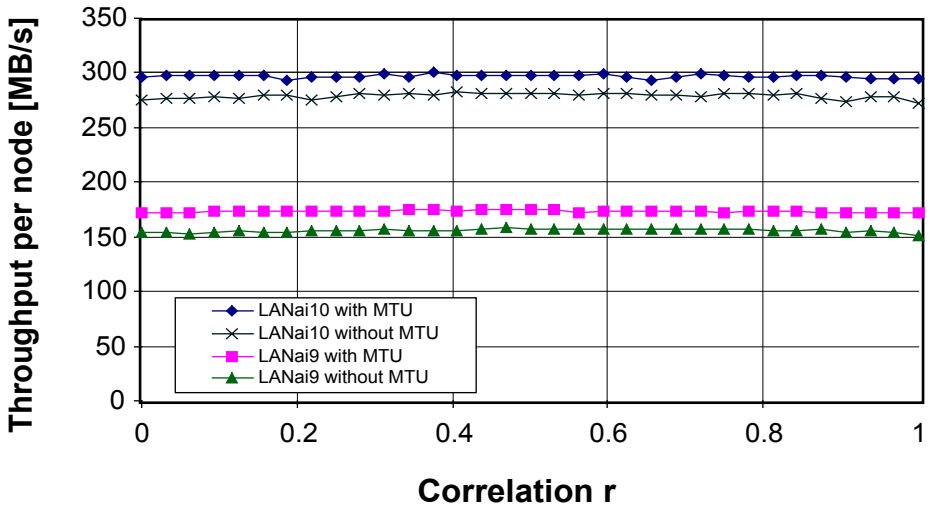


Figure B-19 Throughput vs. correlation coefficient between the FRL data sources for a log-normal distribution with 2 kB mean and 2 kB rms.

usage of MTU. The size of the "jumbo fragment" was set to 200 kB. The average throughput of all FRL nodes as a function of time is monitored. The injection time of the "jumbo fragment" is at t = 10 ms in the FRL. The fragment has to be transported to the NIC memory and is buffered there until it is sent off to the network. The storage period in the NIC depends on the NIC memory size and utilization, and on the drain speed. The time the fragment needs from the creation in the FRL until it reaches the network is of the order of 5ms for a one-rail LANai9 builder and only 2.5 ms for a two-rail LANai10 builder since the latter provides roughly twice the bandwidth of the former.

The behaviour of the throughput is shown in Figure B-20 for LANai9 and LANai10 hardware, with and without MTUs. A substantial degradation of throughput is seen only for LANai9 hardware in the case that no MTUs are used (Figure B-20 a)). At transmission time (t=14ms) the throughput for the nodes not transmitting the "jumbo fragment" drops to zero, while one node (full circle) uses the switch exclusively. If MTUs are used, the switch output port is no longer blocked by one large fragment but other fragments can interleave (Figure B-20 b)). The behaviour is not critical for LANai10 even in the case without MTU. Here, the second rail prevents a blocking of the output ports (Figure B-20 c)). The nodes not transmitting the "jumbo fragment" can still use half of the bandwidth of the FED Builder, while the node with the "jumbo fragment" uses one link exclusively and the second one for fragments of "normal" size.

A microscopic view of the incident is shown in Figure B-21 for LANai9 and LANai10 hardware. The plot focuses on the FRL which produces the "jumbo fragment". Here the event numbers in the system are plotted versus the time of creation in the FRL. The dots correspond to the creation of an event fragment associated to an event number in the FRL. The network is blocked without the employment of MTUs. Only one fragment, the jumbo fragment, is being transmitted through the switch. No other fragments are being received, which can be seen by the absence of points during the super-fragment transmission time at around 14ms for LANai9 hardware. If MTUs are used, the super-fragment is no longer exclusively using one output link, but the segmentation allows other senders to deliver their packets. This is shown in the plot, where the dots with constant event number are the MTUs belonging to the "jumbo fragment" (at t=10ms) being created in the FRL, while the other dots clearly show the progression of the FED Builder traffic even during the transmission time at about t=14ms.

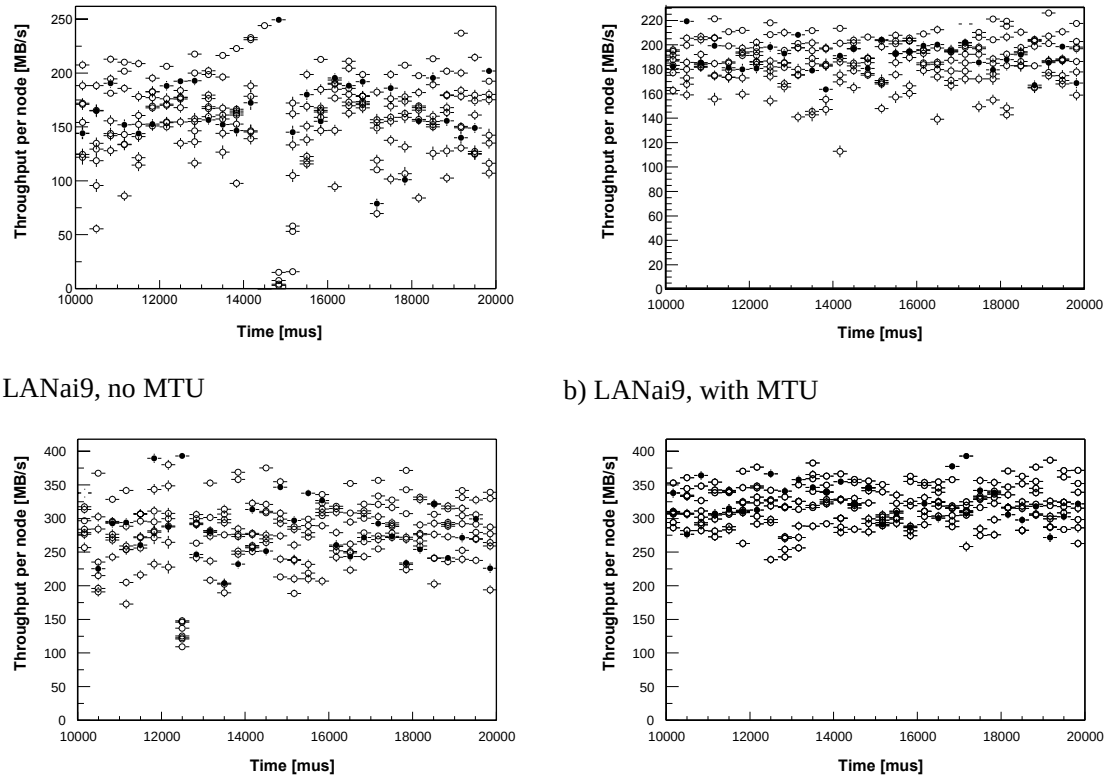


Figure B-20 Throughput per node as a function of time for 8 FRL nodes. A “jumbo fragment” is injected at t=10ms and propagates to the network at about t = 15ms (LANai9) and t = 12.5ms (LANai10). The full circles show the throughput of the FRL node with the “jumbo fragment”, the empty circles show the 7 other FRL nodes.

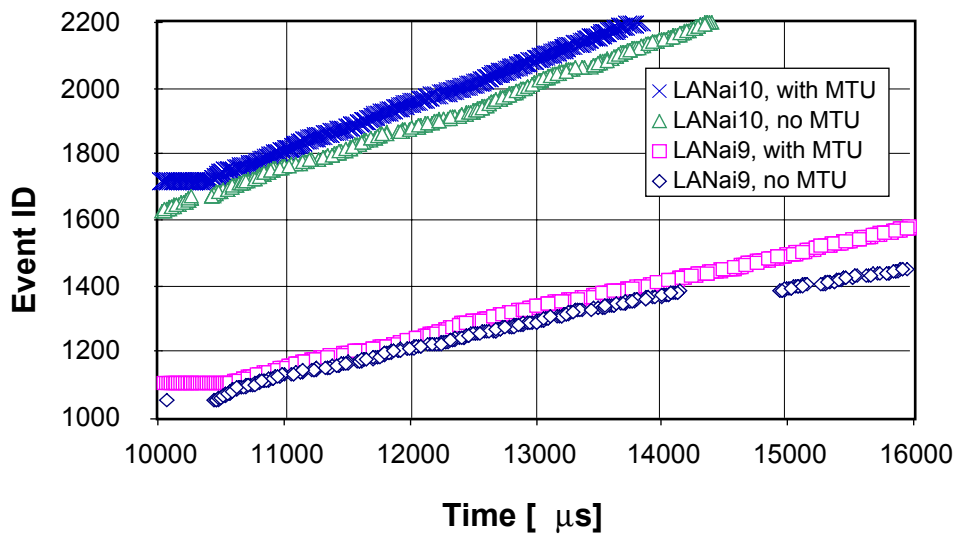


Figure B-21 Event numbers in the system vs. time. A “jumbo fragment” is injected at t=10 ms.

B.4.4 RU Builder Simulation

The scope of the RU Builder simulation is to verify results obtained with test-bench set-ups and to predict on the basis of the reliable description of the test-bench data the performance of the final system with LANai10 hardware.

The simulation of the RU Builder closely follows the set-up of the test-benches (see Chapter 6.5, "RU Builder") as well as the layout proposed for the final system (see Chapter 4.3, "RU Builder"), which includes the EVM. The FED Builder has not been included in the simulation of the RU Builder. The data source is integrated into the RU model. As explained above, the coupling between both builders is low, so that each subsystem can be studied independently of the other as it was done with the test-benches.

Extensive studies of the throughput dependence on the fragment size allowed to verify the detailed model of the hardware in the simulation. Especially the preparation of packets in the NIC, which has to keep up with the barrel shifter cycle, is a time critical process since it puts the NIC to the limit of its processing power and internal memory bandwidth. A detailed description of the internals of the NIC can be found in Chapter A.2, "Myrinet Technology and Products".

Special attention has been paid to reproduce the measurement which can be seen as a benchmark test for the simulation. The result is shown in Figure B-22 where the simulation results are compared with the data obtained from the test-bench operating with 32 RUs and 32 BUs, and no EVM present. In the case that no EVM is present it is assumed that the BUs will always have allocated Event_IDs available. The two simulation curves differ in the assumptions of the memory bandwidth internally available to the NIC processor. If multiple DMAs are executed, the remaining memory bandwidth available can limit the execution time of the program running on the NIC processor. To model this, the processing times were increased depending on the number of ongoing DMAs. The lower curve implements this “memory choking” effect, while the upper curve assumes that sufficient bandwidth is available. The data points below a fragment size of 2 kB and above 6 kB are well described by the simulation without memory choking, while the data points in between are better described by the model with “memory choking”. Since the data-points are enclosed in between the two bands, the bands can be seen as the level of uncertainty in the simulation. The operation point of the RU Builder will be at 16 kB corresponding to the average size of a super-fragment, where the influence of the memory choking is insignificant.

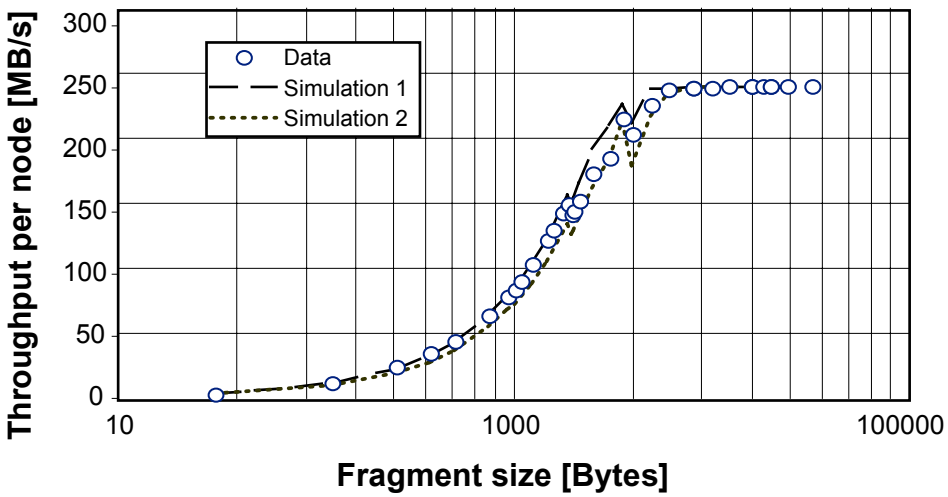


Figure B-22 Throughput per node as a function of the fragment size for a 32×32 network and LANai9 hardware. The bands represent the simulation result for two different performance assumptions. The symbols show the data points obtained with the prototype.

The final system will have an EVM with an RM on the BU side and an BM on the RU side. The effect of the addition of an EVM has been studied. The EVM model is described in Chapter B.4.2.7, "EVM Model (RU Builder)". The addition is expected to affect the performance of the RU Builder since firstly one port of the builder is used for the EVM, and, secondly, the distribution of Event_ID, or rather a shortage of them, can delay the sending of fragments and thus degrade the performance. The first cause of degradation is inherent to the design and irreducible. The effect of the second cause is dependent on the number of available resources and the servicing times.

The simulation is compared to data obtained with the test-bench set-up in a configuration of 31 RUs, 31 BUs and one BM and no packing of control messages (see Figure B-23). The BM serves newly allocated Event_IDs to BUs according to their requests. An allocated Event_ID becomes available as soon as it is released by the BU.

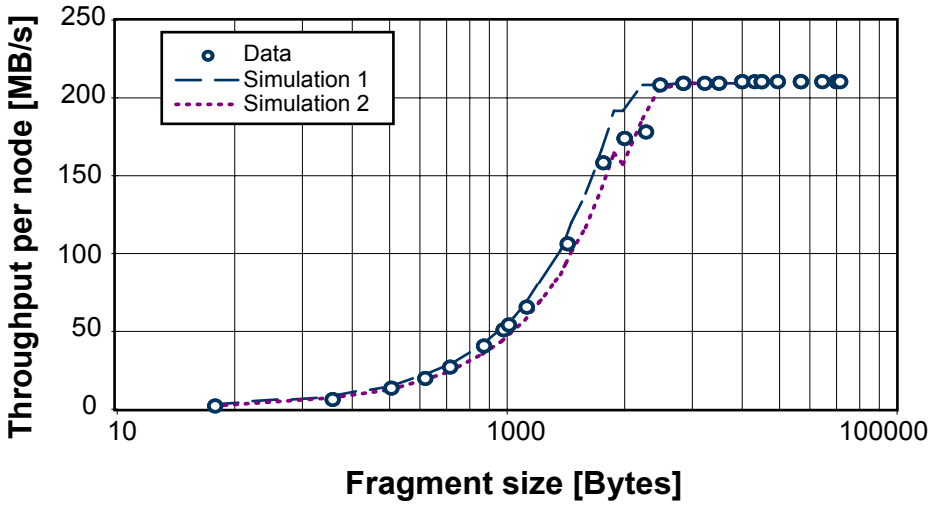


Figure B-23 Throughput per node vs. fragment size for a 31×31 RU Builder with BM. Simulation results and data are compared.

The performance of the final 64×64 system has been simulated based on the parameters describing the 32×32 test-bench and is shown in Figure B-24 with BM. The scaling of the system is as expected for a barrel shifter. The performance with BM is slightly better than for the 32×32 configuration since the fraction of ports used to transfer event data is 63/64 compared to 31/32. The performance of the anticipated LANai10 hardware is shown in the same plot. Compared to the FED Builder with LANai10, only one rail is used. The plateau is now attained already at smaller fragment sizes because the NIC is now faster and has time to accommodate more messages in one barrel shifter packet.

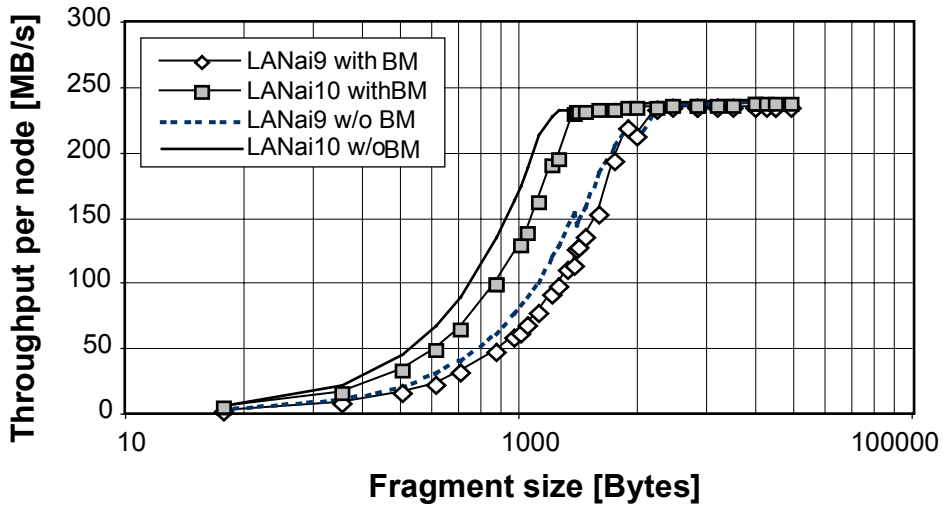


Figure B-24 Throughput vs. fragment size for a 63×63 RU Builder with BM for LANai9 and LANai10 hardware.

B.5 References

B-1 FastIron switch from Foundry Networks, see http://www.foundry.com

B-2 SK9821 from SysKonnect, see http://www.syskonnect.com

B-3 Titan switch from BATM, see http://www.batm.com

B-4 AceNIC from Alteon Corp., see http://www.alteon.com

B-5 RM674TX from Ramix Corp., see http://www.ramix.com

B-6 Netgear NIC from Netgear Corp., see http://www.netgear.com

B-7 The Ptolemy Project, see http://ptolemy.berkeley.edu/

B-8 R. Brun and F. Rademakers, “ROOT - An object oriented data analysis framework”, Nucl. Inst. Meth. in Phys. Res. A 389 (1997) 81-86;
ROOT, see http://root.cern.ch/

B-9 J-P. Dufey et al.,”The LHCb Trigger and Data Acquisition”, IEEE Trans. Nucl. Sci., 47 (2000) 86;
B. Rensch, “A Few Results from EvtBldg-Sim”, Niels Bohr Institute, Copenhagen, 1998.

