CERN/LHCC 2001-40
LHCb TDR 7
19 December 2001

TDR

LHCb Online System

Data Acquisition

&

Experiment Control

Technical Design Report

The LHCb Collaboration

Brasilian Center for Particles Physics, CBPF, Rio de Janeiro, Brasil
P.R.Barbosa Marinho, I.Bediaga, A.F.Barbosa, G.Cernicchiaro, E.Correa de Oliveira, J.Magnin, J.Marques de Miranda, A.Massafferri, A.Reis, R.Silva

University of Rio de Janeiro, UFRJ, Rio de Janeiro, Brasil
S.Amato, P.Colrain, T.da Silva, J.R.T.de Mello Neto, L.de Paula, M.Gandelman, J.H.Lopes, B.Marechal, D.Moraes(1), E.Polycarpo, F.Vinci do Santos
(1) also at CERN

University of Clermont-Ferrand II, Clermont-Ferrand, France
Z.Ajaltouni, G.Bohner, V.Breton, R.Cornat, O.Deschamps, P.Henrard, J.Lecoq, P.Perret, C.Rimbault, C.Trouilleau[1]

CPPM Marseille, Aix University-Marseille II, Marseille, France
E.Aslanides, J.P.Cachemiche, P.Y.Duval, R.Le Gac, O.Leroy, P.L.Liotard, M.Menouni, R.Potheau, A.Tsaregorodtsev, B.Viaud

Laboratoire d'Annecy-le-Vieux de Physique des Particules (LAPP), France
D.Boget, I.de Bonis, D.Decamp, N.Dumont Dayot, N.Fouque, M.Gougerot, M.-N.Minard, B.Pietrzyk

University of Paris-Sud, LAL Orsay, Orsay, France
G.Barrand, C.Beigbeder-Beau, D.Breton, O.Callot, Ph.Cros, B.D'Almagne, B.Delcourt, F.Fulda Quenzer, A.Jacholkowska[2], B.Jean-Marie, J.Lefrançois, F.Machefert, V.Tocut, K.Truong

Technical University of Dresden, Dresden, Germany
R.Schwierz, B.Spaan

Max-Planck-Institute for Nuclear Physics, Heidelberg, Germany
C.Bauer, D.Baumeister, N.Bulian, H.P.Fuchs, T.Glebe, W.Hofmann, K.T.Knöpfle, S.Löchner, A.Ludwig, F.Sanchez Nieto, M.Schmelling, B.Schwingenheuer

Physics Institute, University of Heidelberg, Heidelberg, Germany
S.Bachmann, P.Bock, H.Deppe, F.Eisele, M.Feuerstack-Raible, S.Henneberger, P.Igo-Kemenes, R.Rusnyak, U.Stange, U.Trunk, M.Walter, D.Wiedner, U.Uwer

Kirchhoff Institute for Physics, University of Heidelberg, Heidelberg, Germany
V.Lindenstruth, R.Richter, M.W.Schulz, A.Walsch

Laboratori Nazionali dell' INFN, Frascati, Italy
G.Bencivenni, C.Bloise, F.Bossi, P.Campana, G.Capon, P.DeSimone, C.Forti,M.A.Franceschi, F.Murtas, L.Passalacqua, V.Patera(1), A.Sciubba(1)
(1)also at Dipartimento di Energetica, University of Rome, ``La Sapienza''

University of Bologna and INFN, Bologna, Italy
M.Bargiotti, A.Bertin, M.Bruschi, M.Capponi, I.D'Antone, S.de Castro,P.Faccioli, L.Fabbri, D.Galli, B.Giacobbe, I.Lax, U.Marconi, I.Massa, M.Piccinini,M.Poli, N.Semprini-Cesari, R.Spighi, V.Vagnoni, S.Vecchi, M.Villa, A.Vitale, A.Zoccoli

University of Cagliari and INFN, Cagliari, Italy
S.Cadeddu, A.Cardini, M.Caria, A.Lai, D.Pinci, B.Saitta(1)
(1) also at CERN

University of Ferrara and INFN, Ferrara, Italy
V.Carassiti, A.Cotta Ramusino, P.Dalpiaz, A.Gianoli, M.Martini, F.Petrucci, M.Savrié

University of Florence and INFN, Florence, Italy
A.Bizzeti, M.Calvetti, G.Collazuol, E.Iacopini, M.Lenti, F.Martelli, G.Passaleva, M.Veltri

University of Genoa and INFN, Genoa, Italy
S.Cuneo, F.Fontanelli, V.Gracco, P.Musico, A.Petrolini, M.Sannino

University of Milano-Bicocca and INFN, Milano, Italy
M.Alemi, T.Bellunato(1), M.Calvi, C.Matteuzzi, M.Musy, P.Negri, M.Paganoni
(1) also at CERN

University of Rome, "La Sapienza" and INFN, Rome, Italy
G.Auriemma(1), V.Bocci, C.Bosio, D.Fidanza(1), A.Frenkel, K.Harrison, G.Martellotti, S.Martinez, G.Penso, S.Petrarca, G.Pirozzi, R.Santacesaria, C.Satriano(1), A.Satta
(1) also at University of Basilicata, Potenza

University of Rome, "Tor Vergata" and INFN, Rome, Italy
G.Carboni(1), D.Domenici, G.Ganis, R.Messi, L.Pacciani, L.Paoluzi, E.Santovetti
(1) also at CERN

NIKHEF, The Netherlands
G.van Apeldoorn(1,3), N.van Bakel(1,2), T.S.Bauer(1,4), J.F.J van den Brand(1,2), H.J.Bulten(1,2), C.Carloganu (1), M.Doets(1), R.van der Eijk(1), I.Gouz(1,5), D.Groep(1), V.Gromov(1), R.Hierck(1), L.Hommels(1), E.Jans(1), T.Ketel(1,2), S.Klous (1,2), B.Koene(1), M.Merk(1), F.Mul(2), M.Needham(1), H.Schuijlenburg(1), T.Sluijk(1), J.van Tilburg(1), H.de Vries(1), L.Wiggers(1), E.Zupan(1)
(1) Foundation of Fundamental Research of Matter in the Netherlands
(2) Free University Amsterdam
(3) University of Amsterdam
(4) University of Utrecht

Institute of High Energy Physics, Beijing, P.R.C.
C.Gao, C.Jiang, H.Sun, Z.Zhu

Research Centre of High Energy Physics, Tsinghua University, Beijing, P.R.C.
M.Bisset, J.P.Cheng, Y.G.Cui, Y.Dai, Y.Gao, H.J.He, C.Huang, Y.P.Kuang, Q.Li, Y.J.Li, Y.Liao, J.P.Ni, B.B.Shao,J.J.Su, Y.R.Tian, Q.Wang, Q.S.Yan

Institute for Nuclear Physics and University of Mining and Metalurgy, Krakow, Poland
E.Banas, J.Blocki, K.Galuszka, L.Hajduk, P.Jalocha, P.Kapusta, B.Kisielewski, W.Kucewicz, T.Lesiak, J.Michalowski, B.Muryn, Z.Natkaniec, W.Ostrowicz, G.Polok, E.Rulikowska-Zarebska, M.Stodulski, T.Szumlak, M.Witek(1), P.Zychowski
(1) also at CERN

Soltan Institute for Nuclear Physics, Warsaw, Poland
M.Adamus, A.Chlopik, Z.Guzik, A.Nawrot, M.Szczekowski

Horia Hulubei-National Institute for Physics and Nuclear Engineering(IFIN-HH), Bucharest-Magurele, Romania
D.V.Anghel[3], C.Coca, A.Cimpean, G.Giolu, C.Magureanu, S.Popescu(1), T.Preda, A.M.Rosca(2), V.L.Rusu[4]
(1) also at CERN
(2) also at Humbolt University, Berlin

Institute for Nuclear Research (INR), Moscow, Russia
V.Bolotov, S.Filippov, J.Gavrilov, E.Guschin, V.Kloubov, L.Kravchuk, S.Laptev, V.Laptev, V.Postoev, A.Sadovski, I.Semeniouk

Institute of Theoretical and Experimental Physics (ITEP), Moscow, Russia
S.Barsuk, I.Belyaev (1), A.Golutvin, O.Gouchtchine, V.Kiritchenko, V.Kochetkov, I.Korolko(1), G.Pakhlova, N.Levitski, A.Morozov, P.Pakhlov, D.Roussinov, V.Rusinov, S.Semenov, A.Soldatov, E.Tarkovski
(1) also at CERN

Budker Institute for Nuclear Physics (INP), Novosibirsk, Russia
K.Beloborodov, A.Bondar, A.Bozhenok, A.Buzulutskov, S.Eidelman, V.Golubev, P.Krokovnyi, S.Oreshkin, A.Poluektov, S.Serednyakov, L.Shekhtman, B.Shwartz, Z.Silagadze, A.Sokolov, A.Vasiljev

Institute for High Energy Physics (IHEP-Serpukhov),Protvino, Russia
L.A.Afanassieva, I.V.Ajinenko, K.Beloous, V.Brekhovskikh, S.Denissov, A.V.Dorokhov, R.I.Dzhelyadin, A.Kobelev, A.K.Konoplyannikov, A.K.Likhoded, V.D.Matveev, V.Novikov, V.F.Obraztsov, A.P.Ostankov, V.I.Rykalin, V.K.Semenov, M.M.Shapkin, N.Smirnov, A.Sokolov, M.M.Soldatov, V.V.Talanov, O.P.Yushchenko

Petersburg Nuclear Physics Institute, Gatchina, St.Petersburg, Russia
B.Botchine, S.Guetz, A.Kashchuk(1), V.Lazarev, N.Saguidova, V.Souvorov(1), E.Spiridenkov, A.Vorobyov, An.Vorobyov
(1) also at CERN

University of Barcelona, Barcelona, Spain
E.Aguilo, R.Ballabriga(1), S.Ferragut, Ll.Garrido, D.Gascon, R.Graciani Diaz, S.Luengo(1), R.Miquel[5], D.Peralta, M.Rosello(1), X.Vilasis(1)
(1) also at departament d'Engineria Electronica La Salle, Universitat Ramon Llull, Barcelona

University of Santiago de Compostela, Santiago de Compostela, Spain
B.Adeva, P.Conde, F.Gomez, J.A.Hernando, A.Iglesias, A.Lopez-Aguera, A.Pazos, M.Plo, J.M.Rodriguez, J.J.Saborido, M.J.Tobar

University of Lausanne, Lausanne, Switzerland
P.Bartalini, A.Bay, B.Carron, C.Currat, O.Dormond, F.Dürrenmatt, Y.Ermoline, R.Frei, G.Gagliardi, G.Haefeli, J.P.Hertig, P.Koppenburg, T.Nakada(1), J.P.Perroud, F.Ronga, O.Schneider, L.Studer, M.Tareb, M.T.Tran
(1) also at CERN, on leave from PSI, Villigen

University of Zürich , Zürich, Switzerland
R.Bernet, E.Holzschuh[6], F.Lehner, P.Sievers, O.Steinkamp, U.Straumann, A.Vollhardt, D.Wyler, M.Ziegler

Institute of Physics and Technologies, Kharkiv, Ukraine
A.Dovbnya, S.Maznichenko, O.Omelaenko, Yu.Ranyuk, V.Shulayev

Institute for Nuclear Research, Kiev, Ukraine
V.Aushev, V.Kiva, I.Kolomiets, Yu.Pavlenko, V.Pugatch, Yu.Vasiliev

University of Bristol, Bristol, U.K.
N.H.Brook, J.E.Cole, R.D.Head, A.Phillips, A.Presland, F.F.Wilson

University of Cambridge, Cambridge, U.K.
K.George, V.Gibson, C.R.Jones, S.G.Katvars, C.Shepherd-Themistocleous, C.P.Ward, S.A.Wotton

Rutherford Appleton Laboratory, Chilton, U.K.
C.A.J.Brew[7], C.J.Densham, S.Easo, B.Franek, J.G.V.Guy, R.N.J.Halsall, J.A.Lidbury, J.V.Morris, A.Papanestis, G.N.Patrick, F.J.P.Soler, S.A.Temple, M.L.Woodward

University of Edinburgh, Edinburgh, U.K.
A.Barczyk, S.Eisenhardt, A.Khan, F.Muheim, S.Playfer, A.Walker

University of Glasgow, Glasgow, U.K.
A.J.Flavell, A.Halley, V.O'Shea, A.Pickford, F.J.P.Soler

University of Liverpool, Liverpool, U.K.
S.Biagi, T.Bowcock, R.Gamet, M.McCubbin, J.Palacios, C.Parkes, G.Patel, V.Wright

Imperial College, London, U.K.
G.J.Barber, D.Clark, P.Dauncey, A.Duane, U.Egede, M.Girone(1), J.Hassard, R.Hill, M.J.John[8], S.Jolly, D.R.Price, P.Savage, L.Toudup, D.Websdale
(1) also at CERN

University of Oxford, Oxford, U.K
M.Adinolfi, G.Damerell, J.H.Bibby, M.J.Charles, N.Harnew, F.Harris, I.A.McArthur, J.Rademacker, N.J.Smale, S.Topp-Jorgensen, G.Wilkinson

CERN, Geneva, Switzerland
G.Anelli, F.Anghinolfi, F.Bal, M.Benayoun(1), R.Beneyton[9], W.Bonivento(2), A.Braem, J.Buytaert, M.Campbell, A.Cass, M.Cattaneo, Ph.Charpentier, E.Chesi, J.Christiansen, J.Closier, P.Collins, G.Corti, C.D'Ambrosio, H.Dijkstra, J.P.Dufey, M.Ferro-Luzzi, F.Fiedler, W.Flegel, F.Formenti, R.Forty, M.Frank, C.Frei, I.Garcia Alfonso, C.Gaspar, P.Gavillet, G.Gracia Abril[10], A.Guirao Elias, T.Gys, F.Hahn, S.Haider, J.Harvey, B.Hay[11], E.van Herwijnen, H.J.Hilke, G.von Holtey, D.Hutchroft, R.Jacobsson, P.Jarron, C.Joram, B.Jost, D.Lacarrère, M.Laub[12], M.Letheren, J.F.Libby, C.Lippmann, R.Lindner, M.Losasso, P.Mato Vila, H.Müller, N.Neufeld,

K.Osterberg, C.Padilla, U.Parzefall, S.Ponce, F.Ranjard, W.Riegler, F.Rohner, S.Roiser, T.Ruf, S.Schmeling, B.Schmidt, T.Schneider, A.Schopper, W.Snoeys, W.Tejessy, F.Teubert, J.Toledo Alarcon[13], O.Ullaland, A.Valassi, P.Vazquez Regueiro, I.Videau, P.Wertelaers, A.Wright[14], K.Wyllie
(1) on leave from Université de Paris VI et VII (LPNHE), Paris
(2) on leave from INFN Cagliari, Cagliari

[1]now at Thales Microelectonics, Grenoble, France
[2]now at Groupe d'Astroparticules de Montpellier (GAM), Montpellier, France
[3]now at Oslo University, Oslo, Norway
[4]now at Pennsylvania University, Philadelphia, USA
[5]now at LBNL, Berkeley, USA
[6] Deceased in 2001
[7]now at Fermilab, Chicago, USA
[8]now at Collège de France, Paris, France
[9] now at Université de Paris Sud, LAL-Orsay, Paris
[10]now at ENWARE, Madrid, Spain
[11]now at SWX Swiss Exchange, Geneva, Switzerland
[12]now at Technical University of Prague, Prague, Czech Republic
[13]now at Polytechnical University of Valencia, Valencia, Spain
[14]now at Lancaster University, Lancaster, UK

Acknowledgements

We would like to thank Manuel Mazo and Konrad Paszkiewicz for the work they have done during their stay as CERN summer students in 2001.

We would also like to acknowledge the CERN's CMS DAQ group for giving us access to their switching test-bed for studies in the context of the performance measurements of the Foundry switch.

Table of Contents

Chapter 1 Introduction

1.1 Overview

The LHCb detector is designed to exploit the large number of b-hadrons produced at the LHC in order to make precision studies of CP asymmetries and of rare decays in the B-meson systems. LHCb is a single-arm spectrometer with a forward angular coverage from 10 mrad to 300 mrad in the horizontal projection and to 250 mrad in the vertical projection [1]. The layout of the spectrometer is shown in Figure 1. The detector can reconstruct a B-decay vertex with very good resolution and provides excellent particle identification for charged particles. It has a high performance trigger, which is optimised to select events with B-mesons efficiently, based on particles with large transverse momentum and displaced secondary vertices.