C Event Flow Control Protocols in RU Builder

All entities in the Readout Unit Builder communicate with each other via message-passing, using a peer-to-peer communication model. The full set of messages in the EVB is shown in Figure C-1 and Figure C-2. Within this communication model, we define a transaction to be the collection of all messag-

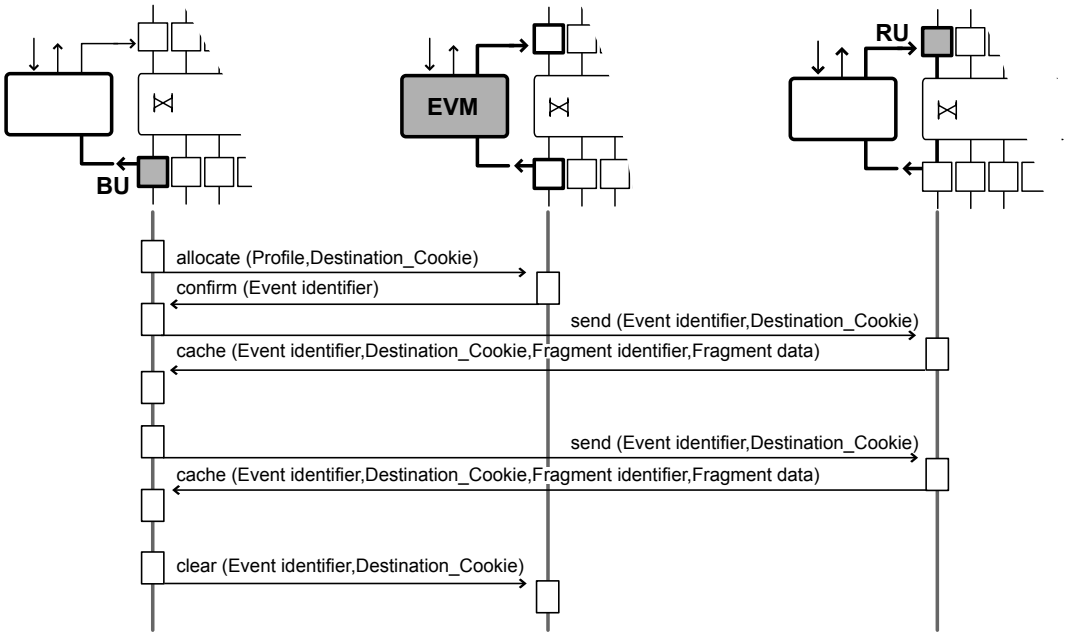


Figure C-1 BU, EVM and RU interaction diagram in the Readout Unit Builder.

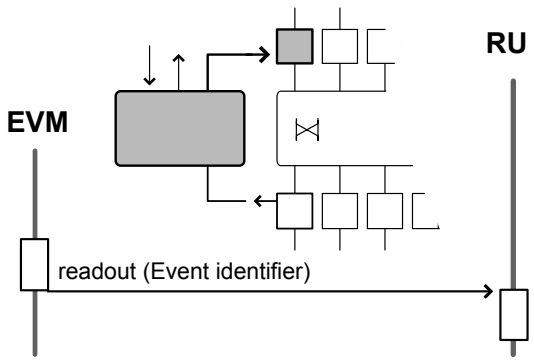


Figure C-2 EVM to RU interaction in the Readout Unit Builder.

es associated with a single desired function in the system. As an example, in Figure C-1, the allocation transaction consists of two messages: the message *allocate* from the BU to the EVM, requesting a new event and the associated response, from the EVM to the BU, *confirm*.

C.1 Transactions in the RU Builder

There are four possible transactions:

(a) Allocation: the goal of this transaction is to provide a BU with the identification tag (event_ID) of an event that is present in the Readout Units, for processing by the HLT algorithm in the Filter Farm Nodes.

(b) Deallocation: the goal of this transaction is to notify the EVM that an event has been completely processed and therefore that all DAQ resources taken up by the event (its identification tag, as well as the memory space in the RUs) can be released. Following this mesage, a BU no longer has the event in question allocated to itself.

(c) Event Data Transfer: the goal of this transaction is to transfer the data corresponding to one event from the Readout Units to the Filter Units. Filter Units request data only from events that are currently allocated to them.

(d) Readout: the goal of this transaction is to provides the RU with the identification tag for each new event in their memory. This tag will be used to address all data corresponding to this event until the next readout transaction with the same identification tag.

Transactions (b) and (d) are single-message (or, “one-way”) transactions: only one entity in each case (the BU for the deallocation and the EVM for the readout) sends a message. The correct response form the receiving subsystem is to be assumed. Transactions (a) and (c) are dual-message transactions, in the sense that the receiver of the first message of the transaction must send a message to complete the transaction. Dual-message transactions are characterized by the inclusion, in the message, of a Destination_Cookie, which is a tag generated by the originator of the transaction (which, so far, can only be a BU) for the following purposes:

(a) providing the receiver of the first message in the transaction with a unique tag identifying the sender. This tag carries not only a single integer, but also a field that is unique to each sender. This prevents other BUs from sending messages with a different BU being the recipient of the response (this can arise if the BU is in error). This second field can be thought of as a “password” for the BU to the EVM.

(b) matching (by the sender of the first message) of the second message of the transaction to the first message of the transaction, i.e. identifying this second message as a response

Transactions (a) and (b) take place through the Builder Control Network. Transaction (d) takes place through the Readout Control Network. We refer to transactions (a), (b) and (d) as “Control Transactions”.Transaction (c), the most complicated one, takes place across two networks: the request for data (send) is sent via the BCN, whereas the response from the RU (cache) is sent via the Builder Data Network. We refer to transaction (c) as a “Data Transaction”.

The three transactions related to bringing an event into a processor in the farm (i.e. the allocation, data transfer and deallocation transactions) are asynchronous with respect to the readout transaction. However, the EVM is charged with ensuring that events allocated to BUs always correspond to events that (a) have been read out already and (b) have not been cleared. A possible extension of this rule is that the same event is sent to multiple destinations. The exact criteria under which such a decision would be made are

not defined at this point, however this capability (on the EVM's part) is one that is currently deemed potentially useful in the future. In summary, the interactions shown in Figure C-1 and Figure C-2 are logically linked by the EVM, which, in the absence of a decision to send the same event to multiple BUs, guarantees that the BU-related transactions refer to genuinely new events in the DAQ.

C.2 Command Flow (Control Transactions)

As stated previously, there are three control transactions: Allocation and Deallocation of events, and the Readout transaction.

C.2.1 Allocate

The allocation transaction is initiated by the Builder Units. A BU that has space available for the reception of another event (for HLT processing) sends an allocate message to the EVM. The message carries two pieces of information:

(a) An event profile: this is a tag describing the type of event to be allocated. It is essentially a "trigger-type" specification.

(b) A cookie (Destination_Cookie): this is a tag created by the BU for its own internal use. This cookie, being internal to the BU, can contain any information whatsoever, in addition to any information relevant to identifying the transaction. As an example, this tag could also contain the identification (or address, or number) of the Filter Node (in the Filter Unit) that is to process the event. While this information is not necessary, it is deemed potentially useful in designing the BU.

The response to the allocate message from the BU to the EVM is a confirm message from the EVM to the BU in question. This confirm message carries two pieces of information:

(a) Event_ID: this is a tag identifying the event uniquely to all active entities in the Event Builder (i.e. RU, EVM and BU). All future transactions relevant to this event will use this Event_ID.

(b) A cookie (Destination_Cookie): this is a direct copy of the equivalent information sent by the BU to the EVM with the allocate message.

Possible Errors in this transaction are:

(a) The non-reception of the allocate command by the EVM: this has no adverse effects on the operation of the Event Builder, other than a buffer in the BU remaining unused until the next BU reset.

(b) The reception of the allocate command by the EVM, but with an incorrect content; there are three possibilities in this case. The worst-case is that the entire message arrives in an unintelligible form. This is equivalent to the non-reception of the message and is treated similarly.

The second worst case is that the Destination_Cookie placed in the message is incorrect in the first place. As an example, an errant BU may send an allocate command to the EVM placing in the message the address of another BU. To avoid this problem, the Destination_Cookie for each BU should be made unique to each BU. The cookie should further not be "known" to the other BUs. An example of such a mechanism is the generation, at initialization, at each BU, of a random number that is communicated, along with the BU number, to the EVM. The combination of these two numbers can then serve as the logical equivalent of a "username" and "password" for all future transactions to the EVM. One may wonder whether this "password" or "key" is necessary, since he

EVM can always use the source identification in the incoming stream (e.g. in TCP/IP the source is identified). In such cases, there is no need for the BU to include its identification in the actual data stream sent to the EVM, and the possibility of the eror just described vanishes.

The third error case is that of a Destination_Cookie that does not conform to what the EVM expects from this BU. In this case, the EVM simply ignores the request. (Alternatives are to answer with an illegal event_ID, e.g. -1, in the confirm message, thus indicating to the BU that something went wrong. The advantage of this is that the BU can immediately identify the problem and potentially take corrective actions — or even retry the command).

C.2.2 Clear

This transaction consists of a single message, clear, sent from the BU to the EVM. It carries two pieces of information:

(a) Event_ID: this is a tag identifying the event to be released from the Event Builder.

(b) A cookie (Destination_Cookie): this is a tag created by the BU for its own internal use. The main reason for having this tag in this message is to enable the cross-check described in the error conditions below.

Possible Errors in this transaction are:

(a) The non-reception of the clear command by the EVM: this has no adverse effects on the operation of the Event Builder, other than a buffer in the RU remaining unused until the next EVM reset. Alternatively, the EVM monitoring process may decide to release such an event_ID once a (long-enough, programmable) timeout period has expired.

(b) The reception of the clear command by the EVM, but with an incorrect content; there are two possibilities in this case. The worst-case is that the entire message arrives in an unintelligible form. This is equivalent to the non-reception of the message and is treated similarly.

The second possibility is that the Event_ID or the Destination_Cookie placed in the message is incorrect As an example, an errant BU may request the clearing of an event that is not allocated to itself. The error is recognised (by the EVM) via the “username”-”password” combination in the Destination_Cookie, along with a comparison of whether the event in question was indeed allocated to the BU in question. The EVM reaction, upon the detection of this error condition is, again, to ignore the request. The EVB can thus continue its operation, and the only side-effect will be that a buffer in the RUs remains unused until such time that the EVM is either reset or, driven by a timeout, decides to clear the event in question on its own.

C.2.3 Readout (of a New Event Accepted by the Level-1 Trigger)

This transaction consists of a single message, readout, sent from the EVM to all the Readout Units. It carries the following information:

(a) Event_ID: this is a tag identifying the event from now onwards, up until the next readout command, with the same event_ID, is issued.

(b) The Level 1 trigger information which allows for cross checking the event identity

Possible Errors in this transaction are:

(a) The non-reception of the readout command by all the RUs: the result of this error is that the EVM "thinks" that a new event is now available for processing, when in fact no such new event is present. Later, when the BU requests the data for the event, it will receive the data corresponding to the old event, if any, that was actually tagged with this event_ID. To avoid processing this event a second time (and possibly even writing it eventually to storage multiple times, an error that would go undetected unless some central intelligence in the Computing Services keps track of all the events forwarded to storage during an entire Run) the BU *must* have access to some information related to the actual event that the EVM intended to have read out by the RUs. This implies that the Event_ID tag returned by the EVM to the BU (in the confirm message) must include some tag relevant to the actual Level-1 Trigger that generated this event. The BU can then, upon reception of the new event data, compare the tag present in the data to that from the confirm command. For that reason, the Event_ID tag contains, as part of the unique Event Identifier, a trigger counter (a monotonously incrementing integer).

(b) The non-reception of the command by only a few RUs. The error is identified (by the BU that will eventually receive the data) via the Event_ID contents just described. The BU is then free to decide whether the number of RUs with the wrong event data is small (and thus the event can still be used) or to simply request a clear for the event (after notifying the Run Control System, of course).

(c) The reception of the message by the RUs, but with incorrect contents. There are two types of error here: (a) the Event_ID received is not the same as the one sent out by the EVM (due to some data corruption in the transmission) and (b) the Event_ID provided by the EVM itself is incorrect. Case (a) is easily covered by introducing the requirements that the RCN, and the full data transport mechanism over this network, has the ability to flag all data corruption cases (e.g. via a CRC).

Case (b) is a serious error: it can occur if and only if, the EVM, i.e. the controller of the EVB, has developed a major error. Moreover, the potential overwrite of some other event (possibly in transit) can lead to many unpredictable error conditions. There is, furthermore, no way of identifying this error condition. Therefore there must be a locking mechanism so that an event fragment may not be deleted or overwritten while it is still being worked on by the system. This may necessitate an explicit clear command which instucts the RU to recycle that event fragment space. A second conclusion is related to which Event_ID has been sent out (errantly) by the EVM. There are two possibilities: first, this wrong Event_ID corresponds to an Event already freed by the BU that owned it. This case does not result in an adverse behavior of the Event Builder. The second possibility is that the wrong Event_ID corresponds to an Event that is currently allocated to some BU which is analysing it. The exact effect of this error depends on the relative timing between the BU accesses to the Event data and the readout of the new event by the RUs.

There are, again, two sub-cases: first, the BU has completed the analysis of the Event and currently has a full copy of the event data. Second, it is still analysing this event — this includes the case where the event has been accepted, but the BU has yet to request or receive the last fragments from the event. The first case results in the event that was just read in being overwitten later without having passed the HLT filter. The second case can be detected by the BU via a cross-comparison of the event's internal data (bunch crossing number, Level-1 accept, etc.). The fundamental problem that this analysis reveals is that the entire sequence of overwriting an event without the original ever going through the HLT process is undetectable. There are two design options, therefore: one, to include an independent clear message (from the EVM to all the RUs) indicating the availability of an Event_ID for a future readout. This message is, essentially, a protection against the EVM later sending an "illegal" Event_ID, i.e. an Event_ID that is still in use. The second option is to assume that the EVM monitoring process will be capable of identifying this error. Note that the

fraction of events for which this error occurs will be very small (due to the inherently small probability of the EVM providing the wrong Event_ID), but also because of the small time interval during which the error can take place in this undetected way.

An alternative to allowing (albeit with a very small frequency) the occurrence of this error is to introduce a second EVM in the system. This second EVM will not be active, like the main EVM, but it will be an engine that (a) performs the same tasks as the real EVM and (b) receives form the main EVM, like all the RUs, the readout signal via the RCN. This second EVM is then a natural monitor of the main EVM: this includes both its decisions and the “other side” (the receiving side) of the RCN. Yet another possibility is that one of the lines of the RCN outputs is sent back into the main EVM which can then know whether there was a global “blackout” on the RCN output. All of these options have different merits but also introduce complexity into the system. The decision on which one will be adopted, however, can be delayed at this point in time.

C.3 Data Flow (Data Transactions)

There is only one data transaction, the send/cache combination between BUs and RUs. The send command is sent by the BU that has this event allocated to itself. The cache command is sent by the RU, in response to the send command, and contains the data from the event requested. The send command carries two pieces of information:

(a) Event_ID: this is the tag identifying the event whose data should be sent back to the BU

(b) Destination_Cookie: this is the destination identification (together with any “password” information and transaction identification the BU may decide to add) provided by the BU. The RU could use this information to determine the address of the BU that is requesting the data.

The response to the send message from the BUs to the RUs is a cache message from the RUs to the BUs. The message contains the data requested. The cache command caries the following four pieces of information:

(a) Event_ID: this is a tag identifying the event whose data is included in the cache message. It should, under normal operating conditions, be equal to the Event_ID in the send message.

(b) Destination_Cookie: this is the same tag as the one contained in the send message.

(c) Fragment identifier: this is a tag identifying the RU that has sent the cache command.

(d) Fragment data: this is the data corresponding to the event sent.

Possible errors in this transaction are:

(a) The Event_ID requested by the BU in question is one corresponding to an Event that is not allocated to this BU. This may result in the RU sending an extra copy to this errant BU. There is no other adverse effect on the EVB.

(b) The Event_ID in the cache message does not correspond to the one in the send message. The BU can simply ignore such replies and report them to the error monitor.

(c) Mismatch in the Destination_Cookie between the send and cache commands. This error is identified by the BU which can choose to drop the cache message and continue its operation (after notifying the error monitor).

(d) No response (i.e. no cache message) to the send from one or more RUs. The BU decides, after some appropriately long and programmable timeout period, that the request will not be fullfilled. It

can then decide (this will depend on the frequency of this error) either to request the same data again, to simply discard the event, or to forward the incomplete event.

(e) The fragment data do not correspond to the Event_ID requested, but all other parameters in the cache message are correct. This is recognised by the BU which then reports (to the error monitor) that the RU in question (or the FEDs connected to it) is out of synchronization.

C.4 Summary

From the above analysis we conclude that the Event_ID tag must be associated with:

1. A Trigger number (used, as explained above) to recover from the non-reception of the readout command by the RUs
2. An “event number” with a finite range (of, say, 100,000). Any new event accepted by the Level-1 is given a (free) number out of this range.

Similarly, the Destination_Cookie must contain the following information:

1. A BU identifier, used by the EVM and the RU for the response message
2. A BU “password”, used by the EVM to ensure that the BU issuing the command is not going to overwite some other BU, but also by the BU itself, when the cookie returns to ensure that the command is one that should have been sent to it and not another BU
3. Tags identifying the transaction in a BU-implementation way. This enables the BU to declare a transaction complete in an efficient way.

Optional tags are any other information the BU may decide to include in the cookie to facilitate its handling of the response message (e.g. for a cache message, the cookie could include the internal BU buffer number).

D Hardware Implementation of the Readout Unit

In Chapter 7, "Readout Column" the functionality of the Readout Unit (RU) has been introduced. A prototype based on custom hardware has been designed and built. This appendix describes the main functional concepts and the architecture of this prototype.

At the time this prototype was built, the DAQ architecture had not yet been finalized. Event building was assumed to be performed in a single step merging 500 event fragments of an average size of 2 kB to entire events. Therefore the RU had been designed to receive 2 kB fragments at a rate of 100 kHz.

D.1 Building blocks

Figure D-1 shows the functional blocks of the RU prototype:

- The input interface receives data from the Readout Unit Input (RUI) and transfers them via an internal PCI bus to the data memory.
- The output interface transfers data from the data memory to the Readout Unit Output (RUO) via a dedicated PCI bus.
- The data memory functions as a dual-port memory which can be written and read concurrently.
- The control unit receives the event builder messages from the Event Manager (EVM) and the Builder Units (BUs) which drive the event building process. On receipt of these messages the data transfer into and out of the RU memory is performed.

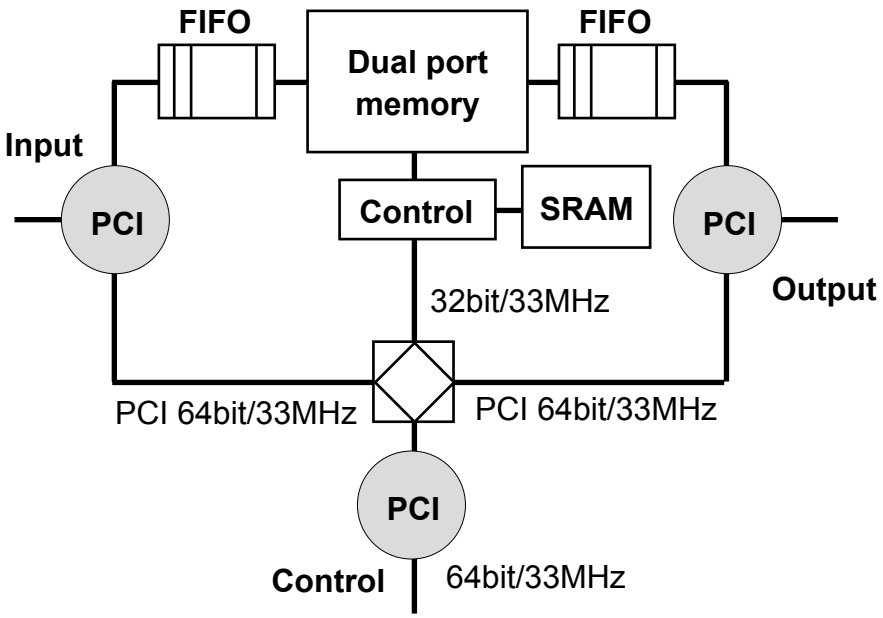


Figure D-1 Block diagram of the RU-prototype based on FPGAs.

These units communicate via four independent PCI buses which are interconnected by a single PCI bridge. The PCI buses at the input and output of the module are implemented with a data width of 64 bits. They interface to the RUI and the RUO, both implemented with programmable Network Interface Cards

(NICs). The control bus and the internal PCI bus are 32 bits wide. All buses are clocked at 33 MHz. The control messages for event building in principle can be received via any of the three PCI interfaces, since all PCI buses are interconnected by a bridge. In the tests performed, the messages from the EVM have been received via the interface to the host PC whereas the Builder Unit requests were issued by the RUO.

D.2 RU Memory Design

The memory of the RU is the main data buffer in the Readout Column. It is required to buffer the data stream of at least one second of data taking at full level 1 trigger rate. With a mean event fragment size of 2 kB this means a buffer memory of at least 200 MB. The RU implements its main buffer with a cost effective DIMM II memory module which is available on the PC market. Modules with up to 512 MB can be used. Since the memory is accessed simultaneously from the input and the output of the RU, some arbitration logic is needed to control the data traffic.

D.2.1 Data Input and Data Output Stream

At 100 kHz trigger rate and an average event fragment size of 2 kB the data throughput of the RU is 200 MB/s. Event fragments are stored in the buffer memory until they are requested by a BU. The incoming and the outgoing data streams are asynchronous to each other. Therefore the data memory is designed as a dual ported memory which can be written an read simultaneously. The memory bus operates at a frequency of 66 MHz with a data width of 64 bits. The resulting memory bandwidth of 528 MB/s is shared between the input stage and the output stage. The control logic of the memory ensures that the memory can be written and read with 200 MB/s concurrently.

D.2.2 Memory Organization

The order in which the BUs request event fragments from the RU is in general different from the order in which fragments are stored in the RU. Since in addition the fragments have a variable length, a Memory Management Unit (MMU) has been developed. The entire memory space is divided into fixed size blocks (the size of the blocks is programmable). Pointers to available memory blocks are held in a list implemented as a FIFO. Incoming data is written into as many data blocks as needed to hold the entire fragment. The information on how the blocks are chained is held in a separate RAM table. Once the fragment has been transferred to the output, the corresponding memory blocks are made available to newly incoming data by writing their pointers back into the list.

D.3 Event Fragment Control Logic

Incoming event fragments are associated with an unique event identifier received from the event manager. The same event identifier is used by the BU to request a specific event fragment. This means that the RU must be able to find the chain of memory blocks associated to a specific event identifier, in order to transfer the corresponding data to the RUO. For this purpose a look-up table is implemented in the RU. It is addressed with the event identifier and contains data fragment descriptors which contain information relevant for the efficient data transfer to the RUO.

E Fault Tolerance of Distributed Systems

The main issue is how to demonstrate the fault tolerance properties of a distributed system. There are two methods that have been used in the field of computer science:

- formal correctness proofs
- fault injection

Although formal methods provide valuable tools for identifying errors in some contexts, these tools lack the sensitivity, for example, to detect timing problems which are crucial for the DAQ system. Hence these methods are not considered in this document.

Before discussing the fault-injection procedure, the overall program is defined:

1. The current DAQ system design is supplemented in places that have been identified as potential sources of faults with additional functionality or improved components.
2. The design is established to be fault-tolerant after the current design is supplemented with the elements in the previous step. To do this, it is demonstrated that the current system and its associated protocols are such that all transient faults are handled in real-time and that, after relatively short periods of time, the system resumes its correct operation.
3. The implementation of the established fault-tolerant protocols is tested by using prototype software and hardware developed in the context of the CMS-DAQ R&D. This the step where fault-injection tests are carried out

Step-3, which is usually not included in error analyses of DAQ systems, is particularly important because fault-handling code is exercised only when a fault occurs. Given that the occurrence of faults should be the exception, rather than the rule, such code is therefore either not tested, or is only minimally tested. To establish the correctness and scaling of the fault handling mechanisms in the design, fault injection at the level of the prototype Event Builder and in a simulation of the full CMS system is used to establish the correctness of the actual implementation.

E.1 Failure Modes of Distributed Systems

In order to define the possible failure modes of distributed systems, the general properties of a distributed system must be defined.

A distributed system is a collection of nodes, where a node is any device or software package which has a self-contained function. For example, an Event Manager process running in a computer is a node; similarly for Readout Units or Builder Units. Another type of node particular to CMS is the Front-end Readout Link hardware module. In general, a node provides a service. A service-provider executes a collection of operations which can be self-triggered by passage of time or can be triggered by client-requests. Our Data-Acquisition system is therefore a collection of nodes or service-providers which interact with one another. An example is when an Event-Manager provides to a Builder-Unit an identification of the next event waiting to be assembled upon request from it. In return, the Builder-Unit notifies the event manager of the completion of the event assembly.

A server may implement its service by using the services of other servers. For example, the Event-Manager application, in providing a service to the Builder-Unit, can use a "Datagram" service which sends messages as sequences of packets; the Datagram server might in turn depend on the physical communica-

tion server which is used. Thus, the user/client and resource/server names are relative to the “depends” relation: what is a resource or server at a certain level of abstraction can be a client or a user at another level of abstraction [E-1].

A server performs a correct service if it behaves in a manner consistent with its service specification; that is, its outputs of internal state-transitions are correct. A failure occurs when the delivered service no longer complies with the agreed description of the system’s expected service [E-2]. Failure is a property of the external behaviour of a service-providing system — which is itself a manifestation of its internal states and state transitions [E-3]. A service may not, and generally does not, always fail in the same way. The ways a service can fail are its *failure modes*.

In a distributed system, a node is an appropriate unit of failure. The node implementation maps internal node failures into simple external failure modes [E-4]. Faults which affect only the internal state of a node, but do not propagate to the rest of the system can not affect the correctness of the behaviour (or integrity) of the entire system [E-5]. The failure modes of nodes are thus defined in terms of the messages that faulty nodes send or do not send. Such nodes can fail, according to the service specification domains described above, by producing messages with erroneous content, messages that arrive untimely, or indeed “impromptu” messages that should never have been sent at all [E-6].

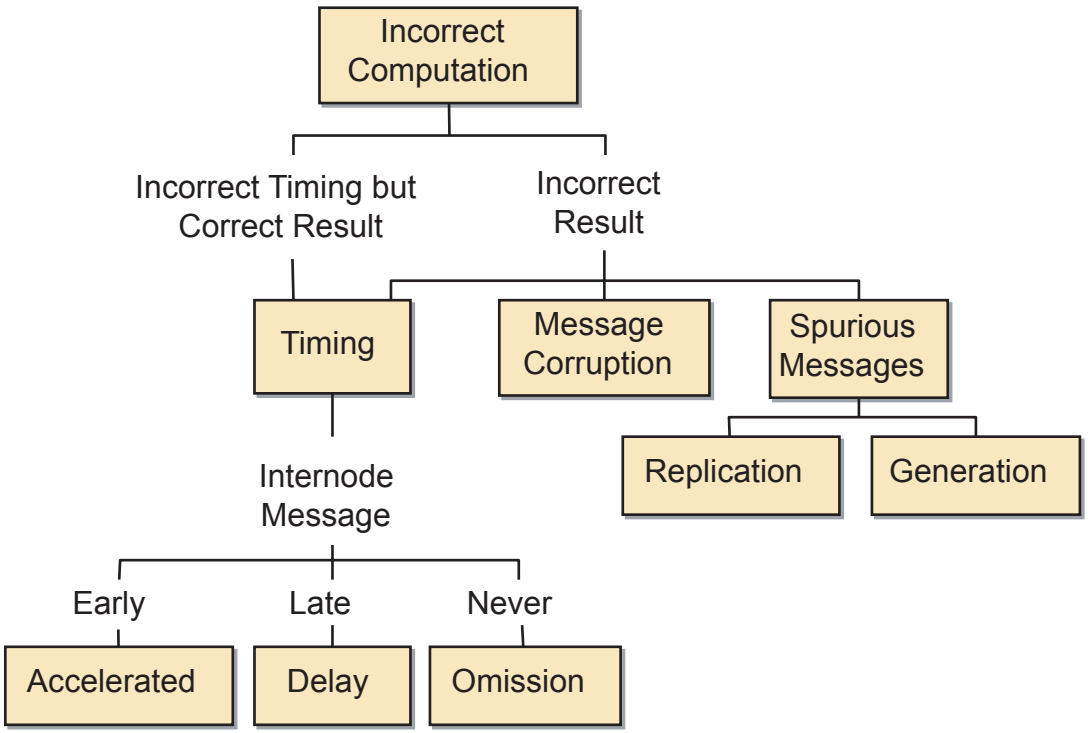


Figure E-1 Possible transaction faults in a computing system relevant to CMS-DAQ system. A detailed analysis can be found in references [E-1] and [E-2].

E.1.1 Fault Model Hierarchy

A subset of possible transaction faults in a computing system which is relevant to CMS-DAQ is shown in Figure E-1.Incorrect timing with a correct result is called a timing fault. In a *timing* fault, the correct result is delivered from a component either too early, too late or never. Timing faults can affect either a computational process or an inter-node message. Timing faults in processes that do not affect the associat-

ed node's output are not considered. Process timing that do affect output lead to one of the other fault types shown in Figure E-1.

- *accelerated* fault: a message is delivered before it is expected.
- *delay* faults message arrives at a destination later than expected.
- *omission* fault: a message never arrives. An omission fault can occur at the node level and can be either a send or a receive omission, or it can occur at the link level.
    - *link omission* fault: occurs when a communication link drops one or more messages.
    - *receive omission:* occurs when a node fails to receive a required message and behaves as though it did not arrive.
    - *send omission* occurs when a node fails to send a required message. Send omissions are particularly important since they can be used to emulate fail-silent[1], crash[2], and fail-stop[3] faults. From the communication standpoint, all three of these fault types can be emulated by send omission on all messages.

An incorrect result produced at a correct time can also be seen by a client as a timing fault. This can happen when the time at which a message is sent depends on a computed result. In fact, depending on how it is used, an incorrect result can lead to almost any type of fault. Two other types of such faults are message corruption and spurious messages.

In *message corruption*, the contents of a message are changed. Either field of a message, header or data, can be corrupted. The resulting error behaviour of the receiver depends on the which field within the header gets corrupted. Since message corruption can occur prior to a message being encoded for transmission, coding techniques are not sufficient to handle this type of fault.

A *spurious messag*e occurs when a node receives a message that it is not expecting. Spurious messages can occur in a variety of situations. For example, some failure modes may result in *message replication*, i.e. multiple copies of a message being sent. Other failure modes may cause *message generation*, where a message is produced when it should not be. A message which is sent to the wrong destination causes a spurious message at the new destination and an omission at the intended destination.

Faults occurring in the Event Builder can be analyzed at two abstraction levels: a system level and a specific component level. System-level faults are those faults that are independent of the specifics of the implementation of the system-components. On the other hand, specific-element faults depend on the details of the implementation. By using an abstraction at the system or functional level, faults specific to the functionality of the system can be analyzed. The relation of system-level faults to specific-element faults is a many-to-many association. Each system-level fault originates from one or more lower-level faults. As an illustration, consider the case of a bit-flip failure in a packet header. As a result of such a bit-flip fault, the possible system-level failure is a miss-routed message, which results in an omission failure and a spurious message. Note that the final observable, the omission failure of the current example, may arise because of numerous faults, for example, from a buffer overflow in a receiver Network-Interface-Card.

The system-level abstraction has the advantage of analysing the system without a reference to a particular implementation. As long as the faults belonging to the lower level and the system-level abstractions have the same coverage, in the following analysis it will be sufficient to consider only the system level faults.

1.A *fail-silent* node stops producing messages although its internal state may continue to be updated.
2.In a *crash* fault, the node stops sending messages and the internal state may be lost completely.
3.A *fail-stop* node halts but maintains the internal state it had just prior to halting.

A transient fault [E-4] is a class of faults which exist only for a short interval of time and disappear without an explicit repair action from a higher authority. A transient fault might lead the system into an error state which can be permanent. If this error state is not detected and handled properly, it will lead to a system failure which will degrade the system availability and reliability. A system with the ability to recover from inconsistent states caused by transient faults without an outside intervention and regardless of the fault's origin, is said to be a self-stabilizing system [E-2].

The DAQ system will automatically recover to a consistent state if all sub-systems recover within a finite time interval from inconsistent states. This self-stabilizing property of the event-building part of the DAQ system can be guaranteed even in the case of transient faults. However, the self-stabilizing property does not necessarily guarantee a continuous reliability.

E.2 Fault Classification

In a distributed system, faults occurring in a subsystem can affect overall system behaviour only if they propagate into other subsystems. Inversely, faults which affect only the internal state of a subsystem, but do not propagate to the rest of the system, cannot affect the correctness of the behaviour of the entire system. This property allows the emulation of each possible failure by a corresponding transaction failure [E-6]. Another consequence of this property is that the origin of a fault cannot be identified uniquely.

An in-depth study of this classification of transaction faults has been done in [E-1]. A specific implementation in the domain of performance evaluation using fault-injection techniques is described in [E-6]. The analysis can be significantly simplified, if only the possible observable transaction failures relevant to CMS are considered, as shown in Figure E-1.

In the classification of Figure E-1 there are three main classes of faults: those relating to timing, message corruption and spurious messages. In order to prove that the design is fault-tolerant, it must be shown that there are three and only three fault classes and that there exist such protocol extensions that are can detect and recover from these faults. The first part of the proof is not tractable formally, and therefore completeness cannot be claimed. Instead the analysis proceeds with a description of all possible faults that can be identified.

E.3 References

E-1 C. Flaviu, "Understanding Fault-Tolerant Distributed Systems", CACM, February 1991.

E-2 J. C. Laprie, "Dependability: Basic Concepts and Terminology", IFIP WG 10.4 Dependable Computing and Fault Tolerance. Springer,1992

E-3 J. Rushby, "Critical System Properties: Survey and Taxonomy", Reliability Engineering and System Safety, 43 (1194) 189.

E-4 H. Kopetz, "Real-Time Systems: Design Principles for Distributed Embedded Applications", Kluwer Academic Publishers, 1997.

E-5 D. M. Blough and T. Torii, "Fault-Injection-Based Testing of Fault-Tolerant Algorithms", in Message-Passing Parallel Computers, IEEE 27th Annual International Symposium on Fault Tolerant Computing, 1997.

E-6 C. Flaviu, "Abstraction for Fault-Tolerance", 13th IFIP World Computer Congress, Hamburg, 1994.

F Summary of Level-1 Trigger

The Level-1 Trigger System [F-1] is organized into three major subsystems: the Level-1 calorimeter trigger, the Level-1 muon trigger, and the Level-1 global trigger. The muon trigger is further organized into subsystems representing the 3 different muon detector systems, the Drift Tube Trigger in the barrel, the Cathode Strip Chamber (CSC) trigger in the endcap and the Resistive Plate Chamber (RPC) trigger covering both barrel and endcap. The Level-1 muon trigger also has a global muon trigger that combines the trigger information from the DT, CSC and RPC trigger systems and sends this to the Level-1 global trigger. A diagram of the Level-1 Trigger system is shown in Figure F-1.

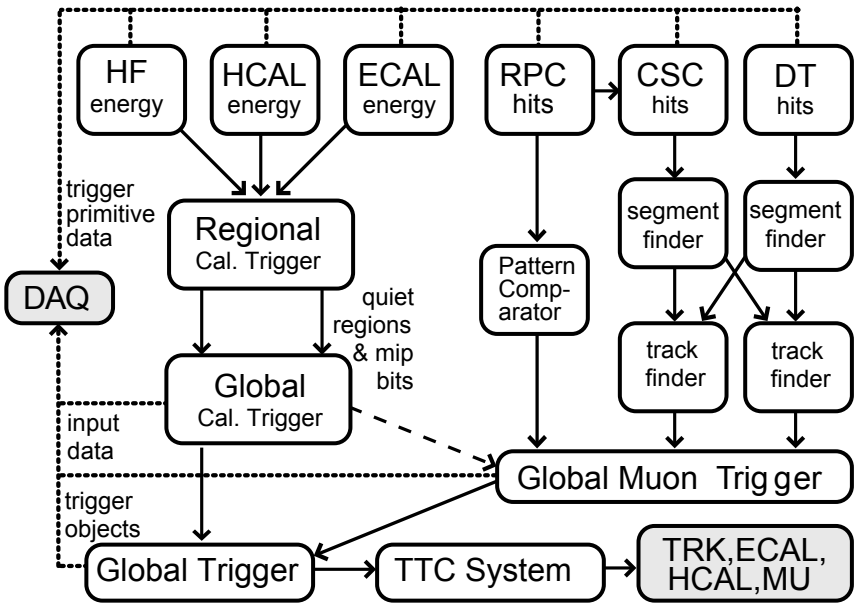


Figure F-1 Overview of the Level-1 Trigger system.

The data used as input to the Level-1 Trigger system as well as the input data to the global muon trigger, global calorimeter trigger and the global trigger are transmitted to the DAQ for storage along with the event readout data. In addition, all trigger objects found, whether they were responsible for the Level-1 Trigger or not, are also sent. The decision whether to trigger on a specific crossing or to reject that crossing is transmitted via the Trigger Timing and Control system to all of the detector subsystem front-end and readout systems.

F.1 Calorimeter Trigger Description

The calorimeter trigger begins with ($0.35\eta \times 0.35\phi$) trigger tower energy sums formed by the ECAL, HCAL and HF upper level readout Trigger Primitive Generator (TPG) circuits from the individual calorimeter cell energies. For the ECAL, these energies are accompanied by a bit indicating the transverse extent of the electromagnetic energy deposit. For the HCAL, the energies are accompanied by a bit indicating the presence of minimum ionizing energy. The TPG information is transmitted over high speed copper links to the Regional Calorimeter Trigger (RCT), which finds candidate electrons, photons, taus, and jets. The RCT separately finds both isolated and non-isolated electron/photon candidates. The RCT transmits the candidates along with sums of transverse energy to the Global Calorimeter Trigger (GCT).

The GCT sorts the candidate electrons, photons, taus, and jets and forwards the top 4 of each type to the global trigger. The GCT also calculates the total transverse energy and total missing energy vector. It transmits this information to the global trigger as well. The RCT also transmits an (η,ϕ) grid of quiet regions to the global muon trigger for muon isolation cuts.

F.1.1 Overview of Calorimeter Trigger Algorithms

An overview of the electron/photon isolation algorithm is shown in Figure F-2. This algorithm involves only the eight nearest neighbours around the central hit trigger tower and is applied over the entire (η,ϕ) plane. The electron/photon candidate E_T is determined by summing the E_T in the hit tower with the maximum E_T tower of its four broad side neighbours. This summed transverse energy provides a sharper efficiency turn-on with the true E_T of the particles.