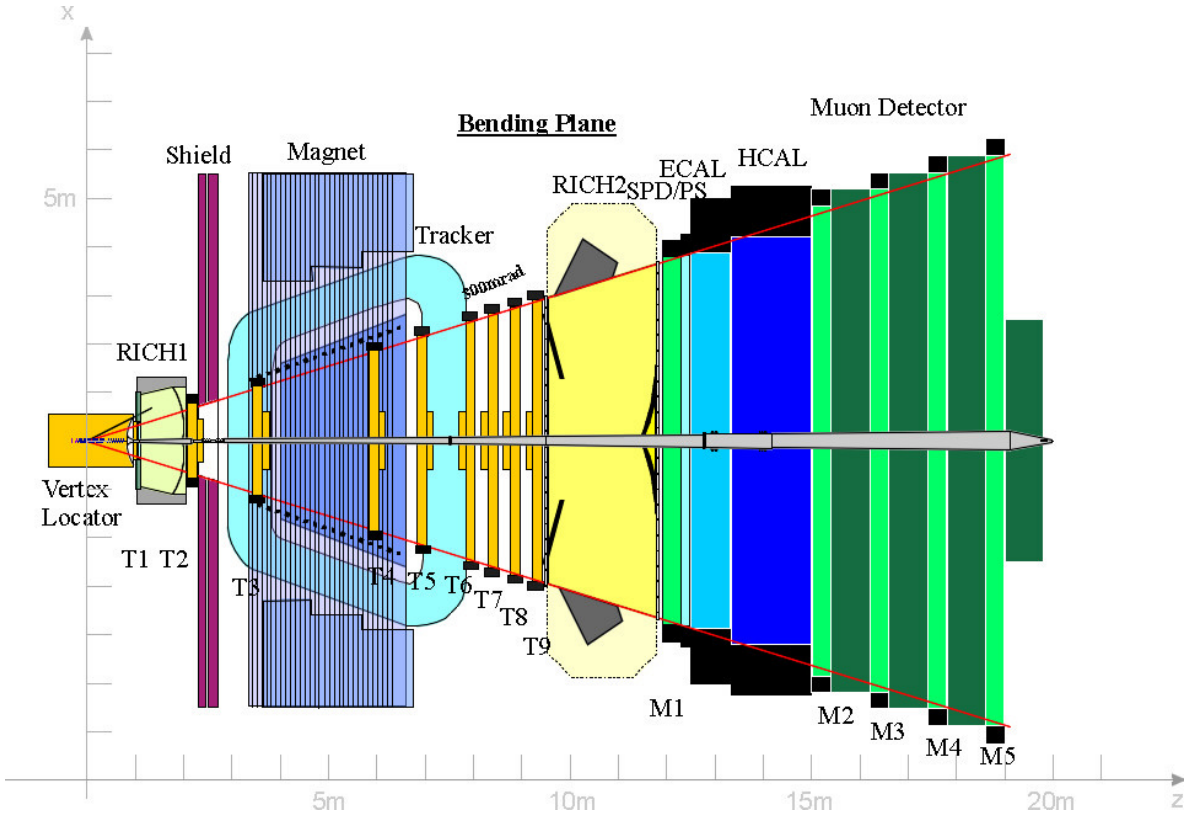

Figure 1 Schematic drawing of the LHCb detector as seen from above.

LHCb comprises a number of different sub-detectors:

- The Vertex Locator (VELO) features a series of silicon stations placed along the beam direction and is used to provide precise measurements of track coordinates close to the interaction region. These are used to reconstruct production and decay vertices of beauty and charm hadrons, to provide an accurate measurement of their lifetimes, and to measure the impact parameter of particles used to tag their flavour.

  Two of the silicon stations in the backward region are designed and read out for the purpose of discriminating single interactions from bunch crossings with more than one interaction. This acts as a pile-up veto and is used as an ingredient of the Level-0 trigger.

- Charged particle identification is achieved through two Ring Image Cherenkov Counters, an upstream detector (RICH1) containing aerogel and C_4F_{10} radiators and a downstream detector (RICH2) having a CF_4 radiator. Three radiators are used in order to cover the full momentum range.
- A spectrometer dipole magnet, which is placed close to the interaction region in order to minimise its size.
- The tracking system consists of a series of stations with Inner (IT) and Outer (OT) components for finding particle tracks in the region between the vertex detector and the calorimeters and for measuring particle momenta.
- The calorimeter system comprises a scintillator pad detector (SPD), a preshower detector (PS) and an electromagnetic calorimeter (ECAL) followed by a hadron calorimeter (HCAL). Together they provide high transverse energy hadron, electron and photon candidates for the Level-0 trigger, identification of electrons for flavour tagging and good reconstruction of π^0s and photons for the study of B-meson decays.
- The muon detector (MUON) uses the penetrative power of muons to provide a robust muon trigger and to identify muons for flavour-tagging and reconstruction of muonic final states of B-meson decays.

LHCb plans to operate with an average luminosity of 2 x 10^{32} cm^{-2} s^{-1}, which should be obtained from the beginning of LHC operation. About 10^{12} $b\bar{b}$ pairs are expected to be produced in each year of running which corresponds to a production rate of ~100 kHz. However, events with fully reconstructed interesting $b\bar{b}$ final states represent only a small fraction of the total $b\bar{b}$ sample due to the small branching ratios and limited detector acceptance. The LHCb trigger system will select the small fraction of interesting events from the large number of $b\bar{b}$ and other pp inelastic events.

The first two levels of trigger will be applied whilst data from the detectors are still buffered in the front-end electronics. The role of the data acquisition system (DAQ) is to collect zero suppressed data corresponding to triggered events and to assemble them into complete events. It must also filter and fully reconstruct interesting events, using high level trigger algorithms running in a powerful CPU farm, and dispatch them to permanent storage. The Experiment Control System (ECS) will be used to configure the readout system, to control and monitor the state of the detector components, to steer the actual data taking and to provide checks of the quality of the data recorded for physics analysis.

In the following, we first give an overview on the architecture of the trigger and data acquisition systems, followed by a discussion of the design and implementation goals. We then describe how the design and implementation of each component is covered in the body of this Technical Design Report (TDR).

1.1.1. Trigger / DAQ Overview

Data flows through the various stages of the DAQ system under the control of a four level trigger system (Figure 2). Level-0 is a high p_T trigger operating at the bunch crossing frequency of 40 MHz, and is designed to achieve a total suppression factor of ~15[1]. It has a fixed latency of 4 μs and is distributed to the front-end pipelines in a time-synchronised manner. Level-1 uses the VELO to select events containing one or more secondary vertices. It operates at the Level-0 accept rate, nominally 1 MHz, and has a suppression factor of 25. The Level-1 trigger is also distributed to the front-end electronics. The transfer of data from the front-end electronics to the DAQ system is

[1] While the bunch crossing rate is 40 MHz, the interaction rate in LHCb, at the foreseen luminosity and due to the fact that there are empty bunches, is only 15 MHz.

initiated by a positive Level-1 decision, which runs at 40 kHz. The average event length of these zero-suppressed data is 100 kB and thus the task of the DAQ is to assemble complete events at a total rate of 4 GB/s. The high level triggers comprise sophisticated software algorithms working on complete events. They implement a number of selection criteria that are successively applied reducing the overall rate of accepted events to a nominal 200 Hz. For these events the reconstruction, which has to a large extent already been performed in the course of the triggering process, is finished and the output, Event Summary Data (ESD), is written to permanent storage together with the data collected from the detectors (RAW). The LHCb Trigger and the offline computing system will be described in detail in the Trigger and Computing TDRs.

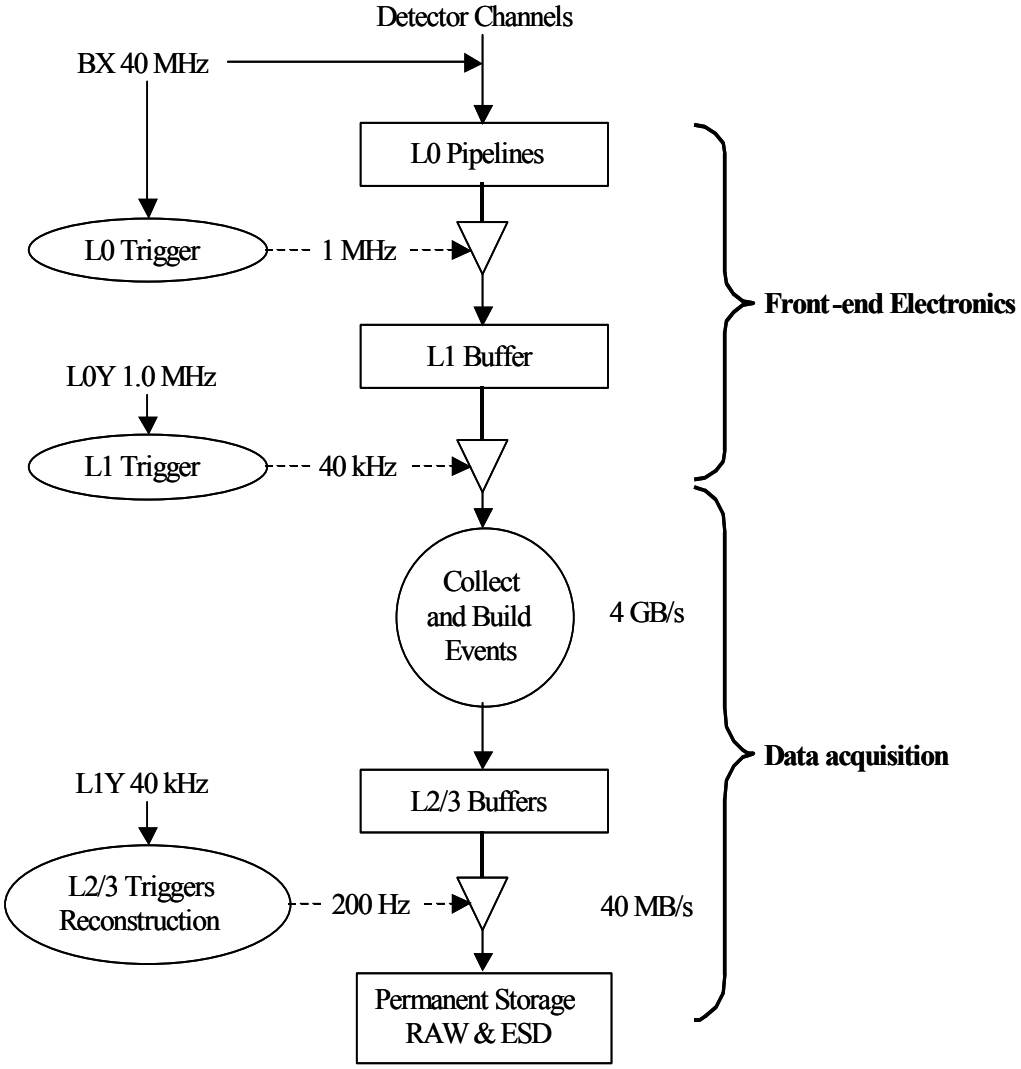

Figure 2 Schematic diagram of the LHCb trigger and data acquisition system

1.1.2. Design and Implementation Goals

Our approach to the design and implementation of the LHCb DAQ and ECS systems has been strongly influenced by our experience building and operating the ALEPH and DELPHI systems at LEP [2], [3]. We have found that the ability to maintain very high running efficiencies, to adapt the system to changing needs and to operate under special running modes is strongly influenced by the way the system is originally conceived and specified. A cohesive online team working closely together and with strong links to sub-detector groups is also considered to be an important ingredient for the success of the project.

For LHCb, special attention has been placed on specification of the system architecture, of the dataflow protocols and of the main functional elements. These specifications are independent of the particular implementation choice, thus permitting upgrades to be made later on that can take advantage of new technologies without requiring change to the underlying architecture. Bearing in mind the extreme data rates (4 GB/s) and the large number of readout elements and links needed to realise the system, a guiding principle has been to keep the design as simple as possible in order to ensure safe and reliable operation. For example, event fragments are routed through the Readout Network to form full events in the destination CPUs without the need of a central flow control unit. This considerably simplifies the dataflow protocol and minimises the number of different functional units that have to be designed and built. In addition, complete events are immediately made available, permitting full flexibility in defining and applying the high-level trigger algorithms. The architecture has also been developed with careful attention to its scalability in order to be able to cope with larger data rates and processing power should this be required in the future.

It has also been a requirement that each sub-detector group should be given some autonomy in the operation of the readout of their hardware and therefore a key feature of the readout architecture is the concept of 'partitioning'. Partitioning is needed in order to support parallel and independent data-taking activities, which will be needed during the commissioning of the system and especially for making calibrations and tests outside of normal data-taking periods. As the name suggests, pieces of the readout system can be partitioned logically so as to create autonomous data acquisition sub-systems. A partition is therefore defined as being any section of the readout system that can be configured to function independently of the rest of the system. Each partition consists of a 'pipeline' in which data flows from the front-end electronics to a subset of the CPUs executing the final software algorithms. Examples of these are the trigger, reconstruction or calibration tasks, depending on the activity in progress. More than one partition may exist at any one time thus permitting parallel data streams. Support of partitioning has particularly important consequences for the design of the system used to distribute the timing and trigger decisions to the front-end electronics.

Wherever possible we have chosen to standardise on common components, to minimise the effort needed to develop the system and to ease maintenance in the long term. Thus there are no LHCb sub-detector-specific implementations of standard readout elements. In addition we also make use of common LHC developments, such as the Trigger Timing and Control (TTC) distribution system [4] and are following more recent efforts in CERN/EP division to specify crates and rack control systems. We also intend to participate in projects organised in CERN/IT division concerning the management of large scale computing fabrics for deployment in CPU farms. Finally, we use the latest state-of-the-art commercial components for their programmability, and hence the extra flexibility they afford us.

A dedicated local area network will be used to provide a communication and control path between the main online computers and each component of the readout system. This control path is used to configure, control and monitor the various elements of the system and is physically completely separate from the path used to collect the data. Provision of a secure and independent communication path to each hardware module is considered to be essential for detecting and recovering from errors in the readout system. This path will also be used by the ECS to acquire slowly changing data from the detector to keep a permanent record of environmental parameters (temperatures, gas pressures etc.).

A common control software framework is under development in the context of the Joint Controls Project [5], in which LHCb team members are actively participating. This framework also makes use of a commercial software package [6], which has considerably eased the development of the large set of control and monitoring applications constituting the online system. The existing software infrastructure provides a very good paradigm for communication between developers from

all sub-detector groups, as well as between developers on different experiments. It should permit the development of a coherent interface for the shift crew, which should simplify the task of running the system, identifying problems and recovering from them.

1.2 Structure of the Document

This Technical Design Report is designed to be a concise but self-contained description of the LHCb data acquisition and experiment control systems. Further technical details can be found in a number of supporting technical notes that are referenced in the body of the text. In Chapter 2, we describe the environment in which the system operates and the requirements placed on it by the physics programme, the LHC collider and the LHCb detector itself. Chapter 3 outlines the design of the architecture and identifies the main functional elements. Chapter 4 describes the detailed implementation of the system, mentioning technology choices. The TDR concludes in Chapter 5 with a discussion of the cost, planning and assignment of responsibilities. Important results from R&D studies relating to development of DAQ and ECS components are described in some detail in three appendices.

Chapter 2 Requirements

The environment in which the online system must operate is represented in the form of a Context Diagram in Figure 3. This shows all the external components with which the online system must interact and which form a useful basis for classifying all the requirements placed on the system. In making the design for the realisation of the DAQ/ECS system, careful consideration has been given to satisfying all these requirements, as well as satisfying the need to operate within the technological and financial constraints and to choose an approach that will match the expertise of the LHCb online team.

It is evident that the DAQ/ECS system must provide facilities for collecting data from the detector and for configuring and supervising the operation of the detector. A fundamental goal is to satisfy the needs of the physics programme. In addition, there is a wide spectrum of requirements that come from the need to communicate with other systems. For example, the data bandwidths that must be sustained by the DAQ are largely determined by the performance of the trigger system. Many requirements on the system come from operational issues, such as the need to support various running modes (tests, calibrations, normal running) and partitioning. As is evident in a colliding beam machine, the operation of the experiments and the LHC accelerator are tightly coupled and communication of control and status information between the two is required at hardware and software levels. The environment in the experimental area must be monitored to ensure the correct operation of basic services, such as ventilation and cooling, and to satisfy safety criteria. All these areas provide input to the functionality and performance required of the system and are discussed in more detail in the following sections.

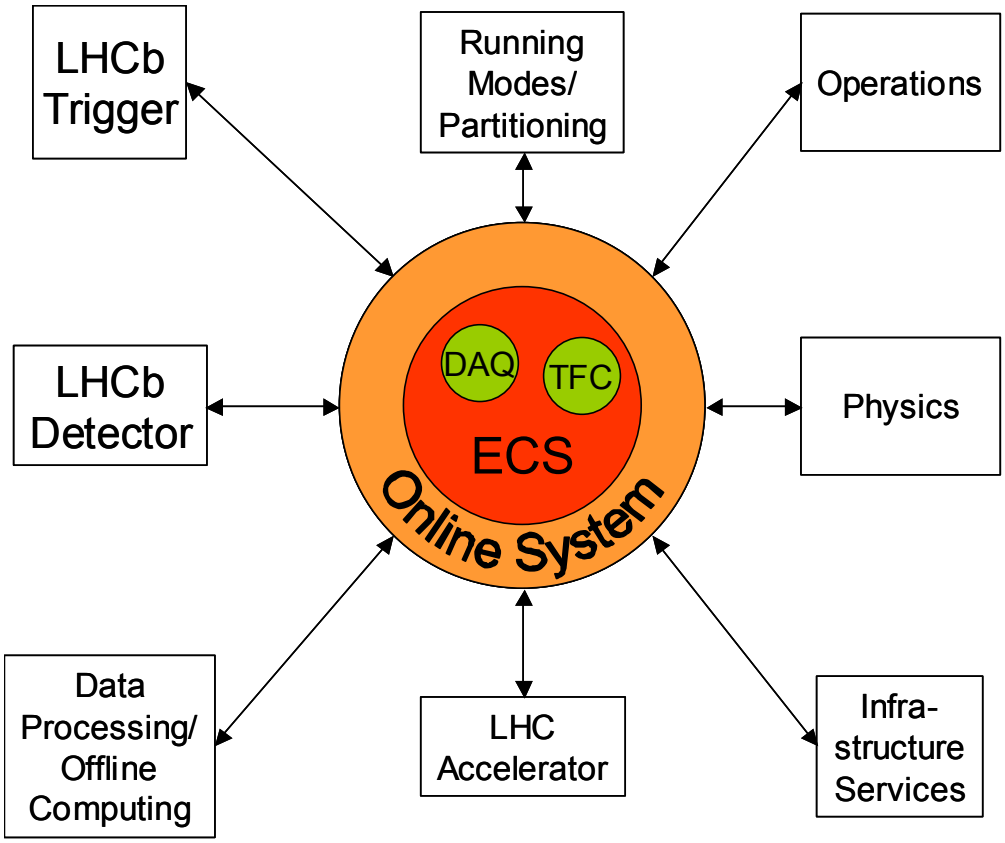

Figure 3 Context diagram showing components in which the LHCb online system will run.