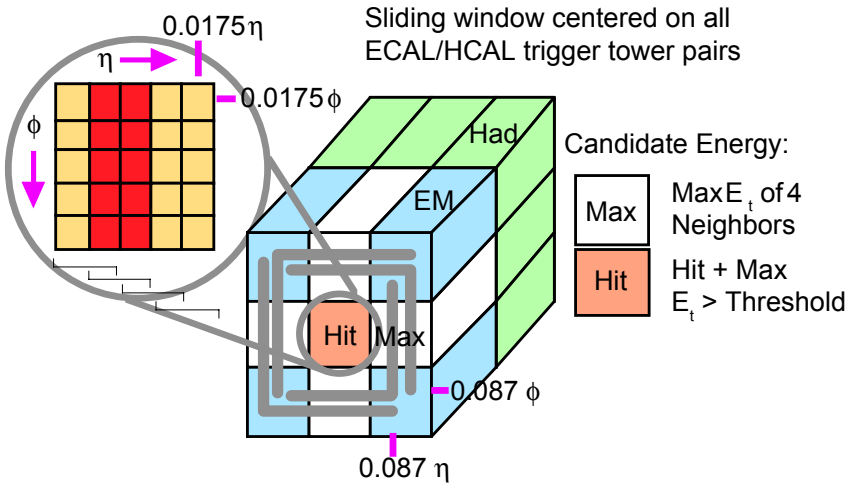


Figure F-2 Electron/photon trigger algorithm.

The non-isolated candidate requires passing of two shower profile vetoes, the first of which is based on the fine-grain ECAL crystal energy profile. The second is based on HCAL to ECAL energy comparison, e.g. H/E less than 5% (HAC veto).

The isolated candidate requires passing of two additional vetoes, the first of which is based on the passing of FG and HAC Vetoes on all eight nearest neighbours, and the second is based on there being at least one quiet corner, i.e., one of the five-tower corners has all towers below a programmable threshold, e.g., 1.5 GeV. Each candidate is characterized by the (η,ϕ) indexes of the calorimeter region where the hit tower is located.

F.1.2 Electron/Photon Triggers

In each calorimeter region (4×4 trigger towers) the highest E_T non-isolated and isolated electron/photon candidates are separately found. The 16 candidates of both streams found in a regional trigger crate (corresponding to 16 calorimeter regions covering $\Delta\eta\times\Delta\phi=3.0\times0.7$) are further sorted by transverse energy. The four highest-E_T candidates of both categories from each crate are transferred to the GCT where the top four candidates are retained for processing by the CMS global trigger.

The nominal electron/photon algorithm allows both non-isolated and isolated streams. The non-isolated stream uses only the hit tower information except for adding in any leakage energy from the maximum neighbour tower. This stream will be used at low luminosity to provide the B-electron trigger. The isolation and shower shape trigger cuts are programmable and can be adjusted to the running conditions. For example, at high luminosity the isolation cuts could be relaxed to take into account higher pile-up energies. The specification of the electron/photon triggers also includes the definition of the η–ϕ region where it is applicable. In particular, it is possible to define different trigger conditions (energy thresholds, isolation cuts) in different rapidity regions.

The efficiency of the electron/photon algorithm, as a function of the electron transverse momentum, for different thresholds applied at Level-1, is shown in Figure F-3. Also shown in Figure F-3 is the efficiency, as function of pseudorapidity for electrons with P_T=35 GeV/c.

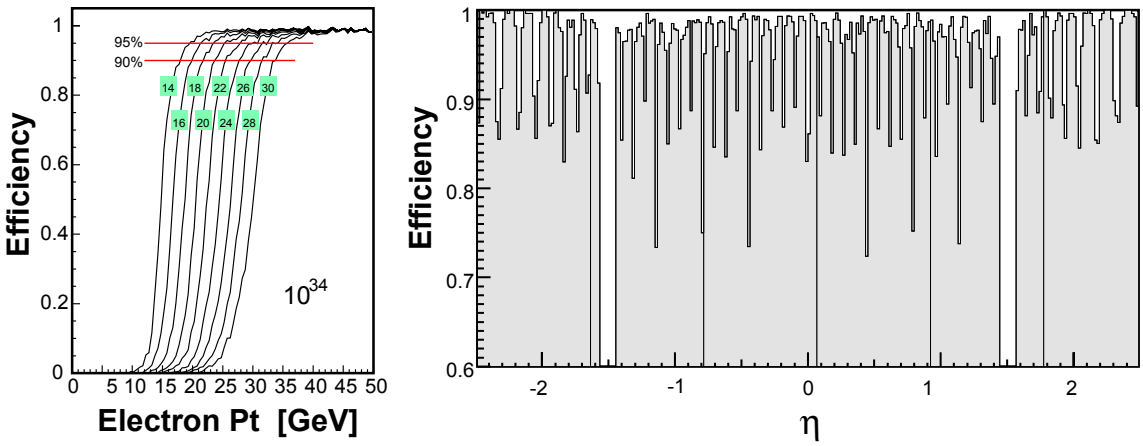


Figure F-3 The efficiency of the Level-1 Trigger for single electrons as a function of the electron P_T. On the right, the efficiency, as function of η, for electrons with P_T=35 GeV/c.

To connect the Level-1 threshold to an effective requirement on the electron transvsers momentum, the electron P_T at which the Level-1 Trigger is 95% efficient, is determined as function of the Level-1 threshold. This is shown in Figure F-4.

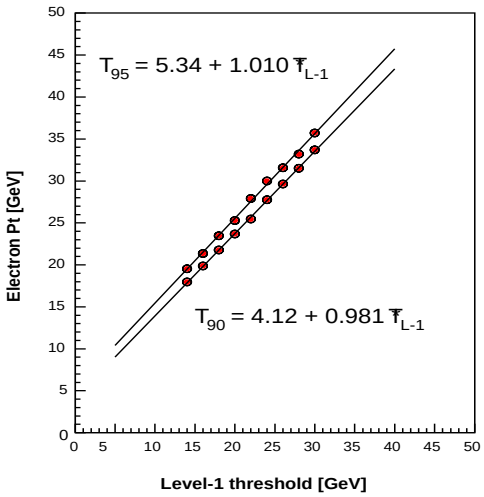


Figure F-4 The electron E_T at which the Level-1 Trigger is 95% efficient as a function of the Level-1 threshold.

From this result, the rate for electron/photon triggers as a function of the effective cut on the E_T, i.e. of the point at which the trigger is 95% efficient, can be computed. Figure F-5 shows the rates for single electrons as a function of the E_T of the electron (95% point).

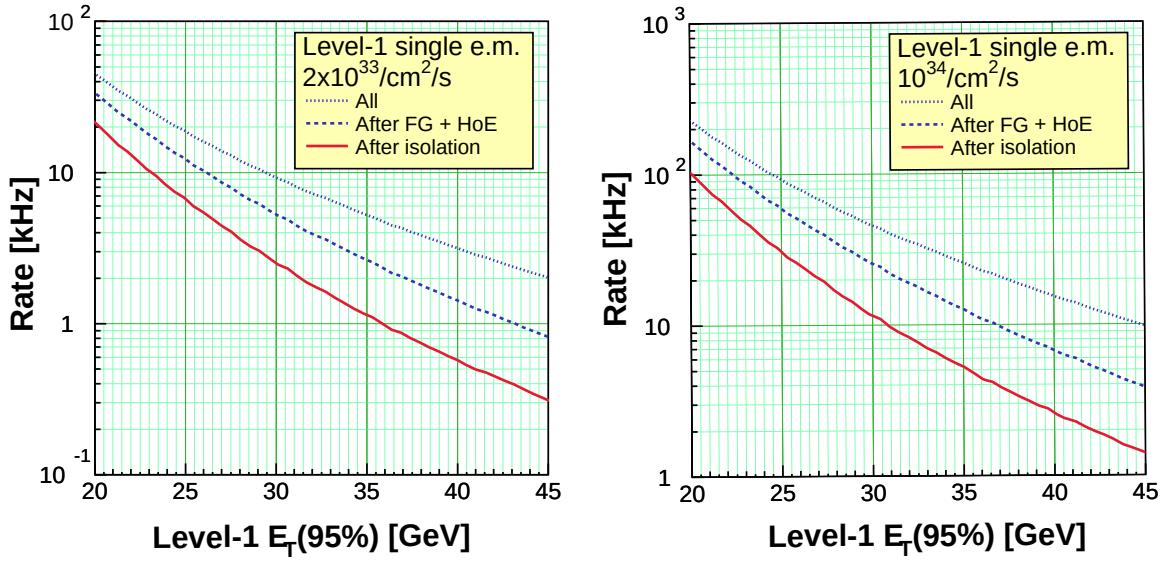


Figure F-5 The rate of the single electron/photon Level-1 Trigger at low (left) and high (right) luminosity.

Double, triple and quad electron/photon triggers can be defined. The requirements on the objects of a multi electron/photon trigger, namely the energy threshold, the cluster shape and isolation cuts and the (η,ϕ) region, are set individually. Requirements on the (η,ϕ) separation between objects can also be defined. The rate for two-electron/photon triggers is shown in Figure F-6.

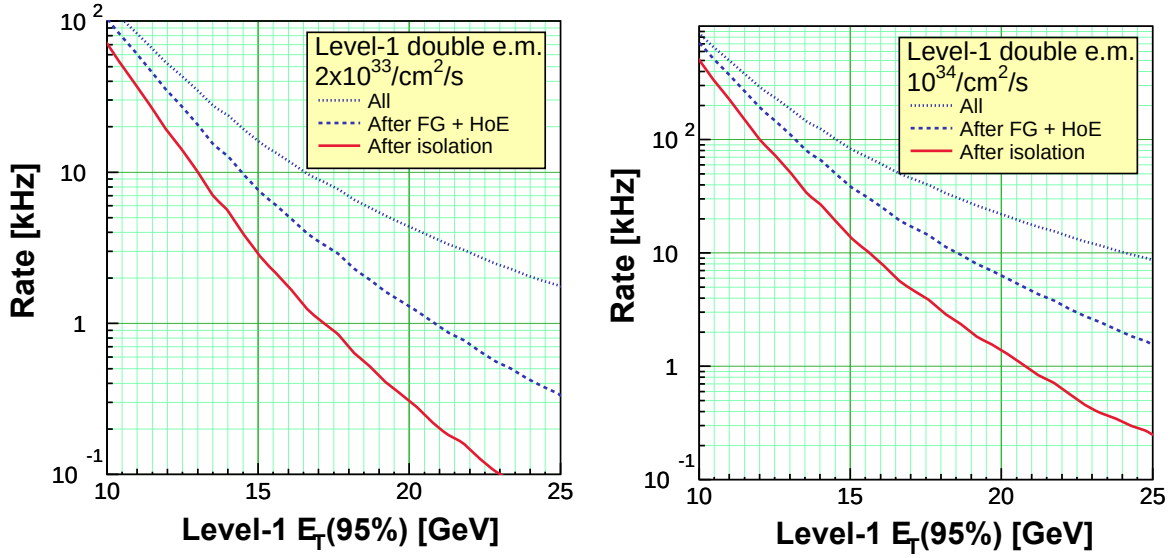


Figure F-6 The rate of the double electron/photon Level-1 Trigger at low (left) and high (right) luminosity.

F.1.3 Jet and τ Triggers

The jet trigger uses the transverse energy sums (em+had) computed in calorimeter regions (4×4 trigger towers), except in the HF region where it is the trigger tower itself. The input tower E_T is coded in an 8 bit linear scale with programmable resolution. Values exceeding the dynamic range are set to the maximum. The subsequent summation tree extends to a 10 bit linear scale with overflow detection. Simulation studies showed that a scale of 10 bits with LSB=1 GeV gives adequate jet trigger performance.

The jet trigger uses a 3×3 calorimeter region sliding window technique which spans the complete (η,φ) coverage of the CMS calorimeters (Figure F-7) seamlessly. The central region E_T is required to be higher than the eight neighbour region E_T values. In addition, the central region E_T is required to be greater than a fixed value, e.g., 5 GeV, to suppress spurious soft.

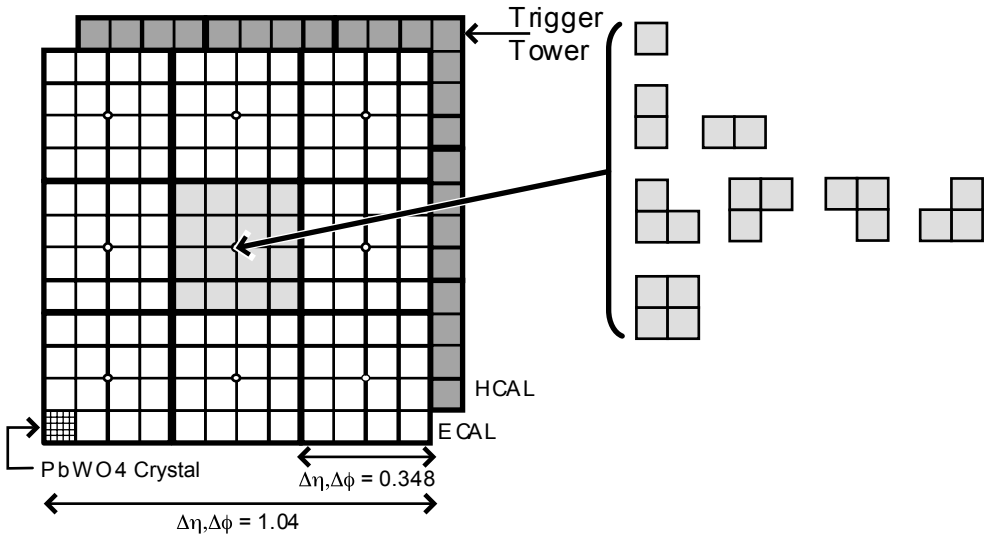


Figure F-7 Jet and τ trigger algorithms.

The jets and τs are characterized by the transverse energy E_T in 3×3 calorimeter regions. The summation spans 12×12 trigger towers in barrel and endcap or 3×3 larger HF towers in the HF. The φ size of the jet window is the same everywhere. The η binning gets somewhat larger at high η due to the size of calorimeter and trigger tower segmentation. The jets are labelled by (η,φ) indexes of the central calorimeter region.

Single and three-prong decays of τ leptons form narrow clusters of energy deposits in the calorimeter. Since the decays involve charged pions which deposit energies in the hadron calorimeter, the electron/photon trigger does not capture them. Therefore, the transverse profiles of active tower patterns are analysed to tag narrow jets as potential τ lepton decays. An active tower is defined as a trigger tower with ECAL or HCAL E_T above a separately programmable threshold. The calorimeter region has τ-veto set ON if the active towers do not form any of the 8 patterns shown in the Figure F-7, i. e., the pattern is not confined within 2×2 contiguous trigger towers of the calorimeter region. A jet is defined as "τ-like" if none of the 9 calorimeter region τ-veto bits are ON. The τ-veto bits are therefore used for both τ-like energy deposit identification and quite stringent isolation. This strict isolation requirement is tolerable because such cuts are necessary to find τs even in later offline analysis.

In addition counters of the number of jets above programmable thresholds in various η regions are provided to give the possibility of triggering on events with a large number of low energy jets. Jets in the for-

ward and backward HF calorimeters are sorted and counted separately. This separation is a safety measure to prevent more background susceptible high η region from masking central jets. Although the central and forward jets are sorted and tracked separately through the trigger system, the global trigger can use them seamlessly as the same algorithm and resolutions are used for the entire η−ϕ plane.

Single, double, triple and quad jet (τ) triggers are possible. The single jet (τ) trigger is defined by the transverse energy threshold, the (η,ϕ) region of validity and eventually by a prescaling factor. Prescaling will be used for low energy jet (τ) triggers, necessary for efficiency measurements. The multi jet (τ) triggers are defined by the number of jets (τs) and their transverse energy thresholds, by a minimum separation in (η,ϕ), as well as by a prescaling factor. The global trigger accepts the definition, in parallel, of different multi jet (τ) triggers conditions.

The four highest energy central and forward jets, and central τs in the calorimeter are selected. Jets and τs occurring in a calorimeter region where an electron is identified are not considered. The selection of the four highest energy central and forward jets and of the four highest energy τs provides enough flexibility for the definition of combined triggers.

Figure F-8 shows the Level-1 rates for jet triggers at low and high luminosity. Another quantity of interest, which is also useful in making comparisons between different algorithms and different detectors, is the rate as a function of the generator-level jet E_T by plotting the rate for the cut on offline jet E_T that gives 95% efficiency for the generator-level jet E_T. This is shown in Figure F-9

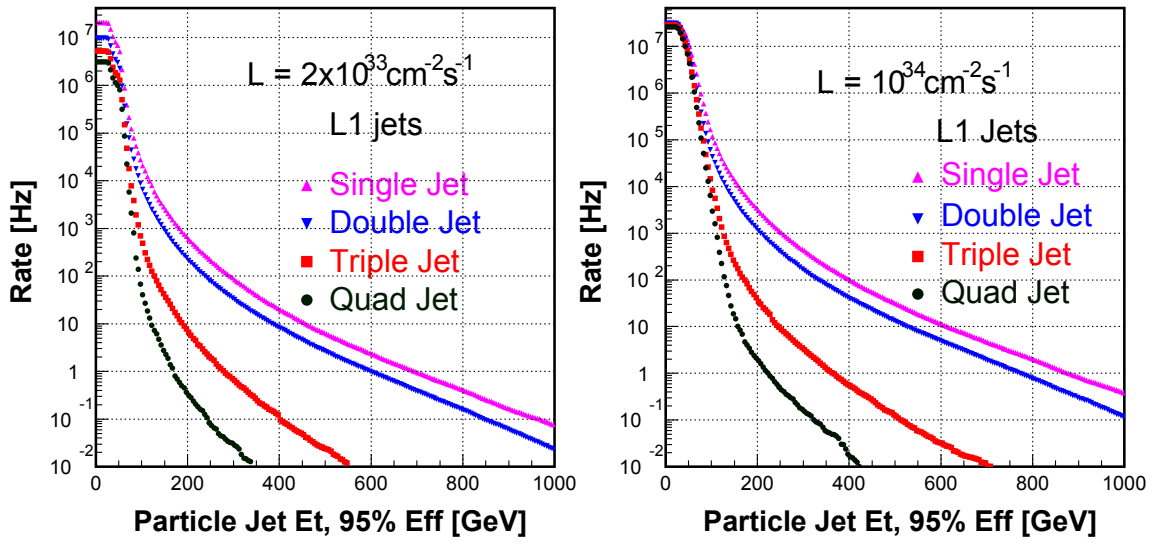


Figure F-9 Rate for the cut on Level-1 jet E_T that is 95% efficient for jets with generator-level E_T of the value given on the x-axis versus generator-level E_T. Left: low luminosity; right: high luminosity.

F.1.4 Energy Triggers

The E_T triggers use the transverse energy sums (em+had) computed in calorimeter regions (4×4 trigger towers in barrel and endcap). E_x and E_y are computed from E_T using the coordinates of the calorimeter region center. The computation of missing transverse energy from the energy in calorimeter regions does not affect significantly the resolution for trigger purposes.