2.1 Physics Requirements

LHCb is an experiment dedicated to the study of CP violation in hadronic systems originating from b-quarks. These effects can be observed in asymmetries in the distribution of some observables when comparing specific final states of B and $\overline{B}$ - mesons. The effective fraction of interesting events is very small (of the order of 10^{-5} or less). Hence, despite the fact that with an LHC luminosity of $2 \cdot 10^{32}$ cm^{-2} s^{-1} more than 100'000 B mesons are produced each second, only a very small fraction of these are selected and stored for further analysis offline. It is therefore of the utmost importance to design and implement as efficient a trigger as possible. The counterpart is that this can only be achieved at the price of a relatively high level of background originating mainly from uninteresting B decays, inducing a high demand on the capabilities of the readout system. Clearly, reliability and efficiency are expected from the online system in order to record as many interesting events as possible.

The data acquisition system must ensure the error-free[1] transmission of the data from the front-end electronics to the storage device. This transfer of data should not introduce dead-time, if the system is operated within the design parameters. The online system will also be responsible for setting-up and monitoring the equipment involved in the data acquisition, in particular the sub-detector front-end electronics.

2.2 LHCb Detector

Ten sub-detectors and two levels of triggering produce data that must be collected by the DAQ for each triggered event. The channel count corresponding to these components varies between a few hundred (trigger systems) and several hundred thousands (the tracker system), totalling approximately 1.1 million for the complete detector (Table 1).

Table 1 LHCb DETECTOR CHANNELS, OCCUPANCIES AND AVERAGE EVENT SIZES

	VELO	**RICH1**	**RICH2**	**IT**	**OT**	**SPD**	**PS**	**ECAL**	**HCAL**	**MUON**	**TRIG**
Number of Channels [k]	205	172	278	340	120	6	6	6	1.5	26	1
Average Occupancy [%]	0.7	1	1	1	10	7	7	7	13	1.5	-
Average Event Size [kB]	5	6	10	10	33	3	3	4	2	2	1

The average total event size has been estimated from average detector occupancies, determined through simulation studies and sums to ~80 kB. An average event size of 100 kB has therefore been assumed for the purposes of calculating the bandwidth that must be sustained by the DAQ. An additional requirement is that the readout system must be able to accept very large events (several MB) carrying calibration data.

The sub-detectors will be located in the US85 cavern of the LHC accelerator and will be inaccessible during data taking. This imposes stringent constraints on the components of the control system that are located at or near the detector and are therefore exposed to radiation. To guarantee continuous control over the electronics in the cavern, the interfaces to the control system have to be immune to radiation effects, especially Single Event Upsets (SEUs). If SEUs occur, they must be

[1] Of course, there is never a data transfer system that operates error free. If errors occur, however, they should be detected and the corresponding data should be flagged as error prone. An acceptable level of error-prone events could be a (unbiased) fraction of 10^{-6}.

detected and the control software should be able to recover from them in a transparent way. The rate of errors due to SEUs in the controls interface hardware should be lower than about one every 20000 seconds (i.e. ~once per fill), since these errors require data-taking to be stopped during error recovery and therefore introduce dead-time.

2.3 LHCb Trigger System

The nature and topology of the events containing B-mesons are such that it is extremely difficult to completely distinguish these events from background events generated by other physics processes, e.g. to discriminate an event containing b-quarks from an event containing other quarks. Independent sets of trigger algorithms largely based on sophisticated pattern recognition code and working on complete event data are required to select each event topology of interest. These algorithms, which constitute the so called high-level triggers, must therefore run on powerful general purpose processors after the event building stage. Two levels of triggering will therefore be applied at the front-end electronics in order to pre-select an enriched sample of interesting events, such that the data acquisition system can be realised with a reasonable amount of resources. Thus two trigger decisions must be distributed to the front-end electronics and temporary buffering of the data is needed after each stage for the latency of the triggers. This has implications for the system used to distribute the LHC clock and trigger signals to the front-end electronics (the Timing and Fast Controls (TFC) system), and also on the architecture of the front-end electronics itself. The latter is described generically in [7] and the specific implementations are outlined in the TDRs of the individual sub-detectors.

The characteristics of these first two trigger levels are described in Table 2 in terms of input/output rates and the detectors whose data are used to reach a decision.

Table 2 CHARACTERISTICS OF THE FIRST TWO LHCb TRIGGER LEVELS

	Level-0	**Level-1**
Input Rate	40 MHz	1.0 MHz (Level-0 accept)
Average Accept Rate	1.0 MHz	40 kHz
Detector Data used	Calorimeter, Muon, Pile-Up Veto	VELO, Level-0
Latency	4.0 μs (fixed)	<2 ms (variable)

The CPU power required to execute the high level algorithms has been estimated from performance measurements made of the pattern recognition and track fitting software used for the reconstruction of simulated events. In the Level-2 trigger, which is designed to match vertex information from the VELO with momentum information provided by the tracking system, most of the CPU requirement comes from the momentum measurement. The Level-2 algorithm is estimated to require 1 SI95·s per event. For the Level-3 trigger, which uses refined and optimised reconstruction algorithms to select B decays with different topologies, the goal is ~10 SI95·s per event. This can be compared to the estimate for the full reconstruction, which is 100 SI95·s per event. These estimates have been made under the assumption of a significant optimisation of the current software. Estimates that are more accurate are expected once the high level trigger studies have been completed and these will be described in future TDRs (Trigger and Computing).

It is expected that the high-level trigger software will be adapted and enhanced with time as experience is gained running the experiment. It will therefore be a considerable advantage to have the full event data immediately made available such that full flexibility can be used in developing

algorithms that can make use of data from all detectors. This will have implications on the choice of the readout protocols, as will be seen in later chapters.

2.4 LHC Accelerator

A dedicated communication protocol will be needed to communicate control and status information between LHCb and the LHC machine. Information from the machine indicating particle intensities and currents, collimator and magnet settings and current operating mode (e.g. 'injecting', 'stable beams', dumping' etc.) must be interlocked with LHCb operations to ensure that the detector is always in a safe operational state. It will also be necessary to log parameters of the accelerator, such as the energy of the beam. The LHC machine will also provide the master clock corresponding to the bunch crossing rate (40.08 MHz) and this must be distributed to the front-end electronics and trigger system via a low-jitter timing distribution system.

Conversely, LHCb will provide the LHC machine with relevant information about the LHCb experiment, such as the status of the magnet, estimates of the background conditions in our detector and measurement of the luminosity (if available). There might be a need to prevent the LHC machine from continuing its current activity, should the radiation conditions in the LHCb detector become unacceptable. This implies a fast feedback and interlock mechanism between the LHCb experiment and the LHC machine. This mechanism and its triggering is outside the scope of this TDR. It should however be possible to trigger the appropriate action from the ECS system as well.

The mechanisms through which information is exchanged between experiment and machine should be standardized and be flexible such that new information can be added when required. It is not expected that the information be updated very frequently, i.e. on a time scale of seconds. This issue is being addressed in common between the 4 LHC experiments and the LHC machine in a joint working group[2].

2.5 Data Processing and Offline Computing

The DAQ system is responsible for formatting data such that their origin can be identified and their integrity verified. In a continuous mode of running, it is practical to reconstruct events promptly as the data are collected. This will give immediate feedback to the shift crew on detector performance and immediate access to the physics. This requires that an accurate calibration and alignment of the detector can be achieved in real-time and that the appropriate parameters can be made available to the reconstruction program. Further data processing will be done by offline applications.

The offline software will also require information on the settings of the trigger, and of the detector support systems (high and low voltage, gas compositions and pressures, operating currents etc.). These data vary with time and therefore this information will have to be recorded in such a way that it can be time-correlated with the corresponding event data. A data repository (Detector Conditions Database, [8]) targeted towards accessing contents by time will be required to store this information, together with calibration and alignment data. The performance of the software in retrieving the information corresponding to the event being processed should scale well as the size of the database increases. This repository will need to be replicated in remote computing centres wherever LHCb data are processed and analysed.

Re-processing of the data must be envisaged to take account of changes in the alignment and calibration as well as in the software used to reconstruct the events. It is also foreseen that the

[2] Data Interchange Working Group.

online CPU facility, which normally executes the high-level trigger algorithms and prompt reconstruction on the event data, will be used during shutdown periods as a computing infrastructure for re-processing (re-reconstruction) of the event data taken during the previous data-taking periods. This means that complete access to the conditions database has to be provided for this running mode.

The physics data will need to be stored on permanent media for a long time and have to be accessible any time from any institute in the collaboration. A natural place to effect the storage is the CERN computer centre. Hence, we will buffer the raw data and the output of the reconstruction temporarily within the online system and send them immediately through links to the computer centre (i.e. making use of the central data recording facility) where they will be stored permanently on magnetic tape.

2.6 Experiment Operations

It should be possible to operate the experiment with a small shift crew of 2 people present in the control room. This implies that control and operation of all aspects of the online system must be accessible from a central console under the command of the shift crew. The main console should therefore have access to the control of all sub-detector support systems (e.g. high voltage) and to the charts and histograms that are used to monitor the integrity of the data coming from all the apparatus.

Many members of the shift crews will be non-experts of the online system. To obtain maximum efficiency of the experiment the system presented to the operators should be user friendly and intuitive, but also self-explanatory by making use of extensive help facilities. Also as many as possible of the routine procedures should be automated, such as starting data taking, raising and lowering the high-voltages of the detector or recovery from common errors.

Remote operation of the experiment or parts of it must be possible to allow experts to exercise control over the equipment to fix problems or improve the performance. This implies that the control system has to be distributed and network-based.

2.7 Running Modes and Partitioning

Partitioning is an important concept denoting the possibility to sub-divide the LHCb online system into smaller functional parts that can be operated independently and concurrently. This notion has significant implications on the design of the system, specifically on all aspects of controls (Fast Controls and Experiment Controls), since it is the ability to control the partitions independently that will allow this requirement to be fulfilled. In the data-flow sub-system, partitioning has to be taken into account at the level of the layout and assignment of components to possible partitions so as not to break the operational independence. For example, if a readout module were shared between two sub-detectors, it would be impossible to initialise the module by one sub-detector, since it would disrupt data taking by the other.

Partitioning will show its power when being used for operating different sub-detectors under different running conditions. A multitude of running modes can be envisaged [10], such as

- Normal physics data taking
- Pedestal and electronic gain calibrations
- Timing calibrations
- Alignment calibration

- Test and debugging activities

The system has to be designed for optimal physics running. However, nothing in the system should prevent the other activities, even if they are given lower priority and run with lower efficiency.

2.8 Infrastructure Services

As with the LHC machine there will also exist an information interchange between the LHCb experiment and CERN's infrastructure services: the technical service and the safety service. This information flow will be mostly unidirectional and will comprise items such as:

- The state of the power distribution system
- The state of the cooling and ventilation system
- Information about safety warnings and alarms at or around the LHCb experimental area
- The lists of personnel accessing the LHCb pit

This information, together with information on environmental parameters (such as: temperature, humidity, radiation levels, etc.) gathered by the experiment itself will be used in order to keep the operators informed and also to protect the sub-detectors and associated equipment from undesired conditions.

There will be three levels of protective actions aimed at preventing damage to the experimental equipment:

- The first level is performed by software, and it will be based on correlations using the above-mentioned information. It can perform actions in an organized and orderly manner. For example, if a temperature in a rack rises above a certain limit the LHCb experiment control system would switch off the crates in this rack one by one and then the rack itself.
- The second level is hardwired, it will be based on sensors installed by LHCb in well-chosen locations and simple logic decisions leading to crude actions. For example if a temperature rises above a certain level (higher than the software threshold) the power to the entire counting room would be cut. This task is the responsibility of LHCb's Detector Safety System (DSS).
- The third level is also hardwired, it will be activated for problems leading to personnel danger and would take even stronger actions, such as cutting the power to the experimental area. This is the responsibility of the CERN Safety System.

2.9 Summary of Performance Requirements

The LHCb DAQ system is designed against the parameters compiled in Table 30. These parameters represent conservative best estimates based on current knowledge.

The high level triggers have the full event available after each positive Level-1 decision. The high level trigger can then be applied in a series of steps of increasing refinement until the event can be either positively accepted or rejected. Broadly speaking we distinguish between two basic steps in this procedure.

The first step, which we call Level-2, is designed to match vertex information provided by the silicon detector with the momentum information provided by the tracking system. This identifies and rejects L1 triggers with fake displaced secondary vertices. Most of the Level-2 CPU requirement comes from the momentum measurement, and existing algorithms have been benchmarked at about 0.1 SI95·s per track.

Table 3 DESIGN PARAMETERS FOR THE INITIAL LHCb DAQ SYSTEM

Parameter	Value
Average Physics Event-Size[+]	100 kB
Average Level-1 Trigger rate	40 kHz
Average Total Data Rate	4 GB/s
Average CPU power for Level-2 Algorithm	1 SI95·s/event
Total CPU power for Level-2 algorithm	40000 SI95
Average Level-2 Accept Rate	5 kHz
Average CPU Power for Level-3 Algorithm	10 SI95·s/event
Total CPU power for Level-3 algorithm	50000 SI95
Average Level-3 Trigger Rate	200 Hz
Average CPU Power for final reconstruction	100 SI95·s/event
Total CPU power for final reconstruction	20000 SI95
Average Event Size to Storage (RAW/ESD data)	200 kB
Average data rate to Storage	40 MB/s

[+] Larger events, up to ~5 MB, must be accepted, albeit at a reduced readout rate.

After optimisation we expect that 1 SI95·s/event will be sufficient for executing the full Level-2 algorithm, with the result that an installed capacity of 40000 SI95 will be required for Level-2. The main effect of Level-2 would be a strong increase in b-purity of the selected events. The rate would then be of the order of 5 kHz.

The second step, which we call Level-3, uses refined and optimised reconstruction algorithms to select B decays with different event topologies (charged two body, dilepton, low multiplicity with neutrals, etc. [9]). The current estimate of the CPU required is 10 SI95·s/event leading to an installed capacity requirement of 50000 SI95.

The trigger rate to storage (200 Hz) has been estimated from what we can reasonably afford to store and process in the initial phase of understanding in detail the behaviour of our detector. N.B. The expected rate of interesting physics events is estimated to be only a few Hz. The trigger software will be adapted as this understanding evolves and the rate to storage may therefore be expected to decrease with time. Assuming a running period of 120 days, we therefore expect to accumulate raw data at a rate of ~2 TB a day, or ~200 TB a year.

The end result of the high level trigger processing is to completely reconstruct those events that pass all trigger cuts using algorithms working with full precision. Our best estimate of the CPU time required to complete the full reconstruction is 100 SI95.s leading to a further installed capacity requirement of 20000 SI95.

The size of the ESD data is estimated conservatively as 100 kB per event i.e. comparable to the raw data. Thus we expect to generate 2 TB of ESD data a day, and 200 TB per year.

Reprocessing of the complete year's data sample will need to be performed at least once and possibly twice. This reprocessing can be performed on the same Event Filter Farm during non-data taking periods, in particular during the shutdown. All the CPU capacity will be available, including processors normally used for the high level triggers. The time available would normally allow at least three full reprocessings of the complete data sample taken during the previous year.

A potential upgrade to achieve higher data throughputs, e.g. due to larger event sizes and/or increased trigger rates, must be envisaged. It should be possible to achieve this through a simple rescaling of the size of the readout system, i.e. by adding readout modules, and not by changing the architecture of the system i.e. by introducing new functional elements or protocols. While we

design the system to be capable of running at a Level-1 trigger rate of 40 kHz, it is a requirement that the system should be upgradeable to run at a Level-1 trigger rate of up to 100 kHz. This implies that all components in the readout stream have to be capable of running at trigger rates of 100 kHz.

Since the number of readout modules is large, the functionality of each type of module and the protocols that govern the data transfer should be as simple and reliable as possible. This will help to maximise the mean time between failures, will simplify the procedure of locating and fixing errors and hence facilitate the whole commissioning phase of the project.

Chapter 3 System Design

In the following we first state some of the design goals, describe the overall architecture and give functional details on each component of the system.

3.1 Design Goals

Given the scale of the system we have followed a number of basic design principles in order to have confidence that we can achieve a safe and reliable operation.

- The system is based around a very small number of functional components and a very simple dataflow protocol. Limiting the number of different module types has been an important design goal in order to ease maintenance and minimise cost.
- Control and data paths have been strictly separated in order to be able to diagnose and recover from system errors.
- All links between modules are via point-to-point links such that the use of busses can be avoided altogether. While this is necessary for performance reasons at the downstream levels of the dataflow system, for homogeneity reasons we adhere to this also in the upstream regions. Again, we believe, this will increase the diagnostic capabilities and hence the overall efficiency of the system.
- As far as possible, we try to use industrial equipment and adhere to standards where existing. For cost reasons, wherever possible, we use mainstream technologies, such as Gigabit Ethernet.
- As will be seen in later chapters of this document, a great deal of emphasis has been put on simulating architectures, protocols, and the various functional elements, specifically the TFC components, the Readout Units and the Readout Network. The results of these simulations gave important feedback for the final design of the system.
- The possibility to extend the range of operating parameters (event size or trigger rate) without changing the architecture or the protocols was also a major design criterion. This is important since the real running conditions will not be known until after the startup of the experiment.