The missing E_T is computed from the sums of the calorimeter regions E_x and E_y. The sum extends up to the end of forward hadronic calorimeter, i.e., |η|=5. The missing E_T triggers are defined by a threshold

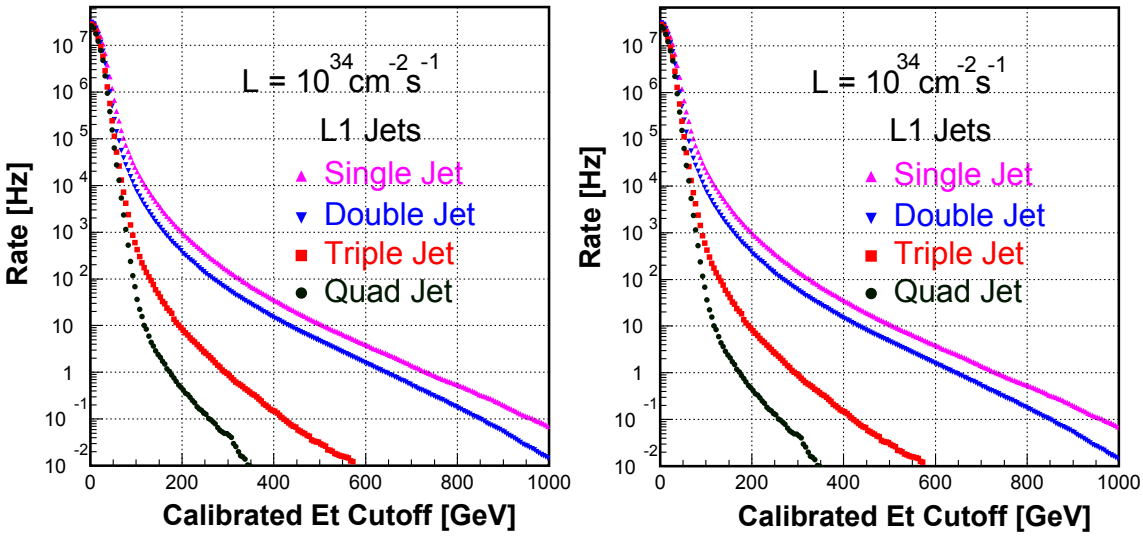


Figure F-8 Level-1 jet trigger rates for low and high luminosity.

value and by a prescaling factor. The global trigger accepts the definition, in parallel, of different missing E_T triggers conditions.

The total E_T is given by the sum of the calorimeter regions E_T. The sum extends up to the end of forward calorimeter, i.e., $|\eta|=5$. The total E_T triggers are defined by a threshold value and by a prescaling factor. The global trigger accepts the definition, in parallel, of different total E_T triggers conditions. The total energy trigger is implemented with a number of thresholds which are used both for trigger studies and for input to the luminosity monitor. Some of these thresholds are used in combination with other triggers. Other thresholds are used with a prescale and one threshold is used for a stand-alone trigger. The lower threshold E_T trigger also provides a good diagnostic for the calorimeter and its trigger.

The H_T trigger is defined as the scalar sum of the E_T of jets above a programmable threshold with a typical value of jet $E_T > 10$ GeV. This trigger is not as susceptible as the total E_T given by the sum of the calorimeter regions E_T deposits to both noise and pileup effects. The Although total E_T is a necessary technical trigger, it has limited use from physics point of view. The H_T trigger can capture high jet multiplicity events such as those from fully hadronic top decay, hadronic decays of squarks and gluinos. Although these events have several hundred GeV/c^2 energy, they may actually fail the jet triggers because individual jet E_Ts may be quite a bit softer than the sustainable thresholds. In addition, the H_T trigger can use individually calibrated jet energies unlike the total E_T trigger which cannot be easily calibrated.

F.2 Muon Trigger Description

Each of the Level-1 muon trigger systems has its own trigger logic. The RPC strips are connected to a Pattern Comparator Trigger (PACT), which is projective in η and ϕ. The PACT forms trigger segments which are connected to segment processors which find the tracks and calculate the P_T. The RPC logic also provides some hit data to the CSC trigger system to improve resolution of ambiguities caused by 2 muons in the same CSC.

The Cathode Strip Chambers form Local Charged Tracks (LCT) from the Cathode Strips, which are combined with the Anode wire information for bunch crossing identification on a Trigger Motherboard The LCT pattern logic assigns a P_T and quality, which is used to sort the LCT on the Motherboard and the

Muon Port Card that collects LCTs from up to 9 CSC chambers. The top 3 LCTs from all the MPCs in a sector are transmitted to the CSC Track Finder, which combines the LCTs into full muon tracks and assigns P_T values to them. The CSC and Drift Tube Track-Finders exchange track segment information in the region where the chambers overlap.

The Barrel Muon Drift Tubes are equipped with Bunch and Track Identifier (BTI) electronics that finds track segments from coincidences of aligned hits in 4 layers of one drift tube superlayer. The track segments positions and angles are sent to the Track Correlator (TRACO), which attempts to combine the segments from the two Super Layers (SL) measuring the ϕ coordinate. The best combinations from all TRACOs of a single chamber together with the SL η segments are collected by the Trigger Server. The Trigger Server then sends the best two segments (if found) to the Track Finder, which combines the segments from different stations into full muon tracks and assigns P_T values to them.

The Global Muon Trigger sorts the RPC, DT and CSC muon tracks, converts these tracks into the same η, ϕ and P_T scale, and validates the muon sign. It then attempts to correlate the CSC and DT tracks with RPC tracks. It also correlates the found muon tracks with an η-ϕ grid of quiet calorimeter towers to determine if these muons are isolated. The final ensemble of muons are sorted based on their initial quality, correlation and p_T and then the 4 top muons are sent to the Global Trigger.

F.2.1 Overview of Muon Trigger Algorithms

The logical structure of the Muon Trigger is shown in Figure F-10. DT and CSC electronics first process the information from each chamber locally. Therefore they are called *local triggers*. As a result one vector (position and angle) per muon per station is delivered. Vectors from different stations are collected by the Track Finder (TF) which combines them to form a muon track and assigns a transverse momentum value. TF plays the role of a *regional trigger.* Up to 4 best (highest P_T and quality) muon candidates from each system are selected and sent to the Global Muon Trigger.

In the case of RPC there is no local processing apart from synchronisation and cluster reduction. Hits from all stations are collected by PACT logic. If they are aligned along a possible muon track, a p_T value is assigned and the information is sent to the Muon Sorter. The Muon Sorter selects 4 highest p_T muons from the barrel and 4 from the endcaps and sends them to the Global Muon Trigger. The Global Muon Trigger compares the information from TF (DT/CSC) and PACT (RPC). Quiet bits delivered by the Calorimeter Trigger are used to form an isolated muon trigger. The 4 highest p_T muons in the whole event are then transmitted to the Global Trigger. Finally transverse momentum thresholds are applied by the Global Trigger for all trigger conditions.

F.2.2 Drift Tube Trigger

The drift chambers deliver data for track reconstruction and triggering on different data paths. The local trigger is based on two SL in the ϕ view of the muon station. The trigger front-end, called the Bunch and Track Identifier (BTI), is used in the ϕ view and the θ view to perform a rough muon track fit in one station measuring position and direction of trigger candidate tracks with at least three hits, in different planes of a Super Layer (SL). The algorithm fits a straight line within programmable angular acceptance. The BTI performs the bunch crossing assignment of every found muon track candidate. The algorithm used in the device is a generalization of the mean-timer method [F-2].

Since this method must foresee alignment tolerances and needs to accept alignments of only three hits, the algorithm can generate false triggers. Hence in the bending plane a system composed by a Track Correla-

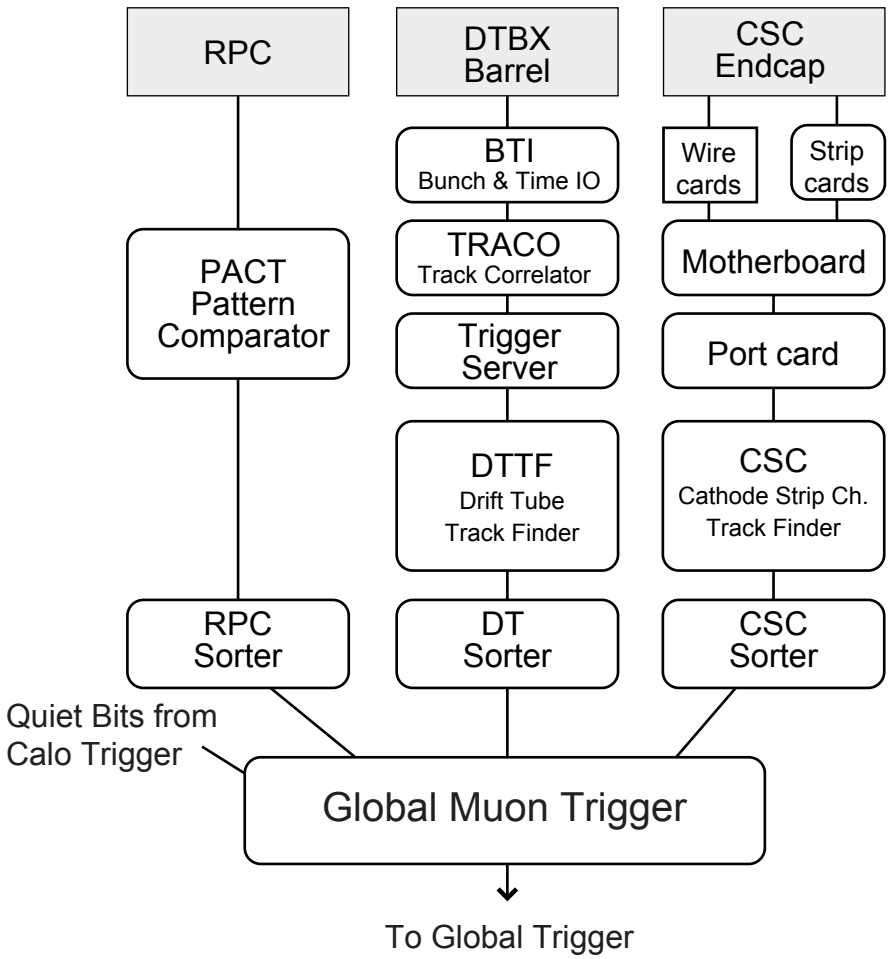


Figure F-10 Block diagram of the Level-1 muon trigger.

tor (TRACO) and a chamber Trigger Server (TS) is used to filter the information of the two ϕ SLs of a chamber in order to lower the trigger noise. The TRACO/TS block selects, at every cycle among the trigger candidates, at most two tracks with the smallest angular distances (i.e. higher P_T) with respect to the radial direction to the vertex.

Track segments found in each station are then transmitted to a regional trigger system called Drift Tube Track Finder (DTTF). The task of the Track Finder is to connect track segments delivered by the stations into a full track and assign a transverse momentum value to the finally resolved muon track. The system is divided in sectors, each of them covering 30° in the ϕ angle. Each Sector Processor is logically divided in three functional units - the Extrapolator Unit (EU), the Track Assembler (TA) and the Assignment Units (AU), as shown in Figure F-11.

The Extrapolator Unit attempts to match track segments pairs of distinct stations. Using the spatial coordinate ϕ and the bending angle of the source segment, an extrapolated hit coordinate is calculated. The two best extrapolations per each source are forwarded to the Track Assembler. The Track Assembler attempts to find at most two tracks in a detector sector with the highest rank, i.e. exhibiting the highest number of matching track segments and the highest extrapolation quality.Once the track segment data are available to the Assignment Unit, memory based look up tables are used to determine the transverse momentum, the ϕ and η coordinates, and a track quality.

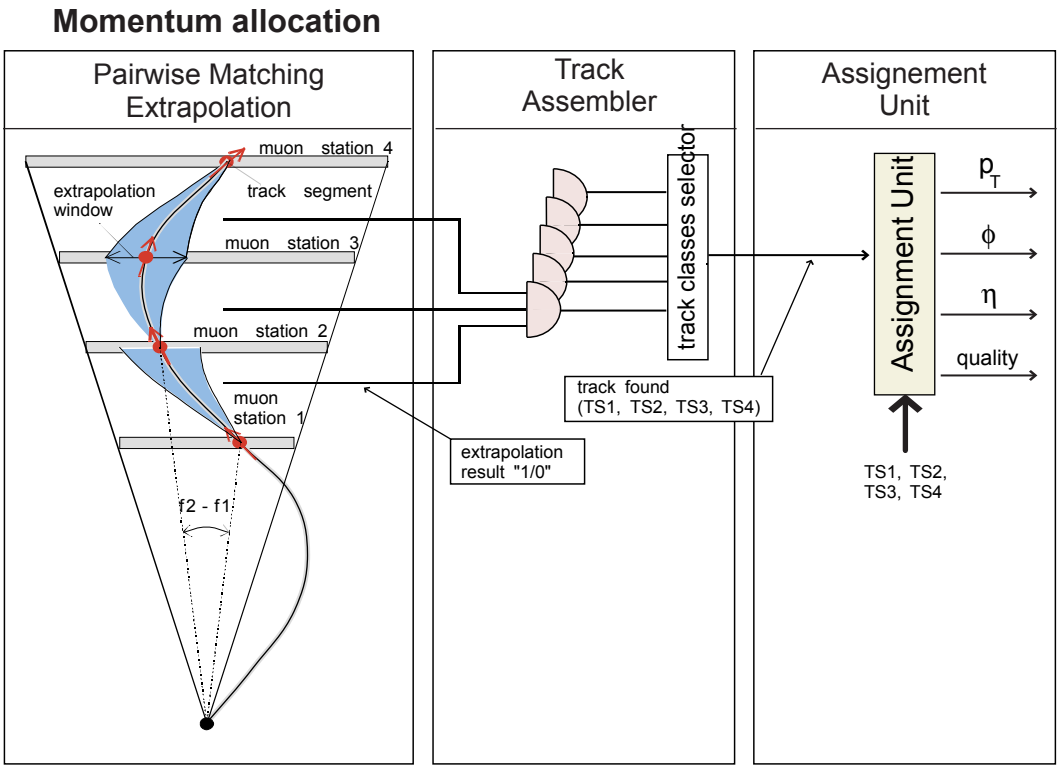


Figure F-11 Principle of the Drift Tube track finder algorithm. The “three-step” scheme utilized is shown. On the left side, the pairwise matching algorithm is described. An extrapolated hit coordinate is calculated using the f1 coordinate and the bending angle of the source segment. The match is considered successful if a target segment is found at the extrapolated coordinate, inside a certain extrapolation window.

F.2.3 CSC Trigger

The task of the Cathode Strip Chamber (CSC) Track-Finder is to reconstruct tracks in the CSC endcap muon system and to measure the transverse momentum (P_T), pseudo-rapidity (η), and azimuthal angle (ϕ) of each muon. The measurement of P_T by the CSC trigger uses spatial information from up to three stations to achieve a precision similar to that of the DT Track-Finder despite the reduced magnetic bending in the endcap. The CSC Local Trigger provides high rejection power against these backgrounds by finding muon segments, also referred to as Local Charged Tracks (LCTs), in the 6-layer endcap muon CSC chambers. Muon segments are first found separately by anode and cathode electronics (see Figure F-12) and then time correlated, providing precision measurement of the bend coordinate position and angle, approximate measurement of the non-bend angle coordinate, and identification of the correct muon bunch crossing with high probability.

The primary purpose of the CSC anode trigger electronics is to determine the exact muon bunch crossing with high efficiency. Since the drift time can be longer than 50 ns, a multi-layer coincidence technique in the anode “Local Charged Track” (LCT) pattern circuitry is used to identify a muon pattern and find the bunch crossing.

The primary purpose of the CSC cathode trigger electronics is to measure the ϕ coordinate precisely to allow a good muon momentum measurement up to high momentum. The charge collected on an anode wire produces an opposite-sign signal on several strips, and precision track measurement is obtained by charge digitization and precise interpolation of the cathode strip charges. The six layers are then brought into co-

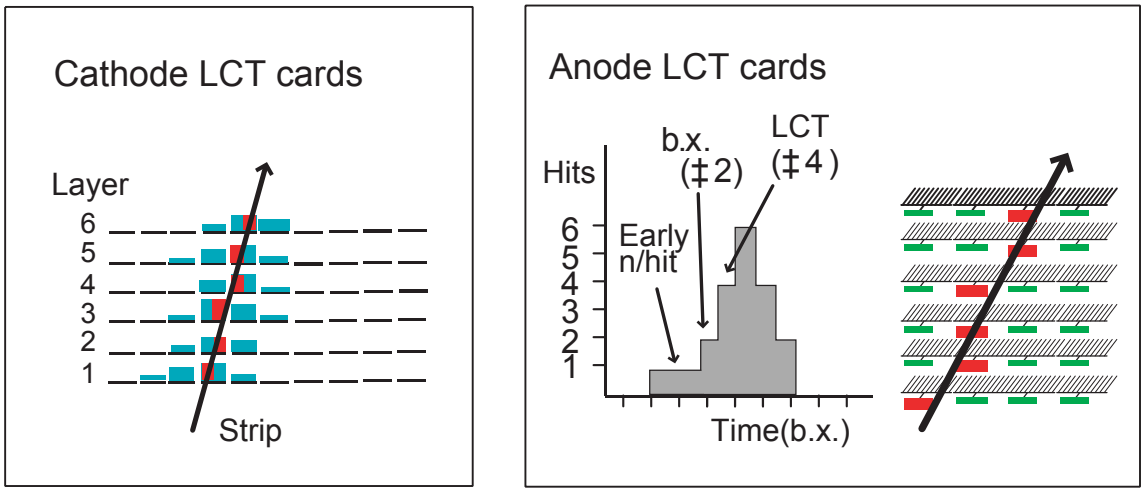


Figure F-12 Principle of the CSC local trigger.

incidence in LCT pattern circuitry to establish position of the muon to an RMS accuracy of 0.15 strip widths. Strip widths range from 6-16 mm.

Cathode and anode segments are brought into coincidence and sent to CSC Track Finder electronics which links the segments from the endcap muon stations. Each Track Finder unit finds muon tracks in a 60º sector. A single extrapolation unit forms the core of the Track-Finder trigger logic. It takes the three dimensional spatial information from two track segments in different stations, and tests if those two segments are compatible with a muon originating from the nominal collision vertex with a curvature consistent with the magnetic bending in that region. Each CSC Track Finder can find up to three muon candidates. A CSC muon sorter module selects the four best CSC muon candidates and sends them to the Global Muon Trigger.

F.2.4 RPC Trigger

The RPC pattern Trigger Logic (PACT) is based on the spatial and time coincidence of hits in four RPC muon stations (see Figure F-13). Because of energy loss fluctuations and multiple scattering there are many possible hit patterns in the RPC muon stations for a muon track of defined transverse momentum emitted in a certain direction. Therefore, the PACT should recognize many spatial patterns of hits for a given transverse momentum muon. In order to trigger on a particular hit pattern in the RPCs left by a muon, the PACT electronics performs two functions: requires time coincidence of hits in patterns ranging from 3-out-of-4 muon stations to 4 out of 6 muon stations along a certain road and assigns a p_T value. The coincidence gives the bunch crossing assignment for a candidate track. The candidate track is formed by a pattern of hits that matches with one of many possible patterns pre-defined for muons with defined transverse momenta. The p_T value is thus given. The pre-defined patterns of hits have to be mutually exclusive i.e. a pattern should have a unique transverse momentum assignment. The patterns are divided into classes with a transverse momentum value assigned to each of them. PACT is a threshold trigger; it gives a momentum code if an actual hit pattern is straighter than any of pre-defined patterns with a lower momentum code. The patterns will depend on the direction of a muon i.e. on ϕ and η.

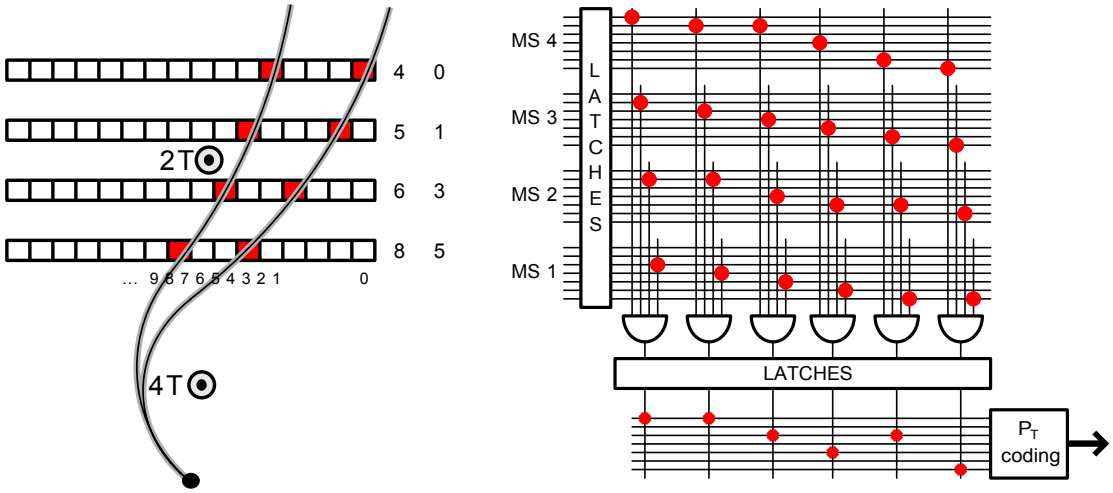


Figure F-13 RPC Trigger principle.

F.2.5 Global Muon Trigger (GMT)

The Regional Muon Trigger reconstructs muon candidates in both the barrel and the endcap regions out of hits or track segments found at the muon stations. For RPCs this is the Pattern Comparator Trigger (PACT) covering the entire η-region. The DTs and CSCs have separate track finders. The GMT receives the best four barrel DT and the best four endcap CSC muons and combines them with 4+4 muons sent by the RPC PACT. It performs a matching based on the proximity of the candidates in (η,ϕ) space. If two muons are matched their parameters are combined to give optimum precision. If a muon candidate cannot be confirmed by the complementary system quality criteria can be applied to decide whether to forward it. The GMT also contains logic to cancel ghost tracks that arise when a single muon is found by more than one muon system and is not otherwise matched, such as at the boundary between the DT and CSC muon systems. The selected muon candidates are ranked based on their transverse momentum, quality and to some extent pseudorapidity and the best four muon candidates in the entire CMS detector are sent to the Global Trigger.

The Global Muon Trigger also receives information from the calorimeters. The Regional Calorimeter Trigger sends two bits based on energy measurements representing isolation and compatibility with a minimum ionizing particle in $\Delta\eta \times \Delta\phi = 0.35 \times 0.35$ trigger regions. The GMT extrapolates the muon tracks back to the calorimeter trigger towers and appends the corresponding ISO (isolation) and MIP (minimum ionizing particle) bits to the track data consisting of p_T, sign of the charge, η, ϕ and quality.

F.2.6 Muon Trigger Performance

A simulation of the Level-1 Muon Trigger has been implemented in the ORCA software framework, with emphasis on performing operations that are exactly bit-compatible with the operations performed by those electronic boards for which prototypes and firmware exist.

The efficiency of the Global Muon Trigger to identify single muons as a function of the generated transverse momentum is shown in Figure F-14 for several Level-1 thresholds. The muons were generated flat in pseudorapidity in $|\eta|<2.1$. The Level-1 thresholds shown are defined to correspond to 90% efficiency. The sharpness of the turn-on curves is determined by the P_T resolution obtained by the GMT, which is

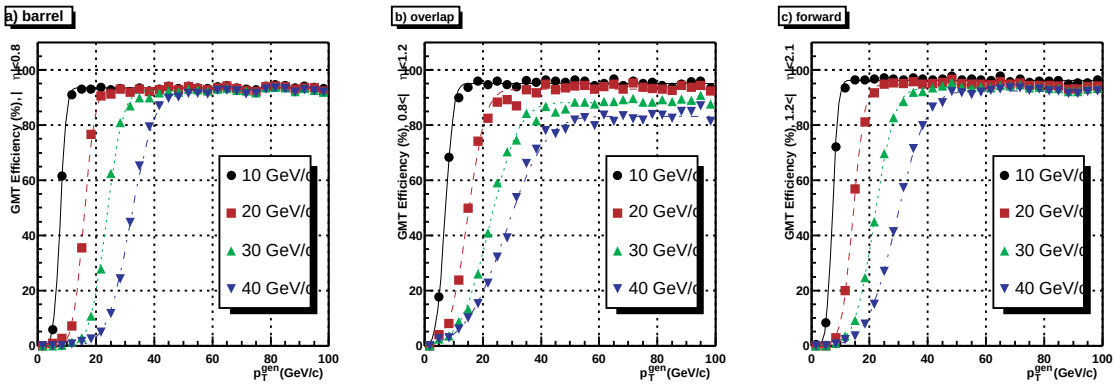


Figure F-14 Efficiency of the Level-1 Muon Trigger to identify single muons above several P_T thresholds as a function of the generated P_T, for three pseudorapidity intervals: a) $|\eta|<0.8$, b) $0.8<|\eta|<1.2$, and c) $1.2<|\eta|<2.1$.

shown explicitly in Figure F-15 by the distribution of the quantity $(1/P_T^{rec} - 1/P_T^{gen}) / (1/P_T^{gen})$, where P_T^{gen} and P_T^{rec} are the generated and reconstructed transverse momenta, respectively. Muons from *W* decays at high luminosity form the reference sample for the P_T resolution. The distributions are broken up into three pseudorapidity intervals: barrel ($|\eta|<0.8$), overlap ($0.8<|\eta|<1.2$) and endcap ($1.2<|\eta|<2.1$). The offset in the average value of these distribution from zero is due to the 90% efficiency threshold definition adopted by the Level-1 Trigger. Finally, Figure F-16 shows the efficiency of the GMT to reconstruct single muons from W decays at high luminosity as a function of pseudorapidity.

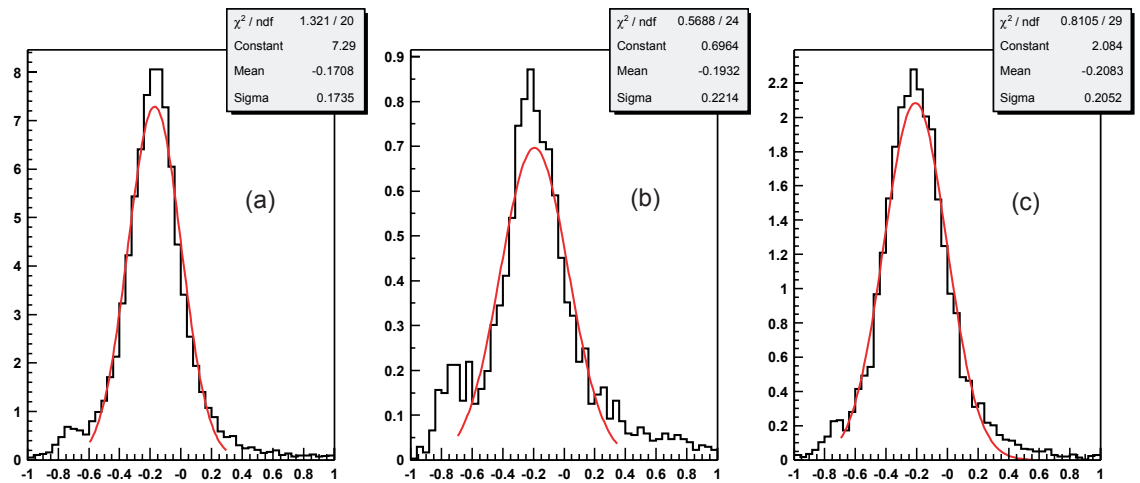


Figure F-15 Distribution of $(1/P_T^{rec} - 1/P_T^{gen}) / (1/P_T^{gen})$, where P_T^{gen} and P_T^{rec} are the generated and reconstructed (by the Level-1 Trigger) transverse momenta respectively, shown in three pseudorapidity intervals: a) $|\eta|<0.8$, b) $0.8<|\eta|<1.2$, and c) $1.2<|\eta|<2.1$.

The single muon trigger rate of the Global Muon Trigger as a function of the P_T threshold for both low and high luminosity is shown in Figure F-17. Also shown is the generated single muon rate and the trigger rates that would occur if the RPC system or the combined DT/CSC systems operated standalone, without the optimization of the GMT.

Contours of equal rate from the Level-1 single and di-muon triggers, in the plane of the applied P_T thresholds, are shown in Figure F-18 for low and high luminosity. The threshold for the di-muon trigger applies

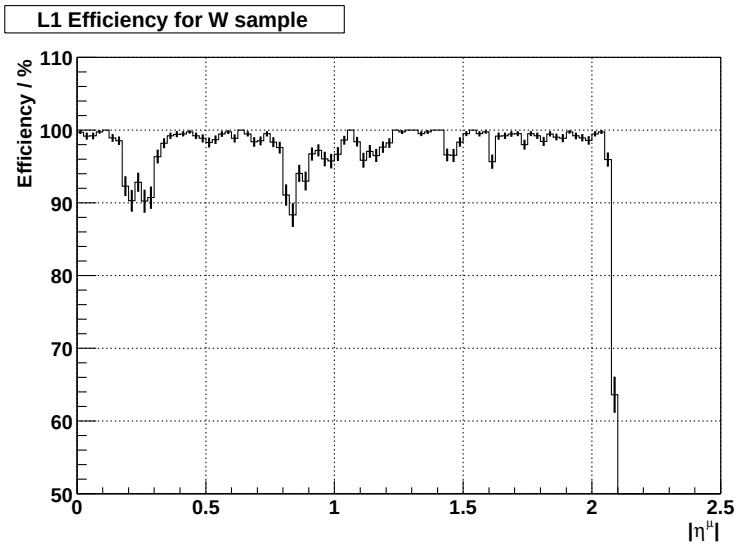


Figure F-16 Efficiency of the Level-1 Muon Trigger to identify single muons from *W* decays at high luminosity as a function of η.

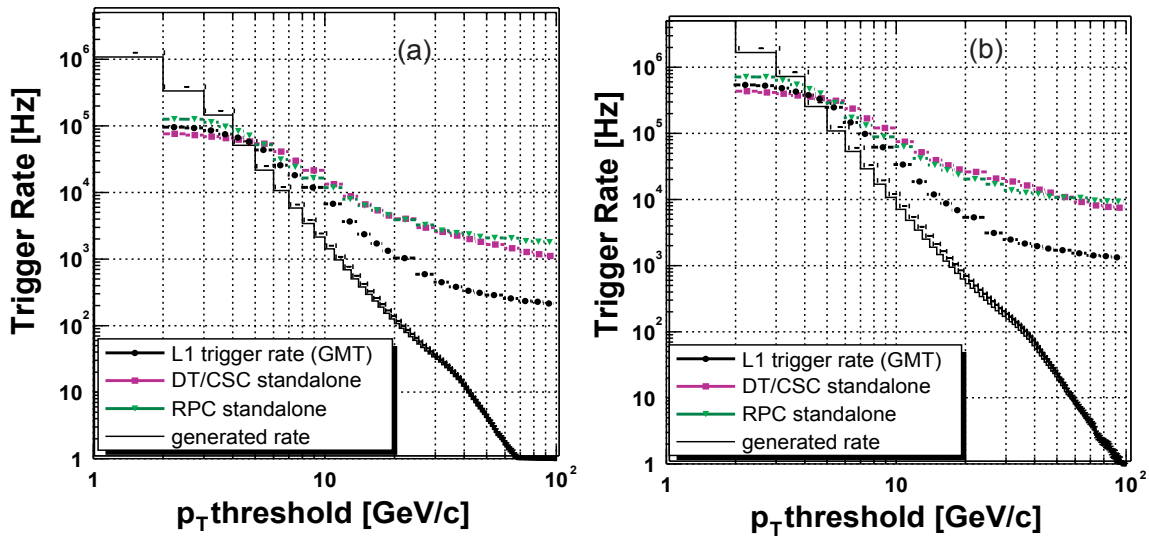


Figure F-17 Level-1 muon trigger rate as a function of P_T threshold for low (a) and high (b) luminosity.

to both muons. Assuming the allocation of Level-1 Triggers proposed in Section 15.1.3, a bandwidth of 4 kHz is available for muon triggers at low luminosity, 8 kHz at high luminosity. The operating thresholds discussed in Section 15.3.4.2 are 14 GeV/c and 3 GeV/c for the single and di-muon triggers at low luminosity, and 20 GeV/c and 5 GeV/c at high luminosity.

F.3 Global Trigger

The Global Trigger accepts muon and calorimeter trigger information, synchronizes matching sub-system data arriving at different times and communicates the Level-1 decision to the timing, trigger and control

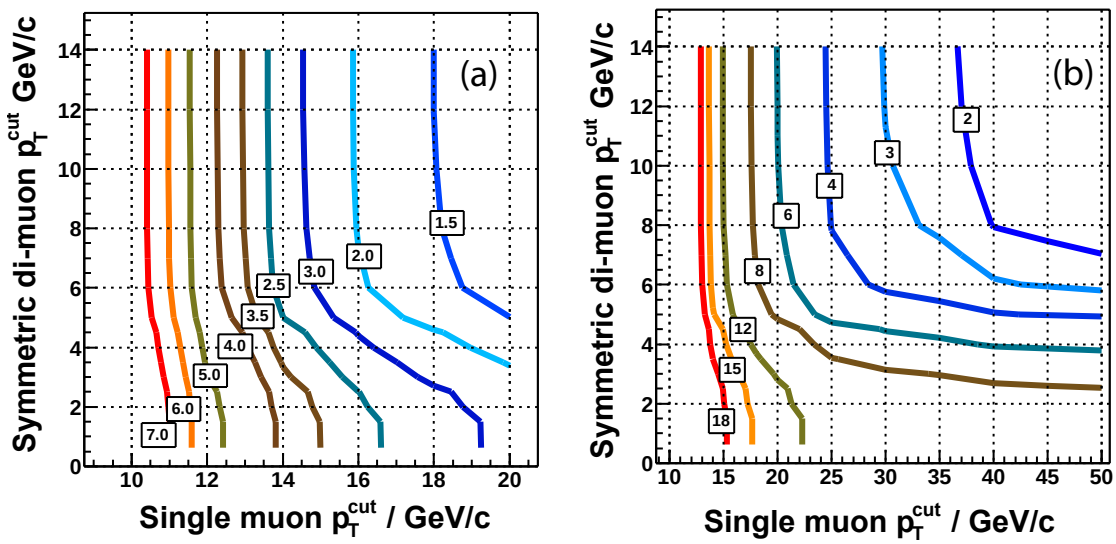


Figure F-18 Contours of equal rate in the plane of P_T thresholds for Level-1 single and di-muon triggers at (a) $L=2\times10^{33}$ cm^{-2}s^{-1} and (b) $L=10^{34}$ cm^{-2}s^{-1}.

system for distribution to the sub-systems to initiate the readout. The global trigger decision is made using logical combinations of the trigger data from the Calorimeter and Muon Global Triggers.

The Level-1 Trigger system sorts ranked trigger objects, rather than histogramming objects over a fixed threshold. This allows all trigger criteria to be applied and varied at the Global Trigger level rather than earlier in the trigger processing. All trigger objects are accompanied by their coordinates in (η,ϕ) space. This allows the Global Trigger to vary thresholds based on the location of the trigger objects. It also allows the Global Trigger to require trigger objects to be close or opposite from each other. In addition, the presence of the trigger object coordinate data in the trigger data, which is read out first by the DAQ after a L1A, permits a quick determination of the regions of interest where the more detailed HLT analyses should focus. Besides handling physics triggers, the Global Trigger provides for test and calibration runs, not necessarily in phase with the machine, and for prescaled triggers, as this is an essential requirement for checking trigger efficiencies and recording samples of large cross section data.

The Global Level-1 Trigger transmits a decision to either accept (L1A) or reject each bunch crossing. This decision is transmitted through the Trigger Throttle System (TTS) to the Timing Trigger and Control system (TTC). The TTS allows the reduction by prescaling or shutting off of L1A signals in case the detector readout or DAQ buffers are at risk of overflow.

F.4 References

F-1 CMS Coll.,“The Trigger and Data Acquisition project, Volume I, The Level-1 Trigger, Technical Design Report”, CERN/LHCC 2000-038, CMS TDR 6.1, 15 December 2000.

F-2 M. Aguilar-Benitez at al., “Construction and Test of the final CMS Barrel Drift Tube Muon Chamber Prototype”, Nucl. Instr. and Meth. A480 (2002) 658

G Extending the Use of Tracking in the HLT

The systematic optimization of the track reconstruction code and the extensive use of “regional” and “conditional” track reconstruction, described in Section 14.4.3, allow fast and robust tagging and detailed characterization of electrons, muons and tau's, as well as of *b*-jets with displaced vertices, with sufficient speed to be used for a large fraction of, and possibly the full, Level-1 event rate, at both low and high luminosity.

The optimization of track reconstruction in the HLT has made rapid progress in the past year, and is still ongoing. In what follows, ways in which extended use of track reconstruction may be applied to further improve the CMS HLT filters are illustrated with a few specific examples.

The HLT strategy described below does not use the standard Level-2 muon selection and instead uses the partial tracking algorithms described in Section 14.4.3. The main components are the same as the standard track reconstruction: fast and efficient pixel reconstruction, aggressive track seed filtering using primary vertex information and Level-1 muon trigger information; partial track reconstruction using filtered seeds. For *B* physics, vertex and mass constraints are then used for the final event selection, while for b-jet tagging, the corresponding tagging algorithm is used.