3.2 System Architecture

The architecture of the system consists of two components. The first is the functional decomposition, describing the different functional elements of the system. The second are the protocols that are governing the connections between the functional elements.

3.2.1. Functional Decomposition

The overall architecture of the dataflow system is depicted in Figure 4.

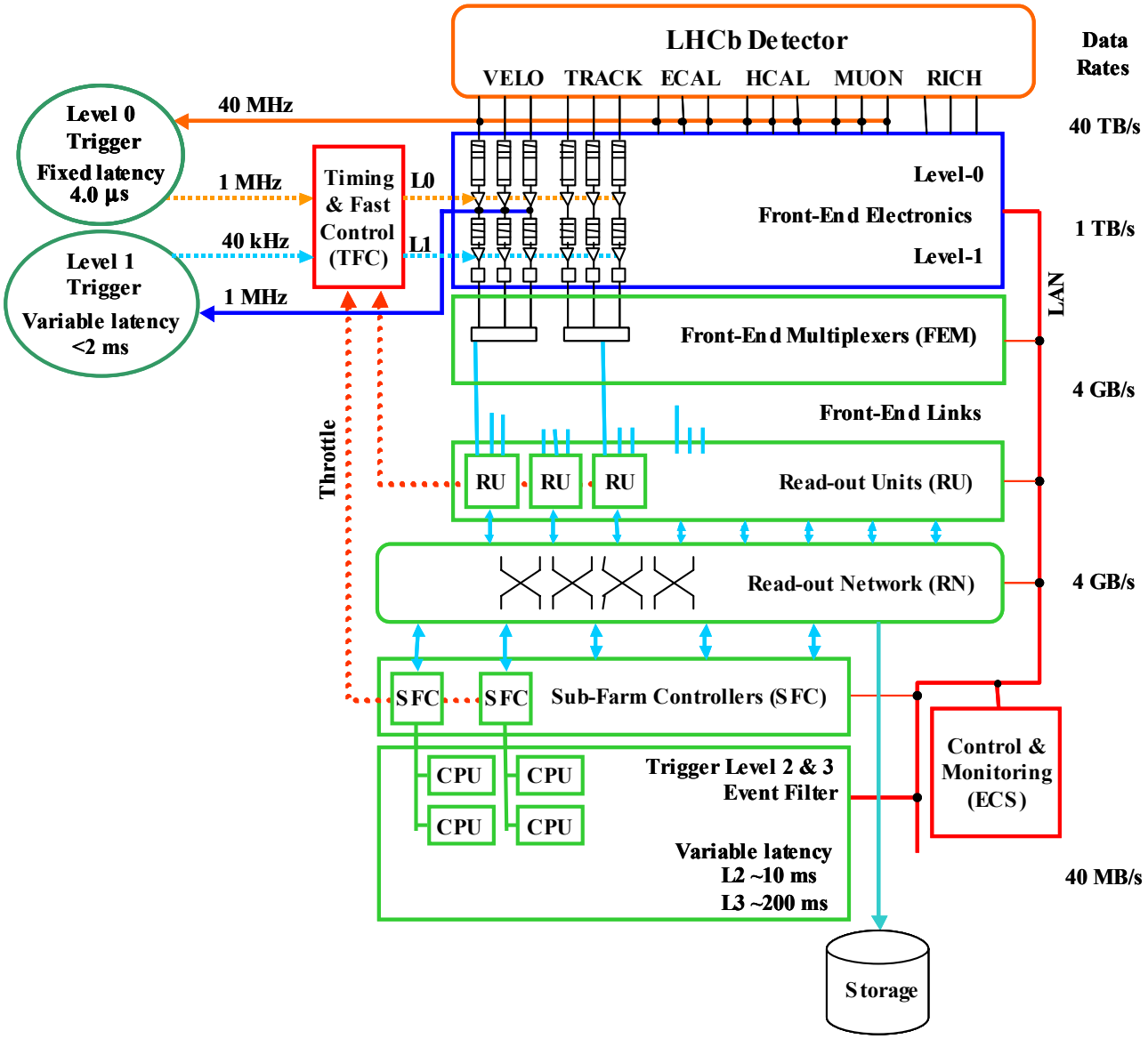

Figure 4 Overall architecture of the LHCb online system.

The main components of the system are as follows:

- The Timing and Fast Control system, which is used to distribute the clock, the decisions of the Level-0 (L0) and Level-1 (L1) trigger systems and other synchronous commands to the front-end electronics
- A data-flow sub-system that collects data from the front-end electronics and transfers them to a CPU farm for execution of the software trigger algorithms. The data-flow system itself is composed of the following elements
  - A multiplexing stage which reduces the number of links from the front-end electronics into the event building network by aggregating the data
  - A 'Readout Unit' layer acting as multiplexer and gateway between the front-end links and the readout network
  - The Readout Network, routing event fragments belonging to the same event from its inputs to a single destination
  - A layer of sub-farm controllers performing the final event building and acting as an interface and insulation layer between the Readout Network and the individual CPUs of the farm

- A CPU farm, providing the hardware infrastructure for high-level filter algorithms
- Temporary storage for physics data and general computing infrastructure for the online system
- The control and monitoring system, which is used to configure all components for data taking and to monitor their operational state. This constitutes the control path.

3.2.2. Data-Flow Protocol and Traffic Control

The protocol used to transfer data from one stage to the next is a 'push' protocol. This means that any module or stage that has data available for transfer will push them to the next higher stage immediately. There is no synchronisation or communication between components of neither the same level nor between components of different levels[1]. This protocol assumes that there is always enough buffer space available in the destination module to receive the data.

Should buffer space get scarce, traffic control has to be exercised to prevent buffer overflow. This is done through the throttle signal to the TFC system, specifically to the Readout Supervisor (see Section 3.3.3), which will inhibit the sending of new data from the Level-1 Electronics, by issuing Level-1 "No" decisions until the throttle signal is removed. In case buffer overflows are imminent, event data, but never event headers, will be removed to prevent loss of synchronization in the DAQ system.

While the protocol from the Level-1 front-end electronics to the RUs is simple, since there are only point-to-point connections between the sources of the data and the destinations, the protocol through the readout network needs a bit more attention. Since scalability is one of the important design goals of the system, a central event manager that would assign a destination to a given event is excluded. It has been decided to use a static destination assignment at the source. This means that each RU will assign a destination according to a fixed and uniform algorithm depending on the Level-1 trigger number. For example, the simplest algorithm would be to assign event number N to destination $D(x)$ where $x = N \bmod m$ (m = number of sub-farms) and D is a table containing the addresses of the destinations within the partition[2]. With such an algorithm, any ratio of CPU powers between different sub-farms can be expressed, provided the table can be made long enough. The destination assignment is, however, static, i.e. it does not take into account the current load of the individual sub-farms or even CPUs. The basic principle is based on the fact that there will be many (~10-20) CPUs per sub-farm and hence the fluctuations will be evened out. In addition, the fact that there is a lot of buffer space available in the SFCs and that the SFCs implement dynamic load balancing will ensure that no bottlenecks appear as long as the system is run with the design parameters. In case the SFC's buffers start to get full, there is still the possibility to raise the throttle signal (through the ECS) to the Readout Supervisor.

The proposed protocol satisfies the requirement from the high-level trigger algorithms that all event data have to be available to the trigger algorithm. The design goal of simplicity is also met, since the RUs do not need to wait for data requests from an event-manager or any other device before they can send the data.

The protocol between the SFCs and the storage controller is not yet defined. It could be either the same as that of the readout system (i.e. raw Ethernet) or, since the rate is expected to be very low (~3-4 Hz per SFC) it could also be a higher-level protocol, such as TCP/IP.

[1] Some link technologies, such as Gigabit Ethernet, foresee a flow-control protocol between connected ports. This could be taken advantage of to ease certain aspects of the dataflow, but should not be a mandatory requirement in the data-flow upstream of the Readout Network.
[2] Note that the RUs only know about sub-farms. The individual CPU is only accessed from its SFC.

3.3 Timing and Fast Controls

The Timing and Fast Control (TFC) system [11] controls and distributes the timing, trigger, and synchronous control information to the front-end electronics. A special feature of LHCb is that the system has to transmit two levels of high rate trigger decisions, which have to arrive synchronously at all the front-end electronics. The system must provide for means to achieve timing alignment of the front-end electronics and introduce minimum jitter. The system must incorporate functionality to prevent buffer overflows in the entire readout chain, and provide means of different types of auto-triggering for tests and calibrations. The TFC system must also support readout partitioning [12] in order to be able to run small sub-systems independently in special running modes. The system should provide statistics on the performance of the synchronous readout.

More specifically the information to be distributed includes the following:

- LHC reference clock at ~40 MHz as received from the LHC timing generators via the LHC machine interface (TTCmi). This clock drives all the electronics in the synchronous readout.
- L0 and L1 trigger decisions.
- Commands resetting event related counters in the front-end electronics used to identify the accepted events and to check synchronisation.
- Commands resetting the front-end electronics in order to prepare it for data taking or to recover from an error condition.
- Calibration commands activating specific calibration systems in the front-end electronics or in the sub-detectors. The TFC system must have a mechanism to guarantee that triggers corresponding to calibration events are accepted.

3.3.1. TFC Architecture and Partitioning

Figure 5 shows a logical picture of the TFC architecture. In order to simplify the implementation of a partitionable system, the entire mastership of the Timing and the Fast Control has been implemented in one module: the Readout Supervisor. It receives the LHC bunch clock and the orbit signal from the LHC machine interface (TTCmi) [4], and the L0 and the L1 triggers from the trigger Decision Units, and has the crucial task of providing the functionality listed above.

The front-end electronics contains buffering to store the incoming data for the latency of the Level-0 and Level-1 triggers and these buffers should never be allowed to overflow. Buffer overflows are prevented by monitoring the occupancy of the buffers in two different ways. The occupancy of the L0 derandomisers and the L1 buffers in the front-end electronics [7] are emulated centrally by the Readout Supervisor. Buffers further down the readout chain monitor their occupancy locally and, in case the buffers get nearly full, signal the Readout Supervisor via a hardwired signal (“Throttle”). Overflow is prevented in the Readout Supervisor by throttling the triggers, i.e. converting trigger accepts to trigger rejects until the occupancy is reduced.

As a consequence of its primary role, the Readout Supervisor must firstly be highly reliable. In addition, it must also be versatile in order to support many different running modes, such as tests, debugging, various types of calibrations, physics data taking etc.

As shown in Figure 5, the system incorporates a pool of Readout Supervisors, one of which is reserved for normal data taking. The other Readout Supervisors are connected to local trigger decision units and are used during tests, calibrations and debugging.

The TFC Switch [14] realises the partitioning of the TFC system. It is a programmable patch panel that allows distribution of the synchronous information to different parts of the front-end electronics. It can be programmed to distribute the information from one Readout Supervisor to one

part of the front-end electronics and simultaneously distribute information from another Readout Supervisor to a second part of the front-end electronics. The two Readout Supervisors can be configured to sustain completely different timing, triggering, and control. The two sub-systems are independent and define two different partitions. In the example in Figure 2, the leftmost Readout Supervisor controls RICH2 in a stand-alone test, while the Readout Supervisor in the centre controls all the other sub-detectors for physics data taking. The three other Readout Supervisors are idle and can be reserved for setting up and driving other partitions.

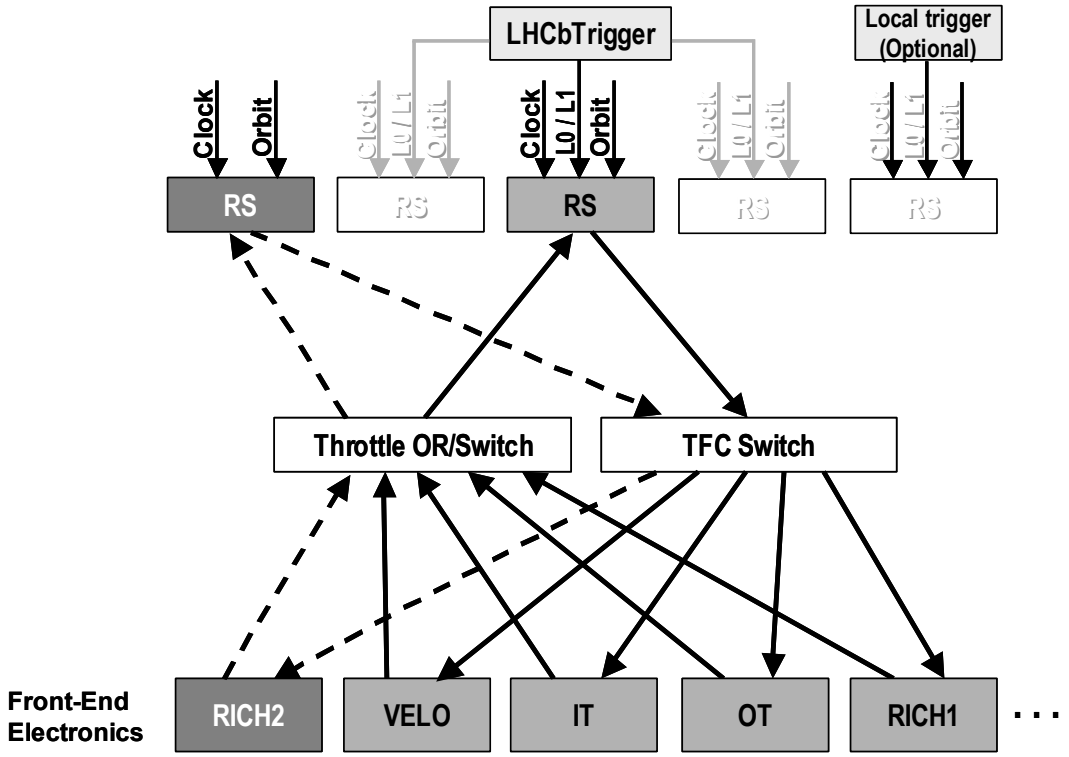

Figure 5 The TFC architecture simplified to show an example of partitioning. The partition containing RICH2 is driven by the most left RS triggered internally. The other sub-detectors are driven by the RS in the centre connected to the LHCb trigger system. The other RS' are unused.

The Throttle Switch [14] feeds back the throttle signals to the appropriate Readout Supervisors from the L1 trigger system, the L1 de-randomisers in the front-end electronics and components in the data-driven part of the DAQ system in case of imminent buffer overflows.

Figure 6 shows the TFC architecture in more detail. Several Readout Supervisors are connected to the trigger decision units to be able to run stand-alone tests with physics triggers. There is one Throttle Switch for throttle signals that throttle the L0 trigger, and one Throttle Switch for throttle signals that throttle the L1 trigger. The TFC distribution network is based on the RD12 Trigger, Timing, and Control (TTC) system [4] used by all four LHC experiments. The TTC system distributes the timing, trigger, and control information optically on two serial channels. Channel A is a low-latency channel that allows transmission of a one-bit trigger signal at 40 MHz. Channel B can transmit two different types of broadcast, which can include 6 or 16 bits of user defined information. The LHCb TFC system utilises the TTC Transmitter (TTCtx) for the conversion of the TTC signal from electrical to optical. TTC receiver chips (TTCrx) incorporated in the front-end electronics receive the TTC signals and decode the channel A and the channel B information.

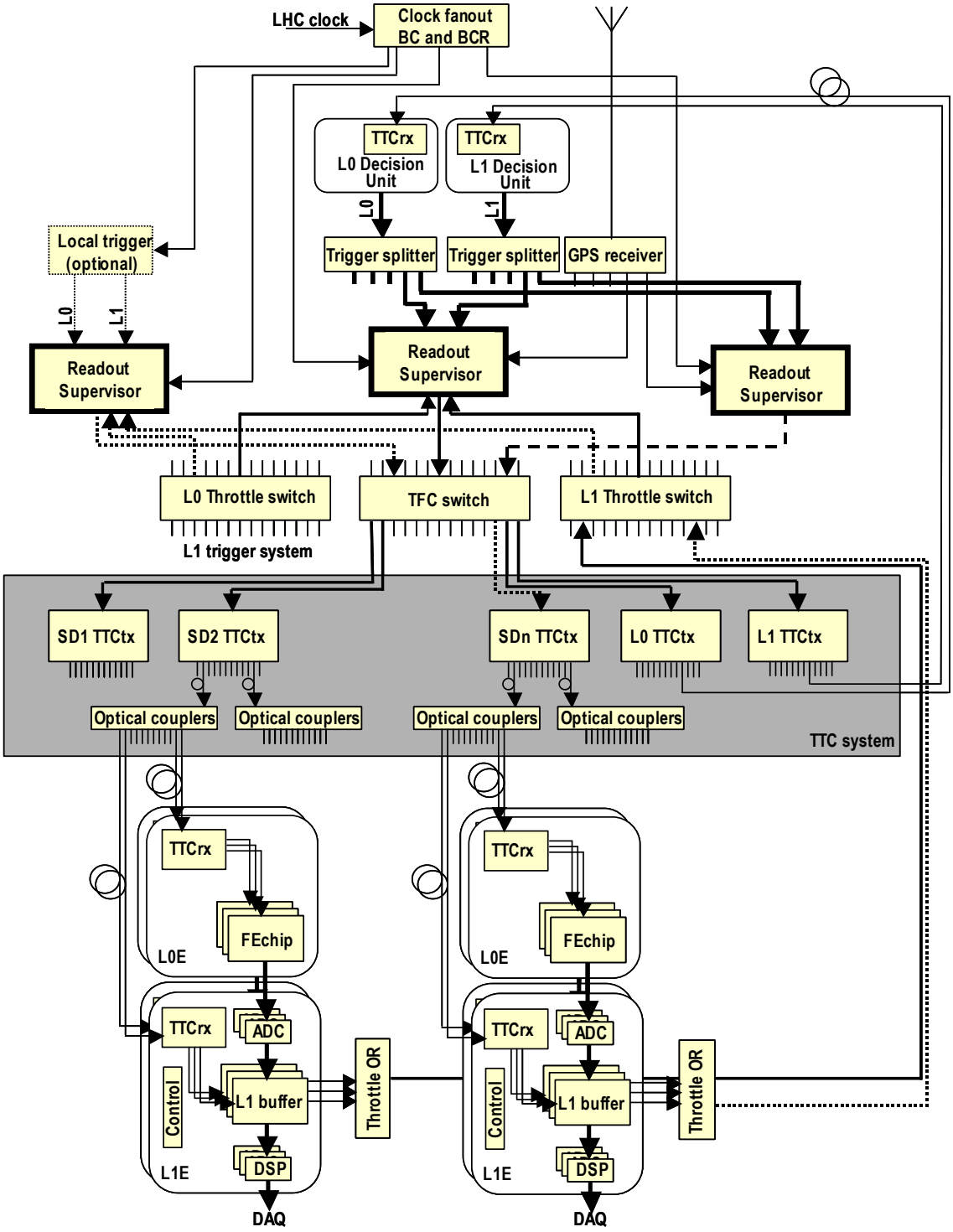

Figure 6 Overview of the TFC system architecture.

The Throttle ORs [14] form a logical OR of the throttle signals from sets of front-end electronics and readout components further down the readout chain. A GPS system allows time stamping status information sampled in the Readout Supervisor.

3.3.2. TTC Distribution System

LHCb is different from the other LHC experiments as it has to transmit two levels of high rate triggers to the front-end electronics: the L0 trigger at 40 MHz and the L1 trigger at ~1 MHz[3]. Nevertheless, the functionality of the TTC system has been found to suit well the LHCb application. The LHC reference clock is transmitted to the front-end electronics using the TTC bi-phase signal. Channel A is used to transmit the LHCb L0 trigger decisions to the FE electronics in the form of an accept/reject signal at 40 MHz.

Channel B is used for several functions:

- Transmission of the commands to reset the Bunch Counters (BCR) and the Event Counters (ECR) in the front-end electronics and the trigger systems. The Bunch Counter counts bunch-crossings, and the Event Counter counts the number of accepted L0 triggers, which in LHCb is referred to as the L0 Event ID.
- Transmission of the L1 trigger decision.
- Transmission of front-end control commands, e.g. electronics resets, calibration pulse triggering.

The information is transmitted in the form of the short TTC broadcast format. The short broadcasts contain six bits of user-defined information and two bits that have been reserved in the TTC system to reset the L0 Event ID (Event Counter) and the Bunch Counter. The different commands listed above are encoded in the six user-bits.

In principle, the TTC channel B bandwidth would allow up to a rate of 2.5 MHz of short broadcasts. However, since this is a unique use of channel B that was not foreseen initially and it is crucial to LHCb, this has been a critical test to perform. Tests have been made and broadcast rates of 1.7 MHz have been measured (see Section 4.1.1).

The TTC receiver chip also provides means to adjust the timing of the TTC information in order to time-align all front-end electronics.

3.3.3. Readout Supervisor

The Readout Supervisor (RS) has the crucial task of controlling the synchronous readout of LHCb. Therefore it must be designed with emphasis firstly on reliability. Secondly, it must be versatile in order to control the readout in the most efficient way and support a wide spectrum of running modes for tests, debugging, and calibration. It may also be necessary to change or add functions in order to handle changes, upgrades, or even unforeseen situations. Therefore, a design criterion has also been modifiability. Below is a short summary of the Readout Supervisor functions. A complete description can be found in [15].

The Readout Supervisor receives the L0 and the L1 trigger decision from the L0 trigger Decision Unit (L0DU) and the L1 trigger Decision Unit (L1DU), respectively. In order to verify that the decision units are synchronised, event identifiers accompany the trigger decisions.

The Readout Supervisor also provides several means for auto-triggering to be used in conjunction with tests and calibration runs: random trigger, periodic trigger, triggering at a programmable time after sending a command to fire a calibration pulse etc.

If the physics trigger rate gets abnormally high or data congestion occurs in the system, there is a potential risk of overflow in the buffers in the front-end electronics and in the DAQ system. In order to prevent this, the Readout Supervisor controls the trigger rates according to the status of the

[3] The nominal Level-1 decision rate is 1 MHz. The maximum Level-0 accept rate is 1.1 MHz and is a consequence of the specification to the Level-0 electronics that one event has to be processed within a maximum of 900 ns [13]

buffers. The status of the buffers is either emulated centrally in the Readout Supervisor or they are monitored locally. In case they are monitored locally, imminent overflows are signalled via the dedicated throttle signals. Data congestion at the level of the Event Filter Farm is signalled via the Experiment Control System (ECS) to the Readout Supervisor.

The Readout Supervisor also has the task of transmitting various synchronous reset commands in order to prepare the front-end electronics for data taking or recover from an error condition.

The Readout Supervisor provides statistics on the performance and the efficiency of the synchronous readout (dead-time, errors, etc) and records local event information that is appended to the event data.

The clock, the L0 and the L1 triggers, and all the control commands are encoded and transmitted by the Readout Supervisor to the front-end electronics as a TTC signal.

3.3.4. TFC Switch

As shown in Figure 7, the TFC Switch allows setting up a partition by associating a number of partition elements (e.g. sub-detectors) to a specific Readout Supervisor. The Readout Supervisor can then be configured to control and trigger the partition in whatever specific mode that is required. Note that the TFC Switch is located before the TTC optical transmitters (TTCtx) and that it is handling the encoded TTC signals electrically.

From the architecture of the TFC system, it follows that the front-end electronics that is fed by the same output of the TFC Switch is receiving the same timing, trigger, and control information. In other words, a part of the front-end electronics connected to a TFC Switch output cannot be operated in a different running mode from another part belonging to the same output. Hence, the association of the front-end electronics to the different outputs of the TFC Switch defines the boundaries between the smallest sub-systems that can be operated independently.

The TFC Switch has been designed as a 16x16 switch and thus allows the LHCb detector to be divided into 16 ‘atomic’ sub-systems. To increase the partition granularity an option exists whereby four TFC Switches are deployed in order to divide the LHCb detector into 32 sub-systems.

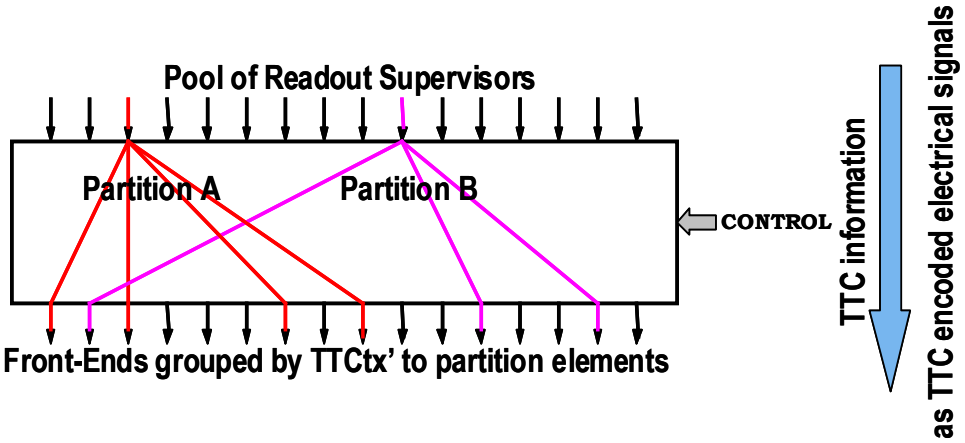

Figure 7 The principle of the TFC Switch. The switch basically acts as a fanout between an input and any set of outputs.

Since the front-end electronics are susceptible to jitter on the TTC signal, the TFC Switch must introduce less than 50 ps of jitter. In addition, the front-end electronics should be time aligned in order to sample the detector signal at the optimal point. However, different Readout Supervisors may be used to operate the front-end electronics at different times, which in reality means that the TFC signals take different paths in the TFC Switch. Since the front-end electronics is susceptible to

the timing, it is crucial that the propagation delays of all paths in the TFC Switch are equalised. The aim is that the phase difference between output ports using any input should be less than 100 ps.

3.3.5. Throttle Switch and Throttle OR

The function of the Throttle Switches is to feed back the throttle information to the appropriate Readout Supervisor, such that only the Readout Supervisor in control of a partition is throttled by the components within that partition. Figure 8 shows an example of how they are associated. The logical operation of the Throttle Switch, as the TFC switch also a 16x16 switch, is to perform a logical OR of the inputs from the components belonging to the same partition. The system incorporates two Throttle Switches, a L0 and a L1 Throttle Switch. The sources of L0 throttles are essentially the components that feed the L1 trigger system. The sources of L1 throttles are the L1 de-randomisers and the event building components.

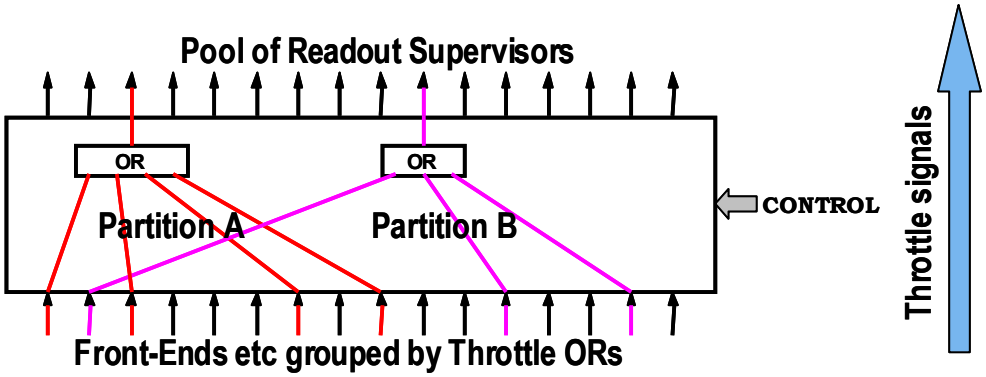

Figure 8 The principle of the Throttle Switches. The switch acts as a programmable OR between any set of inputs and one output.

The Throttle ORs group throttle lines belonging to the same partition elements. They are identical to the Throttle Switches in all aspects except that they OR 32 inputs and have only one output.

3.4 Dataflow System

As mentioned in Section 3.2 the data-flow system is composed of four distinct components. These are responsible for transporting the data from the Level-1 front-end electronics to permanent storage. In the following sections, we will describe the functionality of these components in detail.

3.4.1. Front-End Multiplexer Layer

The purpose of the Front-End Multiplexer (FEM) is to aggregate the data fragments originating from several Level-1 front-end electronics boards, which have very low data rates into bigger fragments with the final aim of reducing the number of links into the readout network and making better use of the single link bandwidth. As an example we take the case of the VELO detector, which is expected to deliver 3.2 MB/s from each Level-1 front-end electronics board (cf. Table 7). It would be highly inefficient and costly to feed each of the 100 VELO-links into the readout network. Therefore, it is advantageous to aggregate ~25 of those links onto one input of the readout network. On the other hand, the SPD/PS detector will feed 39 MB/s on one link into the DAQ system. There is clearly not too much room for aggregation in this area and hence, a flexible data aggregation scheme is needed.

The aggregation is done by merging the data corresponding to the same event-number arriving on different input links, after having removed the transport headers and trailers. The resulting data are

framed again with transport information and sent out on the output link to the next higher stage in the readout. Figure 9 shows graphically the event building process in the FEM modules. The data fragments of *n* input streams (in Figure 9 *n* is equal to 4) are merged according to the event number on one output stream, while the original event building information contained in the headers is removed and substituted by a single new header reflecting the characteristics of the newly created data fragment.

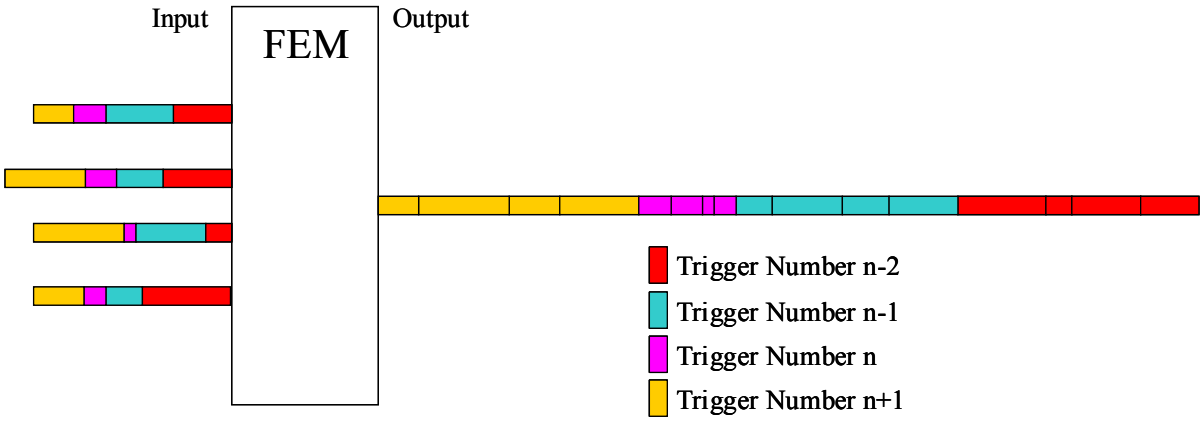

Figure 9 Schematic view of the data aggregation or event building process

3.4.2. Readout Unit Layer

The functionality of the Readout Unit (RU) is in the first instance the same as that for the FEM (3.4.1). The Readout Unit collects data from several input links, concatenates and buffers them and finally dispatches them into the Readout Network. If the Readout Network becomes congested it can, for certain technologies, suspend the sending of data to it, and hence block the Readout Units. This can in turn lead to congestion in the Readout Units, and implies that significant buffering is needed in these modules[4]. This is, by construction, not the case for the FEMs, since the readout protocol applied between FEMs and Readout Units assumes that there is always buffer space available in the Readout Unit to receive the data from the FEMs. Should there be scarcity in the buffer space in the Readout Unit, it will issue the hardware throttle to the Readout Supervisor in order to stop the issuing of Level-1 accept decisions to the front-end electronics.

The Readout Unit must adapt the protocol used on the input links to that of the Readout Network. The extent of this translation will be depending on the choices for the input link technology and the technology for the Readout Network. In terms of networking, they act as a gateway between the two technologies.

A third feature required of the Readout Units has to do with the event building process. Unlike the FEMs, the RUs can send their data to more than one destination through the Readout Network. The fragments of a given event-number must arrive at only one destination. Hence, the RUs have to support a destination assignment mechanism (see Section 3.2.2).

3.4.3. Readout Network Layer

The Readout Network implements two main functions:

1. It provides the connectivity between the RUs and the Sub-farm Controllers such that each RU can communicate with any Sub-Farm Controller,

[4] A temporary congestion of 10 ms would lead to the accumulation of 400 event fragments in the RU, which corresponds to a buffering requirement of the order of 800 kB.

2. It provides the necessary bandwidth, such that all data arriving in the RUs can be sent to the Sub-Farm Controllers. The aggregate data bandwidth required is ~4GB/s (100 kB events at 40 kHz).

The operation of sending all data fragments belonging to a particular event, from each RU to the assigned SFC, is called *event-building*. At first sight, this data traffic would require N data paths for each end point, or N^2 data links between N RUs and N SFCs. However most of these links would be active only during 1/N of the time.

The connectivity requirement can be fulfilled, in principle, by a 'non-blocking' switching network such as those used by the telecommunications industry or for the interconnection of processors. The property of being non-blocking means that data transfers can take place in parallel between sources and destinations, for any combination where one source at most is connected to a destination at a given time. This property could be used for event-building if a global control system could change, at regular time intervals, the non-blocking interconnection pattern of the RUs and the SFCs so that after N such patterns any RU has been connected once to every SFC. This traffic-shaping scheme is called a *barrel shifter*. However, we wish to avoid such a complication since it implies the use of a global control system over the Readout Network.

We show in Chapter 4 that event-building over a "packet switching" network is feasible even if the RUs send their event fragment to the assigned SFC, without caring about possible contention in the network. In fact, packet switches provide internal buffering to resolve contention. We still have to prove that the event-building traffic, as it is specified, will not overflow those buffers, with an acceptably low probability. It will be shown that this is possible under the condition that the overall load on the network is significantly less than 100%, in other words that the installed bandwidth exceeds the value of 4GB/s mentioned above.

For the selected technology, Gigabit Ethernet, a bandwidth of approximately 6 GB/s would be sufficient to sustain the "normal" traffic with a very low probability of data loss. It is important to ascertain this since the throttle mechanism is, in principle, unable to avoid data losses due to buffer overflow inside the Readout Network. Consequently, the Readout Network should be implemented in such a way that the throttle mechanism only comes into action when an excessive data flow persists.