G.1 Benchmark Channels for Inclusive jet *b*-tagging

Final states which contain *b*-jets have been proposed as very promising Higgs discovery channels. In this section the selection efficiencies using a *b*-tag right after the Level-1 selection for two channels is studied:

- $WH \rightarrow \mu\nu bb$, with a Higgs mass of 115 GeV/c^2, triggered at Level-1 by a high-P_T muon
- fully hadronic decays of $t\bar{t}H \rightarrow bjj\bar{b}jjb\bar{b}$, with a Higgs mass of 120 GeV/c^2

The cross sections and rates of these channels, together with those of the main backgrounds expected, are given in Table G-1. The main process which contributes to the rate for the first channel derives from the $Wjj \rightarrow \mu jjX$ production, in which the muon can come either from the direct leptonic *W* decay as well as from other muons present in the event, such as for instance *c*-quark semi-leptonic decays. This process has a cross section of about 16 nb. Despite the larger cross section, the $Z/\gamma \rightarrow \mu jjX$ process has less impact due to its softer muon P_T spectrum. For the $t\bar{t}H$ channel the main background arises from QCD jets. This background can be reduced by requiring the presence of at least two or three identified *b*-jets.

G.1.1 *WH* with *H* Decaying to Jets

The study has been performed at low luminosity for the staged pixel detector case. Events are triggered at Level-1 by a muon with P_T>10 GeV/c at least two Level-1 jets with E_T>20 GeV and $|\eta|$<5, giving a total rate of 2.2 kHz. A muon track is then searched for directly in the tracker using a regional approach. Regional seeds are made of combinations of pixel hits in a cone of size $|\Delta\eta|$<0.1 and $\Delta\phi$<0.3 around the Level-1 muon direction, centered in the origin with a tolerance of 15 cm along the z direction, a maximum 2D impact parameter of 100 μm and a P_T>10 GeV/c. For the signal events there is almost always a single candidate track and the efficiency to find such a track is about 90%. The muon track z impact point defines the z coordinate for the primary vertex of the event with 50 μm precision. Muon isolation is then required: tracks with P_T>1 GeV/c are reconstructed within a cone of $|\Delta\eta|$<0.2 and $\Delta\phi$<0.2 around the muon direction, compatible from coming within a cylinder of 1 mm diameter and length, centered on the previously found primary vertex. The event is rejected if the P_T sum of the tracks found, excluding the muon

Table G-1 Cross sections and rates for different final states, together with the kinematic cuts applied at generation level.

Process	Kinematic cuts			Cross section	Rate at 2×10^{33} cm^{-2} s^{-1} (Hz)
	P_T^{μ} (GeV/c)	$\lvert\eta^{\mu}\rvert$	P_T^{hard} (GeV/c)		
WH→μνbb				0.131 pb	$2.62\ 10^{-4}$
ttH→bjjbjjbb				0.36 pb	$7.2\ 10^{-4}$
Wjj→μjjX	>3	<2.5		16 nb	32.5
Z/γ→μjjX	>3	<2.5		22 nb	44.2
QCD 1 GeV/c< P_T^{μ}< 4 GeV/c	1 - 4	<2.5	15-50, >170	17 μb	$34\ 10^3$
QCD 4 GeV/c< P_T^{μ}< 10 GeV/c	4 - 10	<2.5	15-50, >170	9.8 μb	$20\ 10^3$
QCD P_T^{μ}> 10 GeV/c	>10	<2.5	15-50, >170	0.53 μb	$1.1\ 10^3$
QCD 0<P_T^{hard}<15			0 - 15	52.4 mb	$1\ 10^8$
QCD 50<P_T^{hard}<80			50 - 80	24 μb	$48\ 10^3$
QCD 80<P_T^{hard}<120			80 - 120	3.4 μb	$6.8\ 10^3$
QCD 120<P_T^{hard}<170			120 - 170	0.57 μb	$1.1\ 10^3$

track, exceeds 3 GeV/c. The time for the two above steps is less than 150 ms on an Intel 1 GHz Pentium-III CPU. Two calorimetric jets are further required to be within the tracker acceptance. Tracks of P_T>2 GeV/c and a maximum of 7 hits are then reconstructed in cone of $\Delta\eta$<0.4 and $\Delta\phi$<0.4 around the jet directions and compatible with coming from the primary vertex. A simple *b*-tag algorithm is applied to the jets, as outlined in Section 15.6.1, which requires two tracks with a 2D impact parameter significance larger than 2. The time to reconstruct those tracks and apply the *b*-tag is about 130 ms per jet. Details can be found in reference [G-1].

The main background comes from QCD events in which at least one muon triggered the Level-1 selection. Figure G-1 shows the rate for minimum-bias events and the efficiency for the signal events as a function of the single muon threshold as measured by the muon Level-1. Already at this stage, for a muon P_T threshold of 15 GeV/c, the background rate is decreased from 2.7 kHz to about 1.6 Hz while a global efficiency of 13% for at least one *b*-jet tagged is maintained.

The above selection is also very efficient for top-quark semileptonic decays and can be used as an alternative with respect to the inclusive single-muon selection described in Section 15.3.3, if two calorimetric jets are present. In particular, taking into account that the *Wjj* cross section with two jets with E_T>15 GeV within the tracker acceptance is about one third of the inclusive *W* cross section, the use of the above selection might allow to redirect the muon HLT resources to other interesting channels [G-1].

G.1.2 $t\bar{t}H$ Fully Hadronic Decays

Final states containing many *b*-jets have an enormous background from QCD multi jet production. When a muon from the *W* decay is present, this channel can be easily selected at trigger level. Nevertheless, preliminary studies have shown [G-2] a valuable increase in the discovery potential by using in addition the fully hadronic $t\bar{t}H$ decays, which instead must be triggered against the large QCD background.

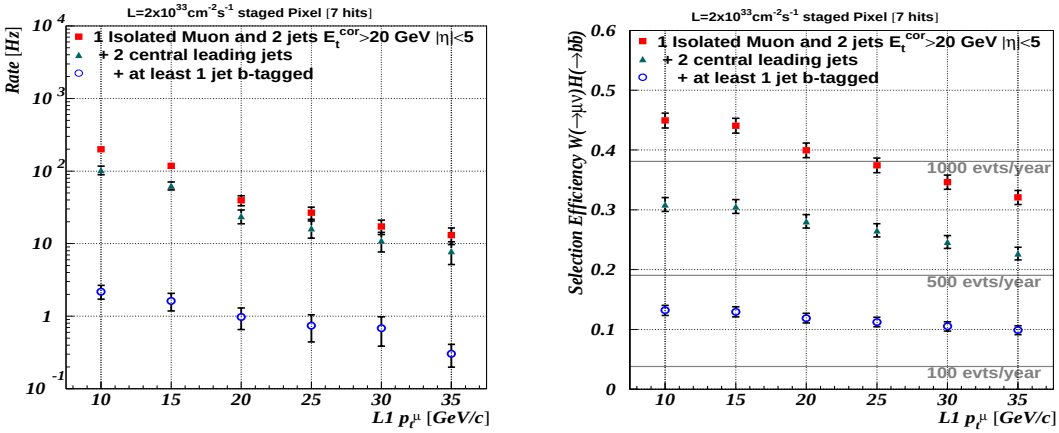


Figure G-1 Rate for minimum-bias events (left) and efficiency for signal events (right) as a function of the single muon threshold as measured by the muon Level-1 Trigger.

At Level-1 events are selected using a calorimetric jet trigger. The efficiency for this channel at Level-1, as a function of the cut on the calibrated jet E_T, for jets with $|\eta| < 2.5$, is shown in Figure G-2 (left). It is possible to achieve 25% efficiency on the signal by requiring 4 jets with E_T >75 GeV and a rate, at low luminosity, of ~10 Hz (Figure 15-39).

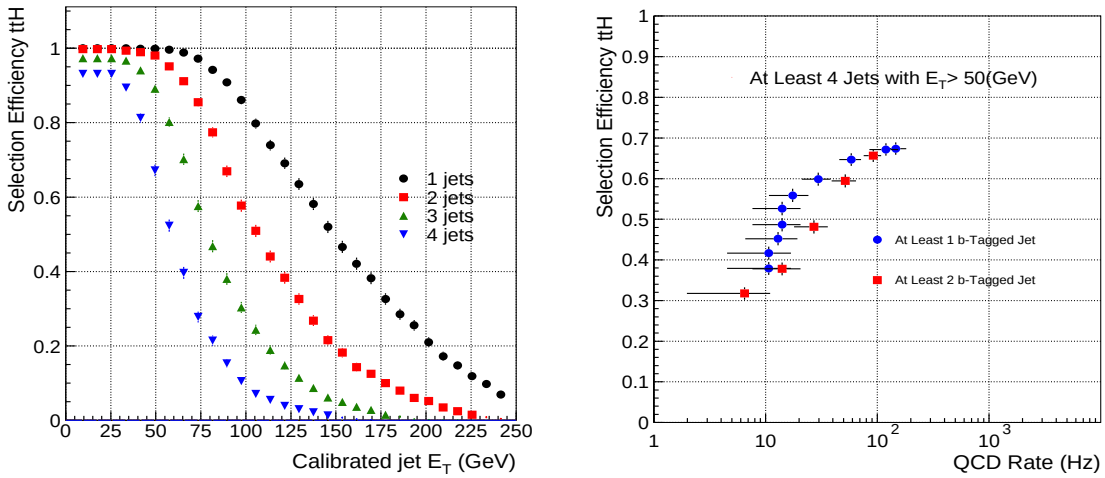


Figure G-2 Efficiency to select $t\bar{t}H$ events. On the left: as a function of the calibrated jet E_T of the first four jets within the tracker acceptance. On the right: as a function of the QCD rate. The two curves show the effect of *b* - tagging at least one jet or at least two jets in the event. The different points refer to different *b* - tag conditions.

The HLT selection requires a number of jets to be identified as *b*-jets. The jets are preselected requiring E_T>50 GeV. The primary vertex is reconstructed using pixel lines with P_T>2 GeV/c. The region of interest is defined as a cone of 0.4 around the jet direction, while partial track reconstruction stops when give hits are found. The *b*-tag algorithm is the one described in Section 15.6, which requires two tracks with large significance on the transverse impact parameter measurement. Figure G-2 (right) shows the efficiency for the signal selection versus the background rate, at low luminosity, for different cuts on the transverse impact parameter significance of the track. It is possible to reduce the QCD rate to about 10 Hz requiring a *b*-tag on the leading jet with 50% of signal efficiency [G-3].

G.2 Exclusive *B*-physics Triggers

Due to the large $b\bar{b}$ cross section, even a fully efficient *b*-tag cannot retain all *b*-hadron decays to allow the selection of exclusive decays. Instead such decays must be reconstructed within the large background. The *B*-physics selection is triggered at Level-1 by the presence of one or two muons. The aim of the HLT selection is to reconstruct the decays either of the associated *b* hadron or those of the two muons (such as in the case of the J/ψ decays), by using invariant mass and displaced vertices cuts.

Results on three channels, whose selection relies on track reconstruction after a Level-1 muon trigger are shown:

- $B_s \rightarrow J/\psi\phi \rightarrow \mu\mu KK$
- $B_s \rightarrow \mu\mu$
- $b \rightarrow \mu + B_s \rightarrow \pi\, D_s \rightarrow \phi(KK)\pi$.

From the results of the first and second channel, an inclusive trigger path for $J/\psi \rightarrow \mu\mu$ is also derived.

Table G-2 lists the data samples used for the signal and background. To save CPU time, events were generated and selected with the following requirements:

- within the Pythia generator the P_T^{hard} of the hard interaction has to be larger than 1 GeV/c;
- in the case of the fully hadronic channel, one *b*-hadron is forced to decay into a muon;
- The second *b*-hadron is forced to decay via the signal channel, and its decay products are required to satisfy the P_T cuts listed in Table G-2.

Table G-2 Cuts at generator level and cross sections for the samples used.

Kinematic cuts	$B_s \rightarrow J/\psi\ \phi \rightarrow \mu\mu KK$	$B_s \rightarrow \mu\mu$	$b \rightarrow \mu + B_s \rightarrow \pi D_s \rightarrow \phi(KK)\pi$	**Inclusive muon sample**
$p^{\mu}_T p^{h}_T \lvert\eta\rvert$	2 GeV/c, 0.5 GeV/c, 2.5	2 GeV/c, -,2.5	4 GeV/c, 0.5 GeV/c, 2.5	4 GeV/c
cross section (fb)	$97\ 10^3$	92	$13\ 10^3$	$24\ 10^{12}$

The algorithms for the reconstruction of the primary vertex and charged particle tracks were developed for inclusive jets are thus not optimal for use with the relatively low-E_T *b*-jets involved in typical *B* samples. In particular, the minimum P_T of the decay tracks needs to be reconstructed from about 700 MeV/c. In order to limit the large number of seeds for this low P_T value only those seeds which are close to the z coordinate of the interaction point that produced the $b\bar{b}$ pair are considered. The seeds are reconstructed using the Fast Track seed generator based on pixel detectors.

G.2.1 *B* Decays to Muon Final States

Events are triggered at Level-1 requiring at least two muon candidates with P_T>4 GeV/c and opposite charge. Two slightly different trigger strategies for $B_s \rightarrow \mu\mu$ and $B_s \rightarrow J/\psi\phi$ are described below.

G.2.1.1 $B_s \rightarrow \mu\mu$ Selection

Events are required to have two oppositely-charged isolated muons coming from a common displaced vertex and having an invariant di-muon mass close to the nominal B_s mass. At a first stage, the fast seed generator reconstructs all pixel hit pairs which can be assigned to the tracks with $P_T > 4$ GeV/c and transverse impact parameter smaller than 0.1 cm. The reconstructed track pairs are used for a first reconstruction of the primary vertex. The three most significant primary vertices are retained per event. At a second stage hit pairs are filtered using the vertex constraint and the regions defined by the Level-1 muons ($\Delta\eta<0.5$ and $\Delta\phi<0.8 \pm 0.4$ correction). Filtered hit pairs are then used for the reconstruction of tracks. This reconstruction is stopped when one of the following three conditions are met: (a) the track P_T is lower than 4 GeV/c at 5 σ, or (b) 6 hits along the trajectory are found, or if (c) the relative resolution on P_T is smaller than 2%. The reconstructed tracks are back propagated to the origin using the standard smoothing procedure. If and only if two opposite charged tracks are reconstructed, the algorithm checks the invariant mass which has to be in 150 MeV/c^2 from the B_s mass (Figure G-3). In order to suppress combinatorial

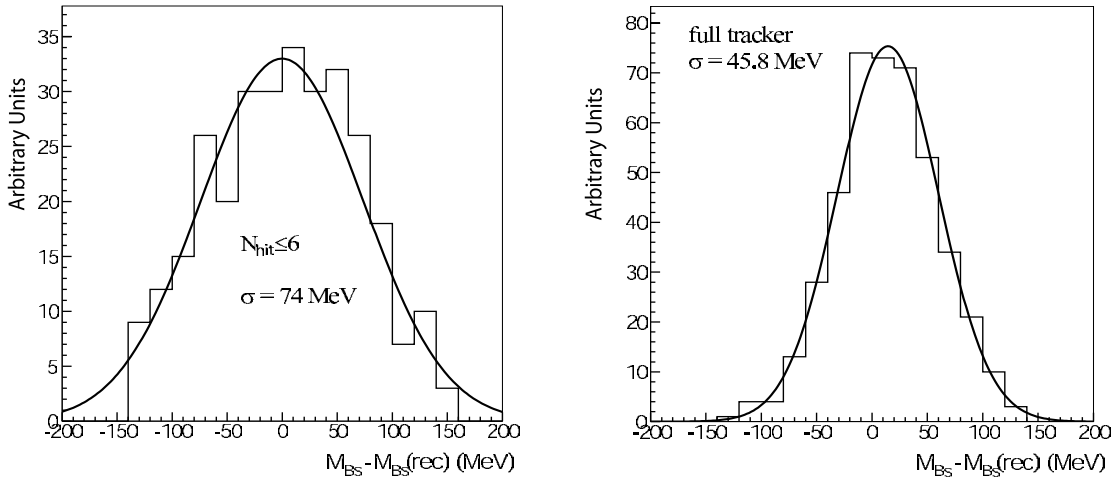


Figure G-3 Invariant mass distribution for $B_s \rightarrow \mu\mu$ decays. Left: in the HLT; right: offline reconstruction.

background, tracks are vertexed and retained if the χ^2 is smaller than 20 and the decay length distance in the transverse plane is larger than 150 μm. The algorithm described uses all the ingredients required by the offline selection [G-4] but with much softer cuts to keep the useful events for further analysis. The global selection efficiency on the signal events is 5% and the timing is on average 240 ms on a 1 GHz Intel Pentium-III CPU, so it could be implemented at the HLT stage. The estimated background rate is below 2 Hz (Table G-3).

Table G-3 Trigger efficiencies, number of events/10 fb^{-1} and background rate for $B_s \rightarrow \mu\mu$ reconstruction (see text). The global efficiency is the efficiency of the combined Level-1 and HLT selections.

L1 Trigger Efficiency	HLT efficiency	Global Efficiency	Events/10 fb^{-1}	Trigger Rate
15.2%	33.5%	5.1%	47	<1.7 Hz

G.2.1.2 $B_s \rightarrow J/\psi\,\phi \rightarrow \mu\mu$ KK Selection

Muons from J/ψ decays are reconstructed in the same way as for the ones of the $B_s \rightarrow \mu\mu$ reconstruction, except the fact the a slightly tighter cuts on the di-muon mass and secondary vertex are required to keep

the background rate at acceptable level. The mass window is decreased to 100 MeV/c^2 (Figure G-4) and the secondary vertex has to be at least 200 μm from the beam with a vertex $\chi^2 < 10$.

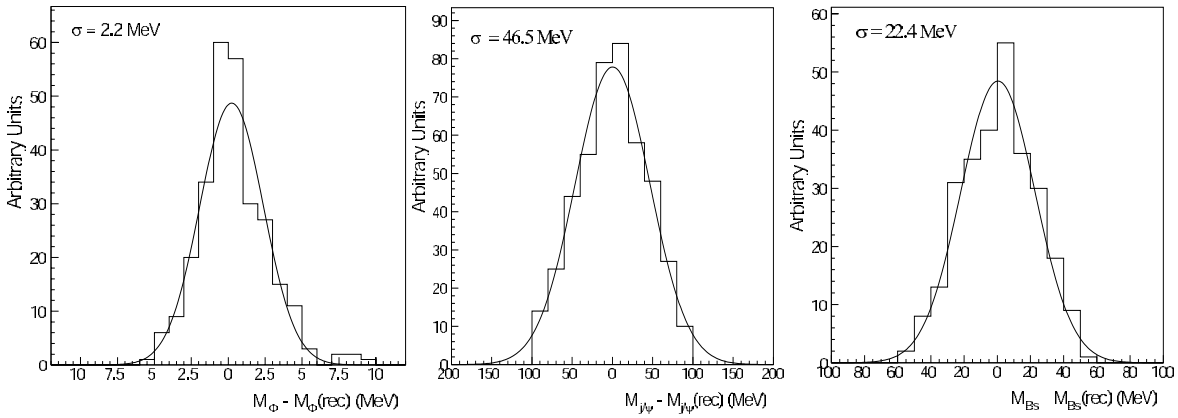


Figure G-4 Mass resolutions for φ, J/ψ and B_s signal candidates.