An additional requirement is that the readout network must be able to accept occasionally very large events carrying calibration data. Precise specifications for such events are not available presently. Such large events may possibly cause buffer overflows. This is clearly unacceptable and will require special attention (see Section 4.4.3).

To ensure scalability of the system, our choice is to avoid the use of a central event manager for assigning dynamically a destination SFC for each event. This function will be implemented as a fixed round-robin destination assignment. The implications of this choice are briefly discussed in Section 3.2.2.

A possible upgrade to higher data throughput, due to larger events and/or to a higher trigger rate, beyond the safe limits of the designed system, will necessitate a redistribution of the front-end data and an increase of the number (N) of input and output ports of the Readout Network. The size of the network, in terms of number of components (switching modules, internal links) scales roughly like NlogN, as illustrated in Figure 10, which shows the number of switching modules required to build an NxN switching network based on 4x4 switches inter-connected in a Banyan topology [16]. A pictorial view of an example for N=64 is shown in Figure 24.

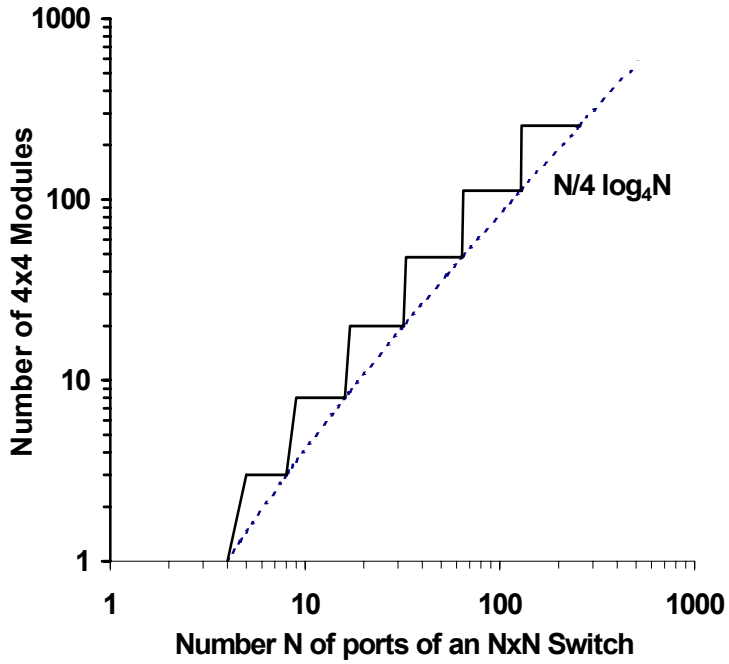

Figure 10 Number of 4x4 modules required building an NxN switching network in a Banyan topology, as a function of the number N of input/output ports. The solid line shows the number needed for a physical implementation, whereas the dotted line represents the formula N/4 $\log_4$N. The solid line always lies above the dotted since fractional modules are not possible.

3.4.4. Sub-Farm Controller Layer

The Sub-Farm Controllers (SFC) must perform three functions. Firstly, they assemble the data arriving from the RUs to form complete events. Secondly, they isolate the readout network and its technology from the network technology within the sub-farm. Thirdly, the SFCs exercise a load balancing function among the CPUs connected to each sub-farm. As can be seen in Section 2.9 an event can spend a long time in a CPU in the case it is accepted by the trigger algorithms and has to be reconstructed. A simple round-robin scheduling would lead to high buffer occupancies in the SFC and uneven loads in the sub-farm nodes. The situation can arise that one or more nodes spending very long times executing the Level-3 algorithm and, without load balancing, they would be fed more and more events, while other nodes, that get events they reject would stay idle.

The SFCs are also responsible for retrieving from the CPUs the raw data of events accepted by the High-Level Triggers (HLT), as well as the reconstructed data, and for sending these data to the storage controller via the Readout Network (Figure 4). In this way the connectivity already provided by the Readout Network and the sub-farm infrastructure is reused.

3.5 Event Filter Farm

The Event Filter Farm will execute the higher level trigger algorithms, which reduce the incoming event rate of 40 kHz to a final data recording rate of 200 Hz. Accepted events will be fully reconstructed online and the output will also be sent to storage. The average data rate into a farm node is ~ 5 MB/s and thus allows the use of cheap 100 Mbit interfaces. For each of these events the raw data (100 kB) as well as the reconstructed data (100 kB) must be stored. A small sample of events rejected by the trigger are recorded for the purpose of monitoring the trigger efficiency.

When there is no data taking, for example during shutdown periods, the full farm will be used for reprocessing the data.

The requirements on processing power and storage capacity are summarised in Table 4. They are justified in more detail in reference [17]. The infrastructure of the Event Filter Farm comprises not only the CPUs executing these algorithms, but also the means to configure, control and monitor them.

Balancing the minimal number of RUs required for the readout network[5], the farm is segmented in ~ 60 sub-farms. Each sub-farm consists of one Sub-Farm Controller (SFC) with its associated CPU nodes. Each sub-farm will initially be equipped with the same number of nodes. Subsequent upgrades will preserve the same structure (i.e. mixture of nodes) in each sub-farm. The system is thus scalable "in depth" by simply adding nodes to the sub-farms. With time, each sub-farm will evolve to resemble a heterogeneous collection of CPUs having different processing powers. One of the tasks of the SFC is to balance the load such that the most powerful CPUs process a correspondingly larger fraction of events.

A farm-node will require CPU power and memory, but it will most likely not require a hard-disk, and certainly no support for graphics or multi-media applications. It will need two network interfaces such that the separation between data and control paths is maintained. Remote console access/hardware management will be possible via the experiment control system. Finally, a CPU node should be economical in terms of power, floor space, cooling etc.

Table 4 DATA VOLUMES AND CPU REQUIREMENTS FOR PROCESSING AND STORAGE OF DATA IN THE EVENT FILTER FARM.

Requirement	Value
Rate of events to storage	200 Hz
Total number of events per day	$\sim 2 \cdot 10^7$
Raw data size per event	100 kB
Reconstructed data size per event	100 kB
Total CPU Power	110000 SI95
Total raw data per day	2 TB
Total reconstructed data per day	2 TB

3.6 Experiment Control System

LHCb will have a homogeneous control system. The Experiment Control System (ECS) will handle the configuration, monitoring and operation of all experimental equipment involved in the different activities of the experiment:

- Data acquisition and trigger (DAQ)
  Timing, front-end electronics, readout network, Event Filter Farm, etc.
- Detector operations (DCS)
  Gases, high voltages, low voltages, temperatures, etc.
- Experimental infrastructure
  Magnet[6], cooling, ventilation, electricity distribution, detector safety, etc.

[5] As simulation (see Section 4.4.2) has shown, this number of RUs is sufficient for coping with the network load. Symmetry suggests to have the same number of sub-farms, however a larger number would also be possible. To have a system with fewer sub-farms than RUs is unwise, because of the risk to overload the single link into a sub-farm.
[6] The LHCb magnet will actually be operated in collaboration with the LHC machine.

- Interaction with the outside world
  Accelerator, CERN safety system, CERN technical services, etc.

The relationship between the ECS and other components of the experiment is shown schematically in Figure 11. This shows that the ECS provides a unique interface between the users and all experimental equipment.

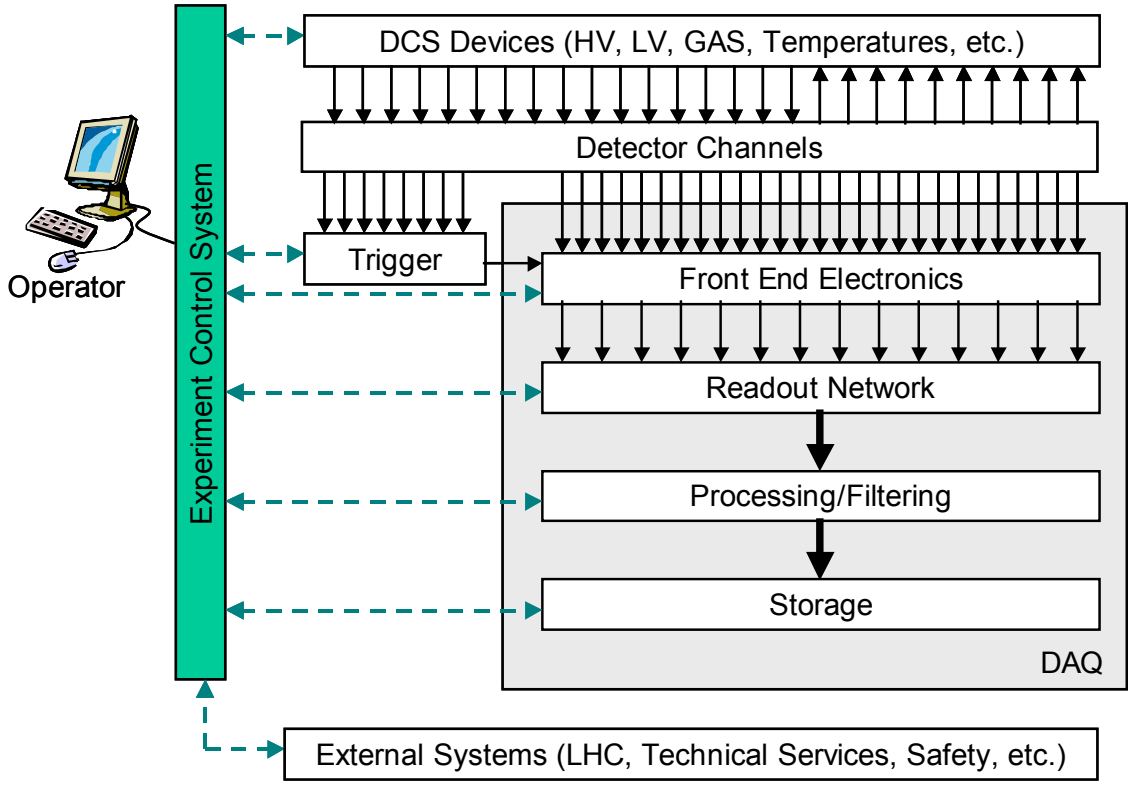

Figure 11 Scope of the Experiment Control System

3.6.1. ECS Architecture

The main task of the control system is to configure, monitor and control the detector's hardware equipment. This task is mainly accomplished by sending commands and settings to the equipment and reading back information. The control system can take decisions on its own, e.g. to recover from errors, and let the operator interact with the system by presenting the information to him/her and accepting commands. All information regarding the equipment (geographical location, access addresses, settings for different running modes, etc.) resides in a configuration database. This database is an integral part of the control system. Since the operation of the detector depends on external conditions the control system also needs to exchange information with external entities, such as the accelerator, CERN Technical Services, etc. A subset of the data gathered by the control system is needed for the offline analysis. These data are stored in the conditions database. Figure 13 shows the ECS context diagram.

From the hardware point of view, the control system will consist of a small number of PCs (high-end servers) on the surface connected to large disk servers (containing databases, archives, etc.). These will supervise other PCs (hundreds) that will be installed in the counting rooms and provide the interface to the experimental equipment. Depending on its type, the equipment can be connected

directly to a PC, to a node in a fieldbus, to a PLC (Programmable Logic Controller) or to a board with VME form-factor[7] (Figure 12).

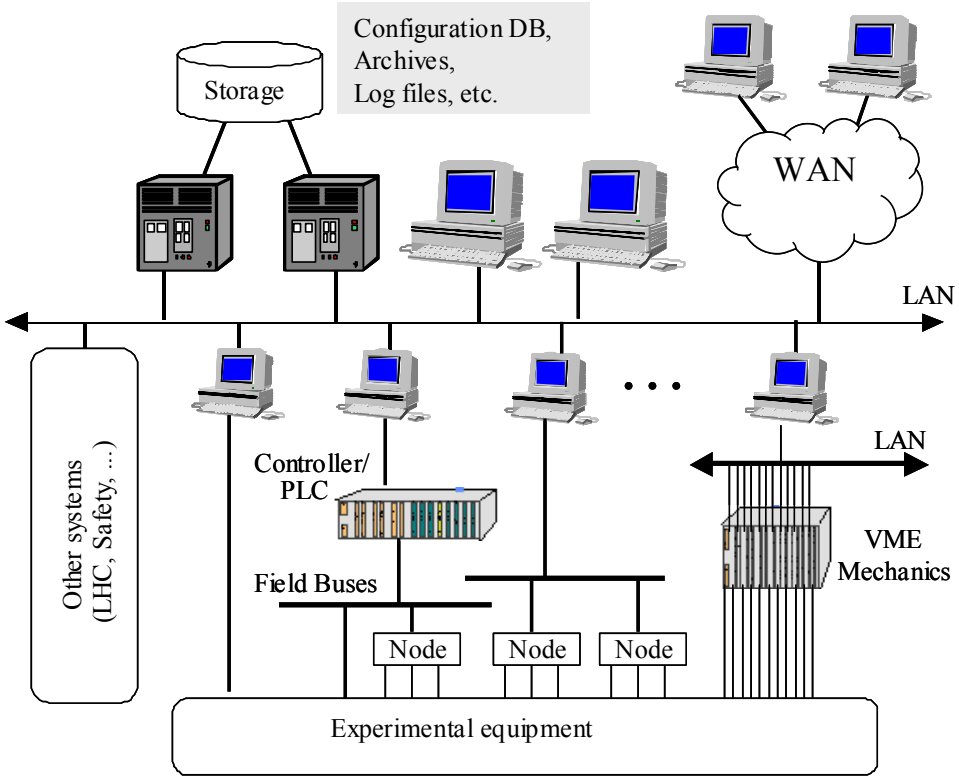

Figure 12 ECS Hardware Architecture

From the software point of view, a hierarchical, tree-like, structure has been adopted to represent the structure of sub-detectors, sub-systems and hardware components. This hierarchy should allow a high degree of independence between components, for concurrent use during integration, test or calibration phases, but it should also allow integrated control, both automated and user-driven, during physics data-taking.

This tree is composed of two types of nodes: “Device Units” which are capable of “driving” the equipment to which they correspond and "Control Units" which can monitor and control the sub-tree below them, i.e., they model the behaviour and the interactions between components. Figure 14 shows the hierarchical architecture of the system.

[7] In LHCb it was decided not to use VME Equipment. However, the LHCb standard front-end electronics boards will have the same form-factor as VME 9Ux400mm.

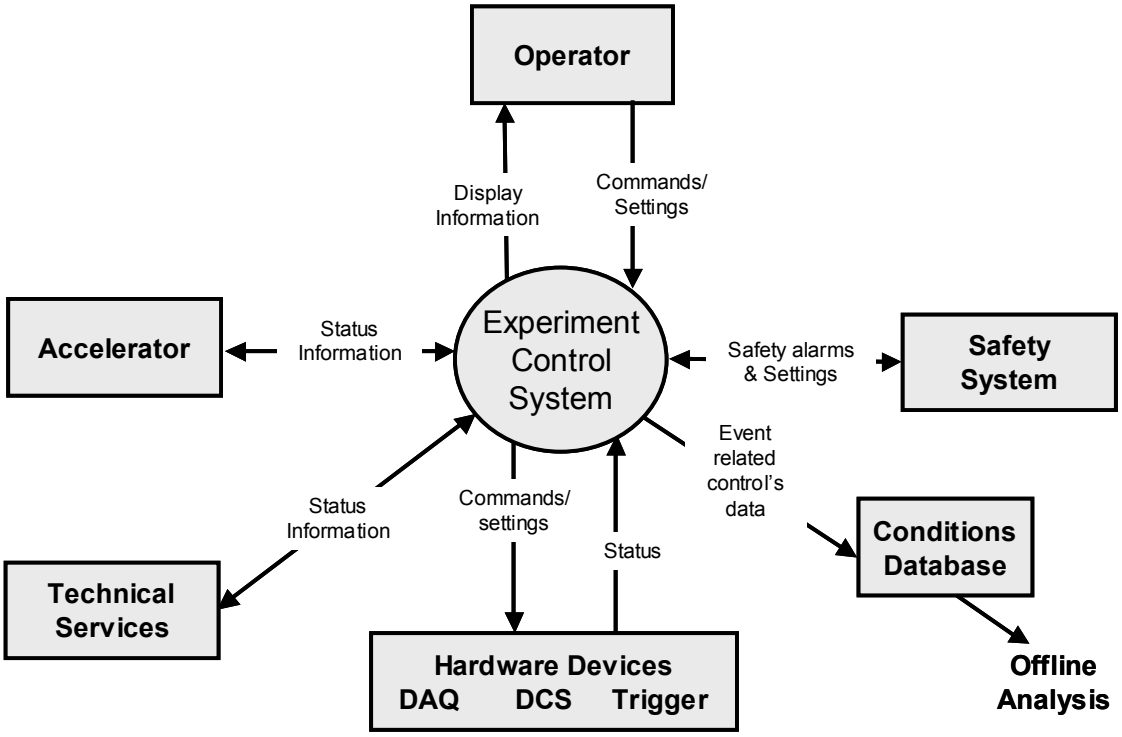

Figure 13 ECS Context Diagram

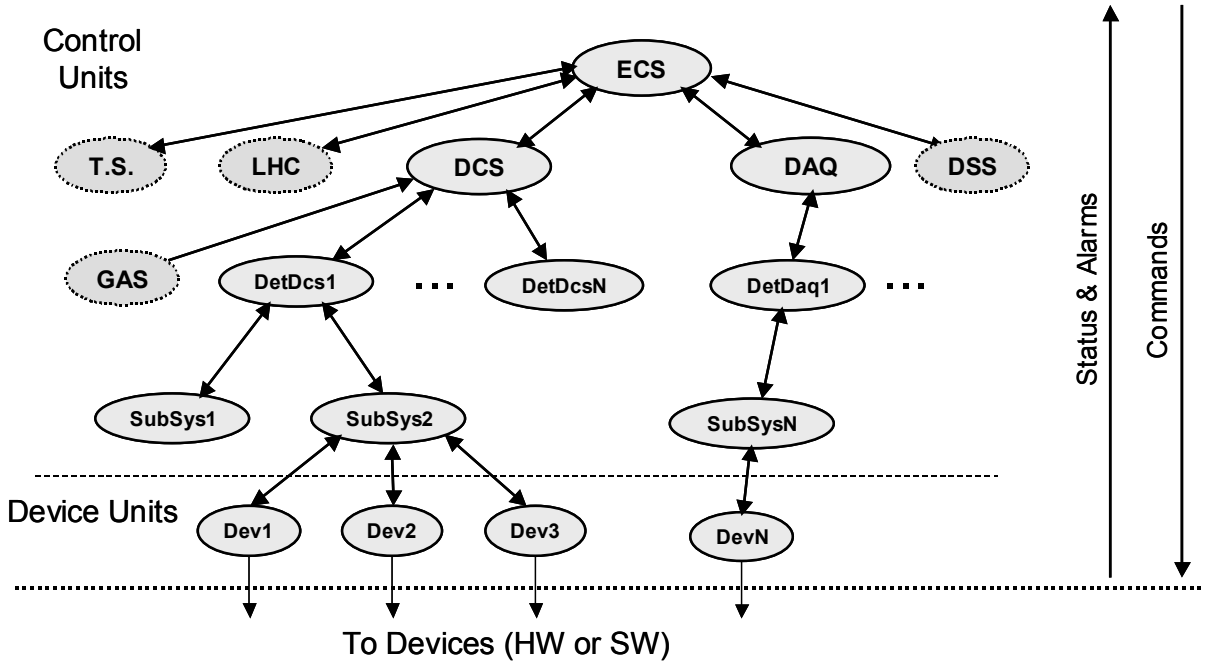

Figure 14 ECS Software Architecture. The dotted ellipses denote external systems, while the solid ones symbolize DCS and DAQ entities.

3.6.2. ECS Design Concepts and Guidelines

The architectural design of the software framework is an important issue. The framework has to be flexible and allow for the simple integration of components developed separately by different teams and it has to be performing and scalable to allow a very large numbers of channels.
In order to allow a coherent integration of ECS sub-systems, a single control framework will be built and distributed to sub-detector developers. The control framework will be based on the JCOP

framework [18], but specifically tailored for LHCb. This framework will be composed of a set of guidelines, tools and components with the aim of simplifying the task of integrating different components to build the overall control application. and of easing the development of specific components.

Some of the components of this framework include:

- guidelines imposing rules necessary to build components that can be easily integrated (naming conventions, user interface look and feel, etc.)
- drivers for different types of hardware, such as fieldbuses, and PLCs
- ready-made components for commonly used devices configurable for particular applications, such as high voltage power supplies, credit card PCs, etc.
- many other utilities, such as data archiving and trending, alarm configuration and reporting, etc.

Some of the concepts and design choices that will help achieving these requirements are:

- Device-oriented access:
  - Device data will be described and accessed as structured data and not as separate single items (as is the case in tag-based systems). This mechanism is more flexible and allows better network performance.
  - Each type of device and its access mechanism through the appropriate driver will be described in the database. Specific devices can then easily be created as instantiations of these pre-defined device types.
- Hierarchical control and finite state machine modelling
  - It should be possible to represent the behaviour of each sub-system in a simple way. Finite State Machines (FSM) provide an intuitive and convenient mechanism to model the functionality and behaviour of a component. For example a high voltage sub-system can be described as having states “off” and “on” and transit between them by executing actions “switch on” or “switch off”.
  - It should be possible to organize the control system as a hierarchy of sub-systems (containing devices and/or other sub-systems). This hierarchy could have several levels of abstraction. For example, a sub-detector may contain several sub-systems (high voltage, low voltage, etc.) and is in turn contained in the experiment.
- Distributed and decentralized decision making
  - In order to cope with the scale of the system, the control tasks will be distributed over many machines in a transparent manner. This provides for a scalable architecture, which can be easily adapted to the required performance.
  - Sub-systems should be able to work in stand-alone mode and when necessary perform actions autonomously even when being controlled centrally. This allows for parallelism giving in general better efficiency for automated operations such as error recovery procedures.

3.7 Summary of Key Features

The architecture and the performance are inherently scalable due to the absence of a central element that has to act on an event-by-event basis (‘Event-Manager’). More performance in terms of data rate can easily be obtained by adding more RUs, and consequently more switch ports and SFCs, to the system. The limit is reached, when the output links at the Level-1 front-end electronics are saturated.

The amount of available CPU power for data processing can be increased by adding more CPUs to each sub-farm. There exists no architectural limit to the amount of CPU power that can be made available for HLT data processing.

The scalability of the ECS system is achieved through its highly hierarchical structure.

The system is conceptually simple. All components have a relatively small and well-defined functionality. The data transfer protocol is also kept to a minimum, such that the functionality in the sending nodes can be kept straightforward.

Uniformity of the system is another feature. We avoided duplication of work wherever we could. This is best represented by the uniform approach to the control of the experiment, where we use the same tools and system for controlling the DAQ system and the control of the detector.

The design is balanced. The amount of CPU power in the CPU farm and hence the required network bandwidth matches the performance a 100 Mbit Ethernet network can provide. This has significant advantages in the cost of the farm, since there is no need to acquire expensive Gb Ethernet switch ports for each CPU in the farm.

Chapter 4 System Implementation

In this chapter the detailed implementation of the system will be described, including the specific technical choices made for data links and readout modules, and also for the interfaces to the control system.

4.1 Timing and Fast Controls

The Timing and Fast Control system handles the distribution of timing, trigger, and control information to all front-end electronics. The distribution network is based on the RD12 TTC system developed within a common LHC project. However, LHCb is different from the other LHC experiments in one major respect in that two levels of high-rate triggers must be distributed to the front-end electronics. Consequently, it has been of crucial importance to test the RD12 TTC system to verify that this is feasible. The results of these tests are described below.

In addition, in designing the architecture of the TFC system special emphasis has been placed on supporting the partitioning requirements. Mastership is concentrated in one module, the Readout Supervisor, which handles all distribution of all timing, trigger and control signals. Programmable switches are introduced in the Timing and Fast Control distribution network between a pool of Readout Supervisors and the front-end electronics. Partitions are created by allocating a Readout Supervisor from the pool, together with the required subset of the electronics, and by programming the switch to provide connectivity between the two.

Attention is being given to prototyping all LHCb-specific modules and to making tests of all TFC components working together. Details are given in the following sections.

4.1.1. TTC Distribution System

Feasibility tests of the way the TFC architecture exploits the TTC transmission system have been made. In particular, a crucial point to verify was the requirement to transmit L1 triggers and commands as short broadcasts at a rate of 1 MHz on the channel B of the TTC system. In principle, this channel should be able to sustain a rate of 2.5 MHz of short broadcasts but since such an extensive use was not initially foreseen, it was important to verify that there are no limitations in the implementation of the TTC encoder or the TTC receiver.

Lacking a Readout Supervisor, a test bench was devised using existing equipment as shown in Figure 15:

- The TTCvi is TTC-VME-bus interface developed within the ATLAS experiment. It was used to transmit triggers and short broadcasts.
- The FIC, a Fast Intelligent Controller was used as a VME controller to configure the TTCvi.
- The TTCvx is the TTC encoder that will eventually be incorporated in the Readout Supervisor.
- The TTCtx is the electrical-to-optical converter that will be used in LHCb.
- The TTCpr is a PCI card with an onboard TTC receiver chip (TTCrx). It was used to receive the triggers and the short broadcasts, and to transfer them to a host PC.

The test bench was used to check the transmission rate and the integrity of the short broadcasts after serialisation and encoding in the TTCvi, and decoding and deserialisation in the TTCrx. In order to do this, 64 short broadcasts were filled with a pattern from 0 to 63 (the six user bits) and were stored in the FIFO of TTCvi. Using an external pulse generator to drive the transmission from the

FIFO, the series of broadcasts was repeated continuously at different rates. The pattern was simultaneously checked for errors at the receiving end. The measurements show that the TTC system is able to sustain a short broadcast rate of ~1.7 MHz[1]. Although some errors were observed at the maximum rate of 1.7 MHz, no errors were detected up to 1.5 MHz in short-term tests.

In summary, the TTC system has been shown to correspond adequately to the LHCb requirements.

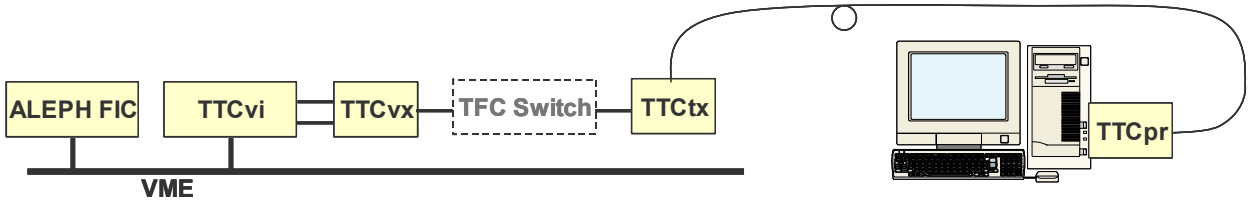

Figure 15 The TTC system test bench.

4.1.2. Readout Supervisor

The Readout Supervisor is a crucial module in the LHCb experiment as it handles all timing, triggering and control of the front-end electronics. In view of this, it has to be extremely reliable. It has also to fulfil the requirement of versatility and modifiability in order to support a large number of running modes.

In order to facilitate the implementation, the Readout Supervisor functions have been organised in logical blocks as shown in Figure 16. The functionality of each block is described below together with some details on the implementation.

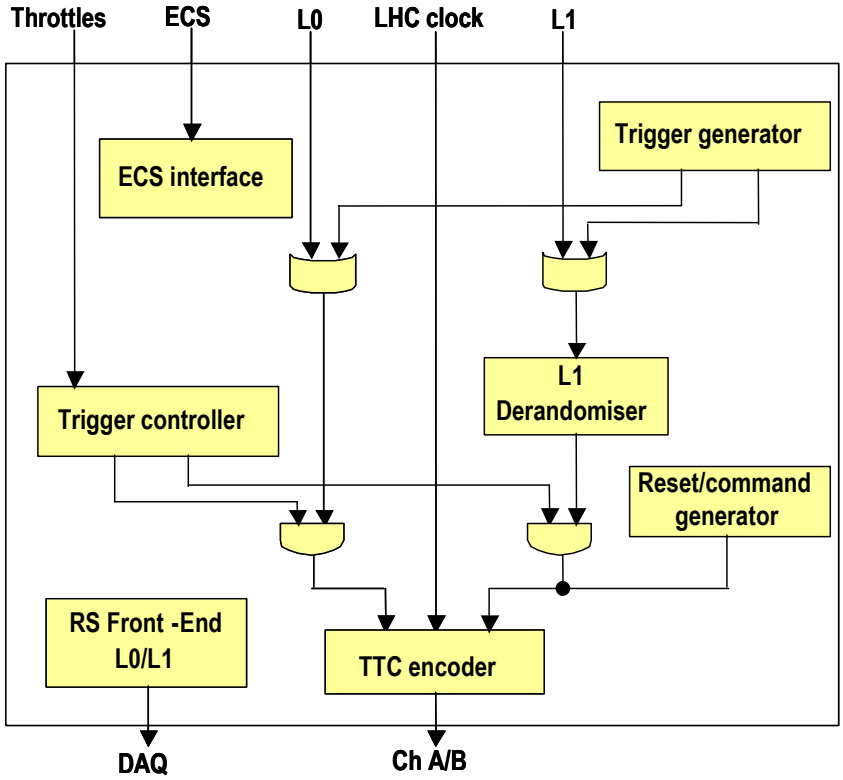

Figure 16 Simplified logical diagram of the Readout Supervisor showing the basic functions.

[1] The TTCvi allows transmitting short broadcasts at maximum every 575 ns.

TTC encoder

In order to transmit the timing, triggering, and control information to the front-end electronics, the Readout Supervisor has a TTC encoder circuit incorporated. The encoder receives directly the LHC clock and the orbit signal electrically from the LHC timing generators via the TTC machine interface (TTCmi) installed in the cavern. This minimises the jitter on the TTC signal at the output of the encoder.

L0 trigger path

The Readout Supervisor receives the L0 trigger decision together with the Bunch Crossing ID from the central L0 trigger Decision Unit (L0DU), or from an optional local trigger unit. The global latency of the L0 triggers is constant and is the sum of the numbers of cycles consumed along the L0 trigger path due to evaluation time and cables. This has been predefined with safe margin to be 160 cycles. Since the exact cable length and the number of cycles consumed within the L0 trigger system are not known yet, the TFC system must be able to accommodate extra cycles. The Readout Supervisor therefore has a pipeline of programmable length at the input of the L0 trigger (not shown in Figure 16). The depth of the pipeline will be set once and for all during the commissioning with the first timing alignment, unless changes are made later along the L0 trigger path.

The Bunch Crossing ID received from the L0DU is used to verify that the L0DU is synchronised.

Occasionally the Readout Supervisor will inject L0 auto-triggers for tests and calibrations. The Readout Supervisor provides internally a mechanism to guarantee that these are kept at Level-1.

L1 trigger path

The RS receives the L1 trigger decision together with a 2-bit Bunch Crossing ID and a 12-bit L0 Event ID from the central L1 trigger Decision Unit (L1DU). The two incoming IDs are used to verify that the L1DU is synchronised.

L1 Trigger Derandomiser

The L1 triggers are subsequently transmitted as short broadcasts containing a 3-bit trigger type and the two least significant bits of the L0 Event ID. However, the L1 buffers in the front-end electronics are implemented as FIFOs and have a constant readout time of 34 cycles (850 ns). Therefore the Readout Supervisor incorporates a L1 trigger derandomiser buffer of 8 k entries. A finite state machine sends the L1 triggers at intervals of 34 cycles.

Trigger Controller

In order to prevent overflows of the buffers in the system, the Readout Supervisor controls the trigger rates according to the status of the buffers. The control is performed by means of throttling triggers that would otherwise overflow a buffer, that is converting trigger accepts to trigger rejects. Due to the distance to the location of the L0 derandomiser buffers in the front-end electronics and the high L0 trigger rate, imminent buffer overflows cannot be signalled via hardwired signals. Instead, since the buffer occupancy depends only on the number of L0 trigger accepts and the fixed buffer readout time and is the same for all buffers, the RS has a finite state machine to emulate the occupancy. If the buffer gets nearly full, the RS throttles the L0 triggers until the occupancy is reduced. The same principle is applied to control the L1 buffers in the front-end electronics.

The buffers further down the readout chain that receive events at lower rate, such as the L1 derandomisers in the front-end electronics and the buffers in the event-building components, monitor locally their occupancy. In case a buffer gets nearly full, a throttle signal is fed back via the

dedicated throttle lines to the RS. The Readout Supervisor then throttles the trigger as long as the throttle signal is on. A timeout mechanism in the Readout Supervisor prevents the system from hanging. The timeout is detected by the Experiment Control System and would generally cause a reset to be sent to the appropriate hardware. Data congestion at the level of the Event Filter Farm is signalled via the Experiment Control System (ECS) to the Readout Supervisor via the ECS interface.

For monitoring and debugging, the Readout Supervisors – and also the Throttle switches and Throttle ORs (see 3.3.5) - have history buffers that log all changes on the throttle lines.

Trigger Generator

The RS also provides several means for auto-triggering to be used in conjunction with test and calibration runs. It incorporates two independent uniform pseudo-random generators to generate L0 and L1 triggers according to a Poisson distribution. The RS also has a unit running several finite state machines for periodic triggering, periodic triggering of a given number of consecutive bunch crossings (timing alignment), triggering at a programmable time after sending a command to fire a calibration pulse, and triggering at a given time on command via the ECS interface. The source of the trigger is encoded in the 3-bit L1 trigger qualifier.

Reset and Command Generator

The RS also has the task of transmitting various synchronous reset commands in order to prepare the front-end electronics for data taking or to recover from an error condition. For this purpose the RS has a unit running several finite state machines to transmit Bunch Counter Resets, L0 Event ID resets, L1 Event ID resets, L0 front-end electronics resets, L1 + L0 front-end electronics resets, etc. The RS can be programmed to send the commands regularly or solely on-command via the ECS interface.

Conflicts may occur when the different RS functions try to send several commands and a L1 trigger decision at the same time. A priority mechanism determines in which order they are sent. The Bunch Counter reset and the L0 Event ID reset can be sent at the same time and have highest priority. L1 trigger decisions have the lowest priority. However, it doesn't mean events are lost, only that the L1 trigger decision is postponed until the command has been sent. In case two commands are conflicting, the command with the higher priority is sent and the other is sent at the same bunch crossing number in the next LHC turn.

Status Counters

The RS keeps a set of counters that record its performance and the efficiency of the synchronous readout (dead-time etc.). In order to get a consistent picture of the status of the system, all counters are sampled simultaneously in temporary buffers waiting to be read out via the onboard ECS interface.