This di-muon selection leads to an inclusive rate of about 15 Hz for low luminosity, 90% of which composed of J/ψ from b decays. This selection is sufficiently inclusive to allow further offline selections, such as identifying $J/\psi K^0_s$ decays. This first stage takes on average 260 ms and can be run on every event passing the Level-1 Trigger. The full reconstruction of the B_s is more demanding in CPU time and therefore must be run at Level-3 to reduce the rate below 1 Hz. To select the kaon tracks coming from the φ meson, all track seeds within a cone of $\Delta R<1.5$ around the reconstructed J/ψ direction are reconstructed up to 6 hits or the relative uncertainty on P_T becomes smaller than 2%. Charged tracks of opposite sign are then paired and retained if the invariant mass is within 10 MeV/c^2 of the φ mass. Once a candidate is found the invariant mass of the J/ψ and φ system is computed and the B_s candidate is retained if it falls within 60 MeV/c^2 of the B_s mass. To further decrease the background, the B_s vertex is required to satisfy $\chi^2<200$.

Table G-4 shows the efficiencies and yields for signal events together with the rates from backgrounds at low luminosity. The average execution time for signal and background events on a 1 GHz Intel Pentium-III CPU is about 260 ms for the reconstruction of the J/ψ and 800 ms for the reconstruction of the full B_s, out of which about 400 ms is spent on the reconstruction of the φ.

Table G-4 Trigger efficiencies, number of events/10 fb^{-1} and background rate for $B_s \rightarrow J/\psi\,\phi$ reconstruction.

Level-1 Muon Efficiency	Level-2 Efficiency	Level-2 Rate	Level-3 Efficiency	Level-3 Rate	Events/10 fb^{-1}
16.5%	13.7%	14.5 Hz	8.7%	<1.7 Hz	83800

G.2.1.3 $b \rightarrow \mu + B_s \rightarrow \pi + D_s \rightarrow \phi(KK)\pi$ Selection

This study has been performed for low luminosity. The Level-1 Trigger is based either on a single muon or on the combination of a low-P_T muon and a relatively E_T jet. Table G-5 shows the expected trigger rates as a function of the different cuts on the muon and the soft jet. Table G-6 shows the yields for signal events. The machine conditions and the instantaneous luminosity may well allow running the Level-1 Trigger using lower thresholds than the nominal ones, which have been determined for a luminosity of for

discovery. The HLT trigger strategy is based mainly on the reconstruction of the different resonances in the decay.

Table G-5 HLT (Level-1) rates, in kHz, as a function of the cuts on the muon and the jet.

Muon p_T (GeV/c)	Rate (kHz)		
	E_T>0 GeV	E_T>20 GeV	E_T>30 GeV
4	0.27 (50)	0.15 (15)	0.08 (5.7)
5	0.19 (33)	0.10 (11)	0.06 (4.2)
6	0.16 (26)	0.082 (8.5)	0.055 (3.6)
7	0.11 (18)	0.062 (6.2)	0.045 (2.7)
10	0.037 (6.4)	0.021 (2.5)	0.014 (1.3)
14	0.017 (3.2)	0.010 (1.3)	0.008 (0.7)

Table G-6 Number of signal events after HLT (Level-1) trigger as a function for different values of the Level-1 selection cuts for an integrated luminosity of 20 fb^{-1}.

Muon p_T (GeV/c)	Number of events (1000)		
	E_T>0 GeV	E_T>20 GeV	E_T>30 GeV
4	11K (106 K)	4.8 K (34 K)	2.3 K (14 K)
5	8 K (78 K)	3.6 K (28 K)	2 K (12 K)
6	7.4 K (65 K)	3.1 K (24 K)	1.5 K (11 K)
7	5.6 K (48 K)	2.4 K (20 K)	1.3 K (9.4 K)
10	2.0 K (20 K)	1.0 K (9.6 K)	0.6 K (5.3 K)
14	1K (11 K)	0.5 K (5.4 K)	0.3 K (3. K)

The pixel primary vertex finder is reconstructed as in Section 14.4.2. Tracks are reconstructed with P_T>0.7 GeV/c and with 3 hits only: two in the pixel detector and one in the first layer of the silicon detector, to allow for sufficiently large lever arm. To reduce execution time, seeds are requested to come from the primary vertex z location ±1 mm. A search is then made for the ϕ, D_s and B_s candidates using the following three requirements: $\Delta R(KK)<0.3$, $\Delta R(\phi\pi)<1.2$ and $\Delta R(D_s\pi)<3.0$.

Figure G-5 shows the invariant mass resolutions for the ϕ, D_s and B_s candidates. The main cuts are based on invariant mass and the p_T of the three resonances. A 3D reconstruction of the D_s vertex is also performed. The details can be found in reference [G-4].

The signal efficiency after all these cuts is about 10%, while the background is suppressed by a factor 250. The mean execution time for signal and background events, referred to a single 1 GHz Intel Pentium-III CPU, is 640 ms. The number of signal events will depend on the allowed bandwidth given to this channel, according to the instantaneous luminosity of LHC. From the above tables even a relatively large threshold of 14 GeV/c gives a rate of 3 kHz after Level-1. Since the allowed bandwidth would be lower than this, it is necessary to scale the rate, which can be either done scaling the rate of the lowest possible muon P_T bin or scale each bin in P_T such that the overall sum matches the allowed band. On the other

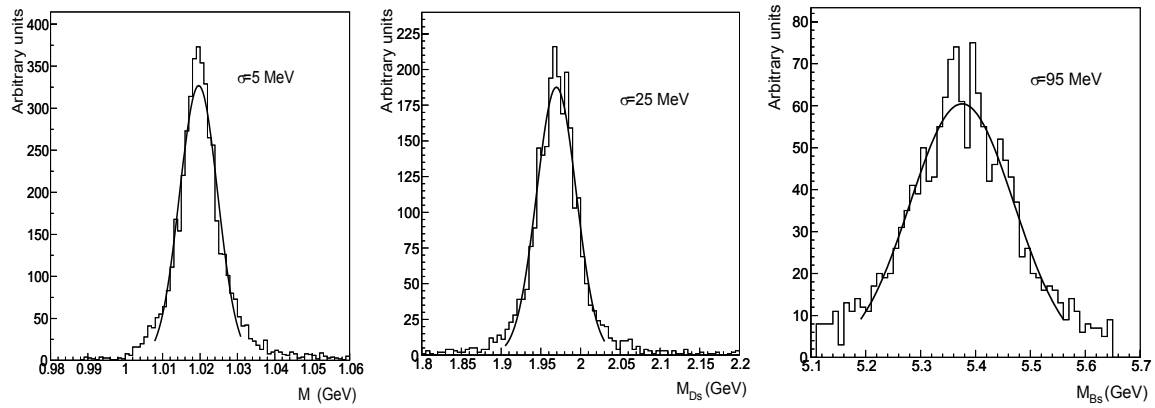


Figure G-5 Invariant mass distribution of ϕ, D_s and B_s candidates.

hand the quality of the analysis will depend on the total number of events, the fraction of signal events and their dilution factor [G-4]. By rising the muon P_T the signal fraction and dilution increases mildly [G-5], while the total number of events decreases. Therefore it is better to fill the allowed bandwidth with events with the lowest possible muon P_T and downscale only the bin which would be needed for filling the band. For example, referring to Table G-5, for an allowed bandwidth of 5 kHz, all events with muon P_T>14 GeV/c shoud be kept, while events with 10 GeV/c<P_T<14 GeV/c should be scaled by a factor two.

G.3 References

G-1 R. Ranieri, “A High-Level Trigger Selection of $WH \rightarrow \mu\nu b\bar{b}$ using the Tracker and *b*-jets”, CMS Note 2002/048.

G-2 V. Drollinger, PhD Thesis, IEKP, Karlsruhe University, Germany. CMS 2001-028/THESIS.

G-3 D. Benedetti, L. Fano, “Study of High Level b-trigger Selection of ttH Fully Hadronic Decays”, CMS Note 2002/044.

G-4 A. Giassi, F. Palla and A. Starodumov, “Study for a High-Level Trigger for the decay channel $B_s \rightarrow \pi + D_s \rightarrow \phi(KK)\pi\pi$ ”, CMS Note 2002/045.

G-5 Z. Xie, PhD Thesis, Scuola Normale Superiore, Pisa, Italy. CMS 2001-018/THESIS.

CMS Trigger/DAQ project: Members and Contributors

Current participants in the CMS TriDAS Collaboration, as well as contributors to the High-Level Trigger, by Country and Institute.

Institut für Hochenergiephysik der OeAW, Wien, AUSTRIA
W. Adam, H. Bergauer, C. Deldicque, J. Erö, R. Fruehwirth, A. Jeitler, K. Kastner, S. Kostner, N. Neumeister[1], M. Padrta, P. Porth, H. Rohringer, H. Sakulin, J. Strauss, A. Taurok, G. Walzel, C.-E. Wulz

Vrije Universiteit Brussel, Brussel, BELGIUM
S. Lowette, B. Van De Vyver[1]

Université Libre de Bruxelles, Bruxelles, BELGIUM
G. De Lentdecker, P. Vanlaer

Université Catholique de Louvain, Louvain-la-Neuve, BELGIUM
C. Delaere, V. Lemaitre, A. Ninane, O. van der Aa

Institute for Nuclear Research and Nuclear Energy, Sofia, BULGARIA
J. Damgov

Helsinki Institute of Physics, Helsinki, FINLAND
K. Banzuzi, V. Karimäki, R. Kinnunen, T. Lampén, K. Lassila-Perini, V. Lefébure[1], J. Nysten, E. Pietarinen[2], D. Ungaro, M. Voutilainen

Laboratoire Leprince-Ringuet, Ecole Polytechnique, IN2P3-CNRS, Palaiseau, FRANCE
P. Busson

Institut de Recherches Subatomiques, IN2P3-CNRS - ULP, LEPSI Strasbourg, UHA Mulhouse, Strasbourg, FRANCE
T. Todorov

RWTH, I. Physikalisches Institut, Aachen, GERMANY
B. Schwering

Institut für Experimentelle Kernphysik, Karlsruhe, GERMANY
P. Gras

University of Athens, Athens, GREECE
G. Daskalakis, A. Sfyrla

Institute of Nuclear Physics "Demokritos", Attiki, GREECE
M. Barone, T. Geralis, C. Markou, K. Zachariadou

KFKI Research Institute for Particle and Nuclear Physics, Budapest, HUNGARY
P. Hidas

Tata Institute of Fundamental Research - EHEP, Mumbai, INDIA
S. Banerjee, K. Mazumdar[1]

Tata Institute of Fundamental Research - HECR, Mumbai, INDIA
S. Banerjee

Università di Bari, Politecnico di Bari e Sezione dell' INFN, Bari, ITALY
M. Abbrescia, A. Colaleo, M. D'Amato, N. De Filippis, D. Giordano, F. Loddo, M. Maggi, L. Silvestris, G. Zito

Università di Bologna e Sezione dell' INFN, Bologna, ITALY
S. Arcelli, D. Bonacorsi, P. Capiluppi, G.M. Dallavalle, A. Fanfani, C. Grandi, S. Marcellini, A. Montanari, F. Odorici, R. Travaglini

Università di Catania e Sezione dell' INFN, Catania, ITALY
S. Costa, A. Tricomi

Università di Firenze e Sezione dell' INFN, Firenze, ITALY
V. Ciulli, N. Magini, R. Ranieri

Laboratori Nazionali di Legnaro dell' INFN, Legnaro, ITALY (associated institute)
L. Berti, M. Biasotto, M. Gulmini[1], G. Maron, N. Toniolo, L. Zangrando

Università di Padova e Sezione dell' INFN, Padova, ITALY
M. Bellato, U. Gasparini, S. Lacaprara, P. Ronchese, S. Ventura

Università di Perugia e Sezione dell' INFN, Perugia, ITALY
D. Benedetti, M. Biasini, L. Fanò, L. Servoli

Università di Pisa, Scuola Normale Superiore e Sezione dell' INFN, Pisa, ITALY
G. Bagliesi, T. Boccali, S. Dutta, S. Gennai, A. Giassi, F. Palla, G. Segneri, A. Starodumov[3]

Università di Roma I e Sezione dell' INFN, Roma, ITALY
P. Meridiani, G. Organtini

Università di Torino e Sezione dell' INFN, Torino, ITALY
N. Amapane, F. Bertolino, R. Cirio

Chonnam National University, Kwangju, KOREA
J.Y. Kim, I.T. Lim, K.J. Ma

Seoul National University, Seoul, KOREA
K.K. Joo, B.J. Kim, S.B. Kim, I.H. Park

Sungkyunkwan University, Suwon, KOREA
B.G. Cheon, Y.I. Choi

Kyungpook National University, Taegu, KOREA
K. Cho, S.W. Ham, D.H. Kim, G.N. Kim, W.Y. Kim, J.Y. Lee, S.K. Oh, Y.D. Oh, S.R. Ro, D.C. Son

National Centre for Physics, Quaid-I-Azam University, Islamabad, PAKISTAN
A. Osman[1]

Institute of Experimental Physics, Warsaw, POLAND
K. Bunkowski, M. Cwiok, W. Dominik, K. Doroba, M. Kazana, J. Krolikowski, I. Kudla, M. Pietrusinski, K. Pozniak[8], W. Zabolotny[8], P. Zych

Soltan Institute for Nuclear Studies, Warsaw, POLAND
M. Górski, L. Goscilo, G. Wrochna, P. Zalewski

Laboratório de Instrumentação e Física Experimental de Partículas, Lisboa, PORTUGAL
R. Alemany-Fernandez, C. Almeida, N. Almeida, J. Da Silva, M. Santos, I. Teixeira, J.P. Teixeira, J. Varela, N. Vaz Cardoso

Joint Institute for Nuclear Research, Dubna, RUSSIA
V. Konoplyanikov, A. Urkinbaev

Institute for Nuclear Research, Moscow, RUSSIA
A. Toropin[4]

Institute for Theoretical and Experimental Physics, Moscow, RUSSIA
V. Gavrilov, V. Kolosov, A. Krokhotine, A. Oulianov, N. Stepanov

Moscow State University, Moscow, RUSSIA
O.L. Kodolova[1], I. Vardanyan

Vinca Institute of Nuclear Sciences, Belgrade, SERBIA
J. Ilic, G. Skoro

Universidad Autónoma de Madrid, Madrid, SPAIN
C. Albajar, J.F. Trocóniz

Instituto de Física de Cantabria (IFCA), CSIC-Universidad de Cantabria, Santander, SPAIN
A. Calderon, M.A. Lopez Virto, R. Marco, C. Martinez Rivero, F. Matorras, I. Vila

Universität Basel, Basel, SWITZERLAND
S. Cucciarelli, M. Konecki

CERN, European Organization for Nuclear Research, Geneva, SWITZERLAND
S. Ashby, D. Barney, P. Bartalini, R. Benetta, V. Brigljevic, G. Bruno, E. Cano, S. Cittolin, A. Csilling, A. De Roeck, P. Favre, A. Frey, W. Funk, D. Futyan, D. Gigi, F. Glege, J. Gutleber, M. Hansen, V. Innocente, C. Jacobs, W. Jank, M. Kozlovszky, H. Larsen, M. Lenzi, I. Magrans, M. Mannelli, F. Meijers, E. Meschi, L. Mirabito[5], S.J. Murray, A. Oh, L. Orsini, C. Palomares Espiga, L. Pollet, A. Racz, S. Reynaud, D. Samyn, P. Scharff-Hansen, C. Schwick, G. Sguazzoni, N. Sinanis, P. Sphicas[6], A. Strandlie, B.G. Taylor, I. Van Vulpen,

J.P. Wellisch, M. Winkler

Paul Scherrer Institut, Villigen, SWITZERLAND
D. Kotlinski

Universität Zürich, Zürich, SWITZERLAND
K. Prokofiev, T. Speer

Cukurova University, Adana, TURKEY
I. Dumanoglu

University of Bristol, Bristol, UNITED KINGDOM
D.S. Bailey, J.J. Brooke, D. Cussans, G.P. Heath, D. Machin, S.J. Nash, D.M. Newbold, M.G. Probert

Rutherford Appleton Laboratory, Didcot, UNITED KINGDOM
J.A. Coughlan, R. Halsall, W.J. Haynes, I.R. Tomalin

Imperial College, University of London, London, UNITED KINGDOM
N. Marinelli, A. Nikitenko[3], S. Rutherford, C. Seez[1]

Brunel University, Uxbridge, UNITED KINGDOM
O. Sharif

University of California, Davis, Davis, California, USA
R. Breedon, P.T. Cox, P. Murray, M. Tripathi

University of California, San Diego, La Jolla, California, USA
S. Bhattacharya, J.G. Branson, I. Fisk, J. Letts, M. Mojaver, H.P. Paar, E. Trepagnier

University of California, Los Angeles, Los Angeles, California, USA
R. Cousins, S. Erhan, J. Hauser, P. Kreuzer, M. Lindgren, J. Mumford, P. Schlein, Y. Shi, B. Tannenbaum, V. Valuev, M. Von Der Mey

California Institute of Technology, Pasadena, California, USA
V. Litvine, S. Shevchenko, R. Wilkinson

University of California, Riverside, Riverside, California, USA
I. Andreeva[1], R. Clare, S. Villa

University of Florida, Gainesville, Florida, USA
D. Acosta, D. Bourilkov[7], A. Korytov, A. Madorsky, G. Mitselmakher, J.L. Rodriguez, B. Scurlock

Fermi National Accelerator Laboratory, Batavia, Illinois, USA
S. Aziz, M. Bowden, J.E. Elias, G. Graham, D. Green, M. Litmaath, V. O'Dell, N. Ratnikova, I. Suzuki, H. Wenzel

University of Maryland, College Park, Maryland, USA
S. Abdullin[3], D. Baden, S.C. Eno, T. Grassi, S. Kunori

Boston University, Boston, Massachusetts, USA
G. Antchev[7], E. Hazen, J. Rohlf, S. Wu

Northeastern University, Boston, Massachusetts, USA
I. Osborne, L. Taylor, L. Tuura

Massachusetts Institute of Technology, Cambridge, Massachusetts, USA
S. Pavlon, K. Sumorok, S. Tether

University of Mississippi, University, Mississippi, USA
L.M. Cremaldi, D. Sanders, D. Summers

Princeton University, Princeton, New Jersey, USA
W.C. Fisher, J. Mans, D. Stickland, C. Tully, T. Wildish, S. Wynhoff

Rice University, Houston, Texas, USA
B.P. Padley

University of Wisconsin, Madison, Wisconsin, USA
P. Chumney, S. Dasu, F.R. Di Lodovico, M. Jaworski, J. Lackey, W.H. Smith

1: Also at CERN, European Organization for Nuclear Research, Geneva, Switzerland
2: Now at Department of Physical Sciences, University of Helsinki, Finland
3: Also at Institute for Theoretical and Experimental Physics, Moscow, Russia
4: Also at Università di Pisa, Scuola Normale Superiore e Sezione dell' INFN, Pisa, Italy
5: Also at Institut de Physique Nucléaire de Lyon, IN2P3-CNRS, Univ. Lyon I, Villeurbanne, France
6: Also at MIT, Cambridge, USA and University of Athens, Greece
7: Also at Institute for Nuclear Research and Nuclear Energy, Sofia, Bulgaria
8: Also at Institute of Electronic Systems, Technical University of Warsaw, Poland