Readout Supervisor Front-End

The RS also incorporates a series of buffers, analogous to a normal front-end chain of a detector, to record local event information and provides the DAQ system with the data on an event-by-event basis. The "RS data block" contains the "true" bunch crossing ID and the event number, and is merged with the other event data fragments during the event-building.

ECS Interface

The RS is programmed, configured, controlled, and monitored via the ECS interface, a controller located on-board. Note that in order to change the trigger and control mode of the RS for testing, calibrating and debugging the design is such that no hardware intervention or reprogramming of FPGAs is required. All functionality is set up and activated via parameters.

4.1.3. Implementation of the Readout Supervisor

In order to conform to the requirements given above, the Readout Supervisor is based entirely on FPGAs [15]. The fast synchronous operation of the many parallel functions of the Readout Supervisor demands use of only the fastest FPGAs [19]. In particular, the handling of the L0 trigger is critical, as the internal L0 path should only contribute with three cycles of latency. In addition, there are many synchronous functions involved in the treatment of the L0 trigger.

The specifications of the Readout Supervisor have been simulated using the VisualHDL tool from Summit Inc. in a high level behavioural model together with a behavioural model of the LHC machine (clock, orbit signal, and bunch crossings), the trigger decision units, and the front-end electronics [20], [21]. The FPGA implementations have been simulated using the Max+Plus II software tool from Altera throughout the design phase, and have been crosschecked using the VHDL simulator of Cadence (Leapfrog). In order to simulate the full implementation of the Readout Supervisor, the behavioural model of the RS in the VisualHDL model mentioned above, has been replaced by the FPGA implementations at gate level including delays.

The first prototype of the Readout Supervisor is currently being tested. The first prototype is a minimal version containing in particular all critical logic that needs testing and all essential functionality. The logical design is shown schematically in Figure 17. The logical connections between the modules have a fully pipelined structure.

The entire Readout Supervisor is programmed, controlled, and monitored via an ECS interface. The FPGAs are programmed via a JTAG chain. All configuration, control, and monitoring of the functionality of the FPGAs are performed by means of read/write registers in the FPGAs via an I/O bus from the ECS interface.

4.1.4. TFC Switch

The TFC Switch is subject to two timing requirements [14]. The front-end electronics requires the timing of the TTC signal to be adjusted in order to sample the detector signals at the optimal point. Since the front-end electronics may receive the timing signal via different paths in the TTC Switch depending on which Readout Supervisor is used, it is crucial that all the internal paths of the TFC Switch from input to output have equal propagation delays. The maximum phase difference is required to be less than 100 ps. The TFC Switch should also contribute minimally to the jitter on the TTC signal. To satisfy these requirements, the switching logic has been implemented in ECLinPS and ECLinPS Lite technology from Motorola. All signal paths were routed such as to equalise the propagation delays.

Measurements performed on the first prototype of the TFC Switch [22] show that despite the equalisation of the lengths of switch paths, there are large skews between the outputs. These are mainly due to strongly varying propagation delays in the 16:1 multiplexers used. It will therefore be necessary to add adjustable delays at the outputs in order to calibrate the board. A high-speed buffered delay line from ELMEC technology with a 50 ps resolution and 40 steps and with very small temperature dependence (± 100ppm/°C) is suitable. The contribution to the total jitter was measured on the first TFC Switch prototype and was found to be approximately 50 ps [22].

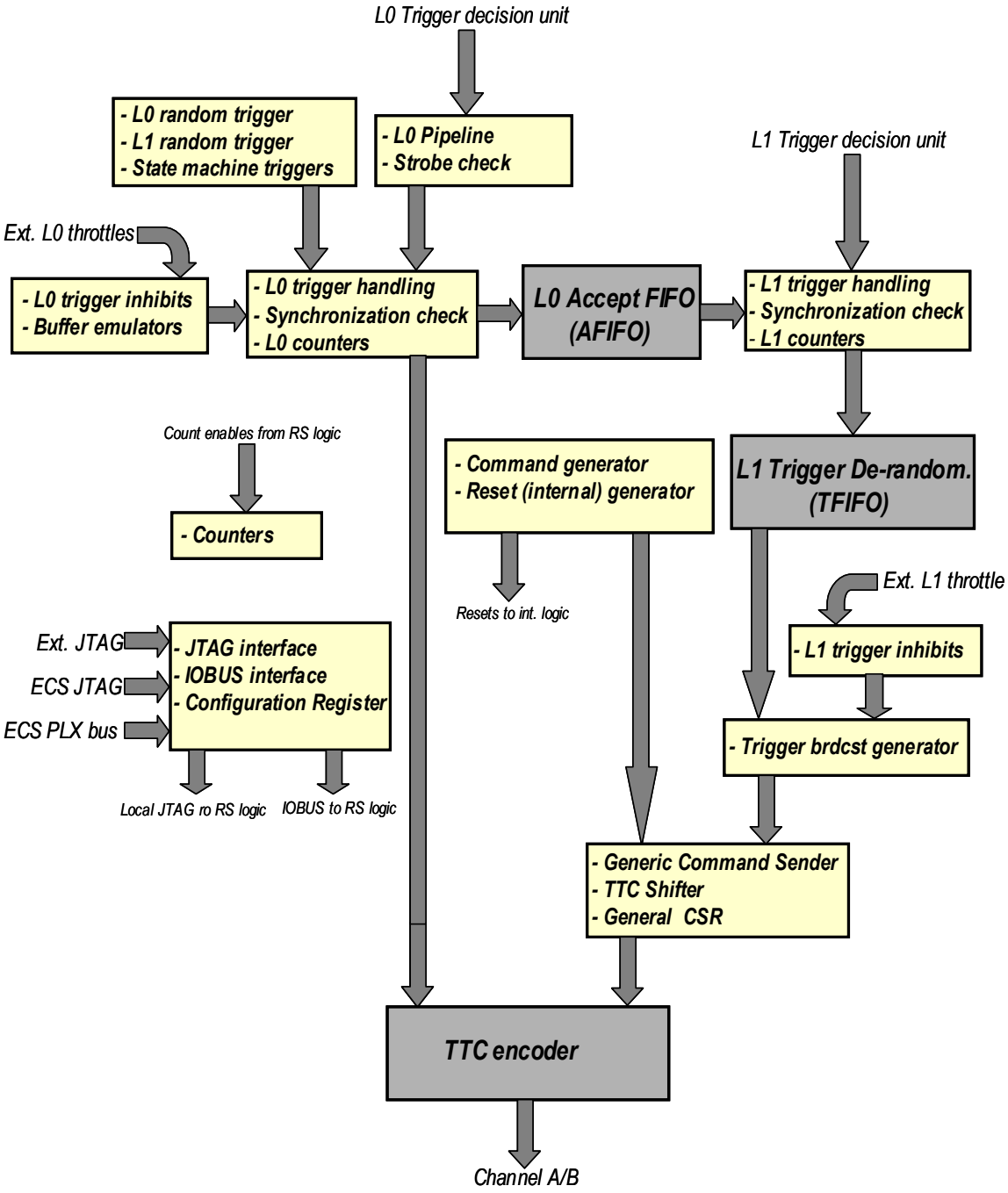

Figure 17 Block diagram of the first prototype of the Readout Supervisor. The shaded modules are implemented in discrete logic.

The first prototype of the TFC Switch has also been tested in the TTC test bench (Figure 15).

4.1.5. Throttle Switch and Throttle OR

The Throttle Switch and the Throttle ORs are not subject to strict timing requirements, and the switch and the OR logic have therefore been implemented using an FPGA. All programming, control, and monitoring are handled by the ECS interface located onboard.

In order to log the throttle history, a throttle signal triggers buffering of the current state of all the inputs, the current state of all the outputs, and the value of a 48-bit timestamp counter in a 32k deep FIFO. The timestamp counter runs at 10 MHz and is reset and readout by the ECS interface.

The first prototypes of the Throttle Switch and the Throttle OR will be built during the first half of 2002.

4.2 Data Link Technology and Link Protocols

LHCb has decided to adopt Gigabit Ethernet (GbE) as link technology from the output of the Level-1 electronics boards to the input of the Sub-Farm Controllers. The reasons for this is that it can be quite safely assumed that GbE will have a lifetime of more than ten years, because of its performance and its popularity in the LAN market. In addition, the price of GbE equipment is expected to drop significantly in the future. After the advent of 10Gigabit Ethernet, it is likely that GbE will eventually arrive on the desktop.

Since also the Readout Network will be GbE based, the choice of GbE for the other links is natural. It implies, however, that an S-Link card for the Level-1 Electronics based on GbE is designed and built[2]. This is underway within the Atlas collaboration [23] and we are collaborating in this effort.

On the links up to the input of the SFCs no high-level protocol will be used and only raw Ethernet frames will be sent from stage to stage for the following reasons:

- High-level protocols, such as TCP/IP, are quite complex and usually imply the usage of a processor to drive them. At the rates envisaged in LHCb, processors with sufficient performance would be too expensive to be deployed in the quantities needed (see Section 4.7.2). The use of UDP is not a solution. Since UDP doesn't guarantee the proper delivery of the data either, there is no gain.
- High-level protocols usually implement guaranteed delivery of the data. To do so, they have to buffer the data in the source and wait for acknowledgements from the destination that the data have arrived. This acknowledgement protocol introduces a non-deterministic behaviour in the buffer occupancy, which makes it difficult to estimate the buffer size required.
- Under normal circumstances, Ethernet is very reliable, especially on point-to-point connections.
- Potential loss of event fragments can be handled by a simple timeout mechanism in the SFC.

4.3 Front-End Multiplexing and Readout Units

Significant R&D has been done in the area of finding viable solutions for the implementation for the Front-End Multiplexers and the Readout Units. While the two functions are, a-priori, independent it turns out that their functionality is sufficiently similar that the same basic module can be used for both. The main difference between them results from the fact that the RUs have to interface to the readout network, and hence must respect the GbE flow control protocol, which is not true for the links between FEMs and RUs.

In this section we will describe the baseline solution for the front-end multiplexing and readout unit functionality, which is based on network processors. An FPGA-based approach, which was also studied and prototyped, is described in Appendix A. Both approaches have been prototyped, in order to demonstrate their viability.

The first section describes the features of the network processor solution, while the second gives the reasons for its adoption as the baseline solution.

[2] LHCb has decided to use S-Link as a standard interface between the Level-1 front-end electronics and the DAQ system.

4.3.1. Network Processor-Based FEM/RU

A network processor is a dedicated processor for network packet, i.e. frame handling. It provides fast memory and dedicated hardware support for frame analysis, address look-up, frame manipulation, check sum calculations, frame classification, and multi-casting. All these operations are driven by software, which runs in the network processor (NP) core. They were primarily designed as powerful and flexible front-ends for high-end network switches and switching routers. Because they are software driven they can easily be customised to various network protocols, or be adapted to new requirements or developments. They make it possible to create big and scalable switching frameworks, because they decentralise the address resolution and forwarding functions traditionally performed by a single, powerful control processor. Thus they enable switch manufacturers to construct large switches (up to 256 Gigabit ports and more), with dedicated software in a short time. Currently the "Gigabit" generation of network processors is on the market, while the next one will be able to handle 10 Gigabit speeds. These processors will be available in the course of 2002. More information on the history of network processors, their general features and future prospects can be found in [32]. Much more information is collected at the Network Processor Central website [33].

The unique features of network processors, namely being able to deal with incoming data packets at very high rates (up to 1 MHz and more), being equipped with large memory buffers and the fact that they are freely programmable, make them excellent candidates for the implementation of the front-end multiplexer and readout unit. Out of the several available network processors on the market, the IBM NP4GS3 was selected for study because of its excellent performance and the availability of software tools and documentation.

The basic components of the IBM NP4GS3 are shown schematically in Figure 18. As in all network processors, it supports two basic functions: packet forwarding and packet processing. Following the general philosophy of "output queuing" in large switches, the packet forwarding is performed in the input ("ingress") part of the processor, while the potentially more time-consuming packet processing is done in the output ("egress") part. Both functions are driven by software operating from the Embedded Processor Complex (EPC). In a typical industry application, several of these network processors would be connected via the Data Aligned Synchronous Link (DASL) to a switching fabric, to form a large multi-port router.

The basic technical problem for the RU/FEM application in LHCb is the concatenation of incoming packets from several sources belonging to the same event in the correct order. In doing so, data and (transport-) error blocks should be concatenated separately. The NP4GS3 has 4 Gigabit Ethernet ports, so using one chip allows up to 3 to 1 packet merging. Adding a second network processor, which can be connected directly without the need for an additional switching fabric, will allow multiplexing up to 7 to 1. The packet merging can in principle be done in both the "packet forwarding" and the "packet processing" stage of the NP. For the RU/FEM application, we have implemented the packet merging software, which runs in the EPC, in the output stage, because of the large amount of buffer memory available there. The 64 MB of external RAM leaves sufficient time to buffer packets and hence to cope with large spreads in arrival times.

The performance of packet merging is limited only by the access time to the external memory. For even faster packet merging, a functionally equivalent algorithm has been developed, which is operating on the input (or ingress) stage, normally used for fast packet forwarding. It profits from the fact that the memory at this stage is on-chip and hence access is very fast. The disadvantage is however that there are only 128 kB of memory, which would make it interesting mainly in a situation where the packets sizes are very small (a few tens of bytes) and the arrival times of packets belonging to the same event do not vary too much. More details for both approaches to packet merging can be found in[34].

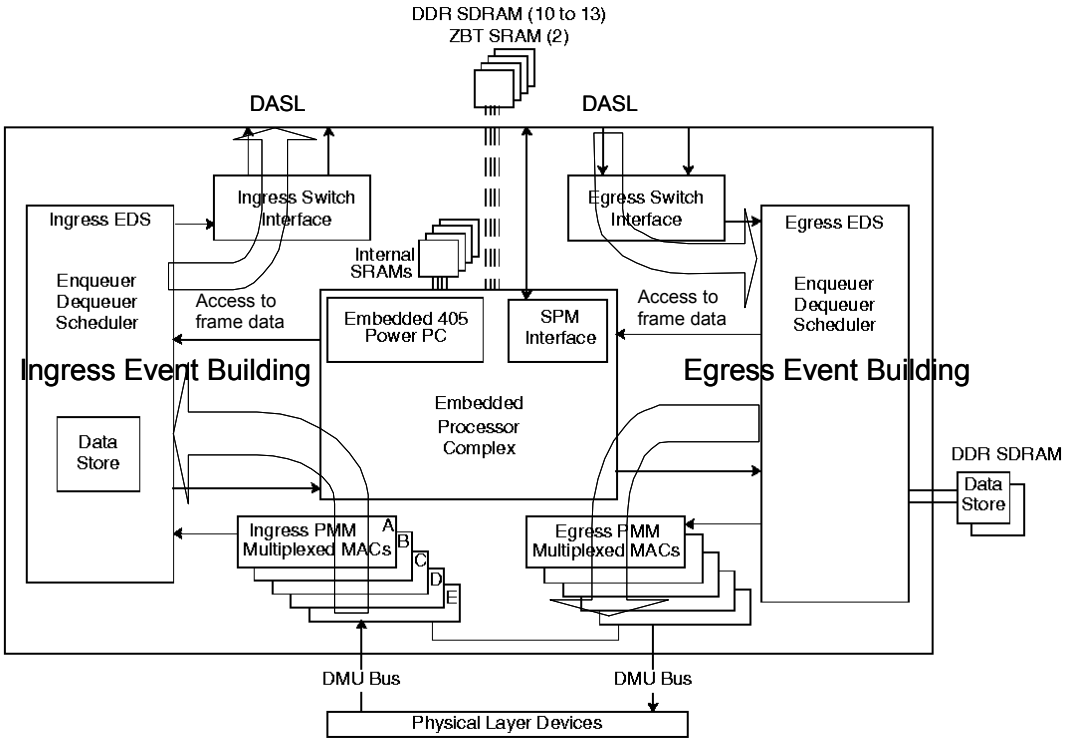

Figure 18 Main components of the NP4GS3 together with an indication of the standard data-flow paths.

In order to use the NP4GS3 as the basis of a FEM/RU module, it will be implemented on a standard LHCb electronics board. Figure 19 shows a block diagram of a FEM/RU module with two network processors mounted on mezzanine cards. The usage of mezzanine cards leaves the flexibility to use only as many NP4GS3s as are actually needed to provide the required multiplexing factor, and hence optimising the cost of the whole FEM/RU system. It also facilitates the overall board design and makes it possible to share common infrastructure, like the ECS interface, between the two NPs. The details of the mezzanine are shown in Figure 20. It is worth noting that when running standard routing software instead of our packet merging code such a module is a full functional 8 Gigabit Ethernet port switch.

The performance of the packet merging was measured using simulator software from IBM [36], which provides cycle-precise timing[3], and dedicated test case generators. In Figure 21, the performance of packet merging is given by plotting the maximum acceptable rate of incoming fragments as a function of the average input fragment size. The results show that for typical average input fragment sizes (200 B to 500 B) the RU can function at rates of over 150 kHz. This has to be compared with the nominal L1 trigger rate of 40 kHz.

Simulation thus convincingly showed that the performance is more than sufficient for the FEM/RU application. In fact, for almost any average size of the incoming fragments, the packet merging is faster than merged packets can be sent out over a single Gigabit Ethernet link (120 MB/s). Thus the limitation in performance is governed by the maximum permissible load on an output link, which has been set to 80 MB/s, as discussed in Section 4.7.2.

[3] This is a RISC architecture, so all instructions of the pico-engine can in principle be executed in one cycle.

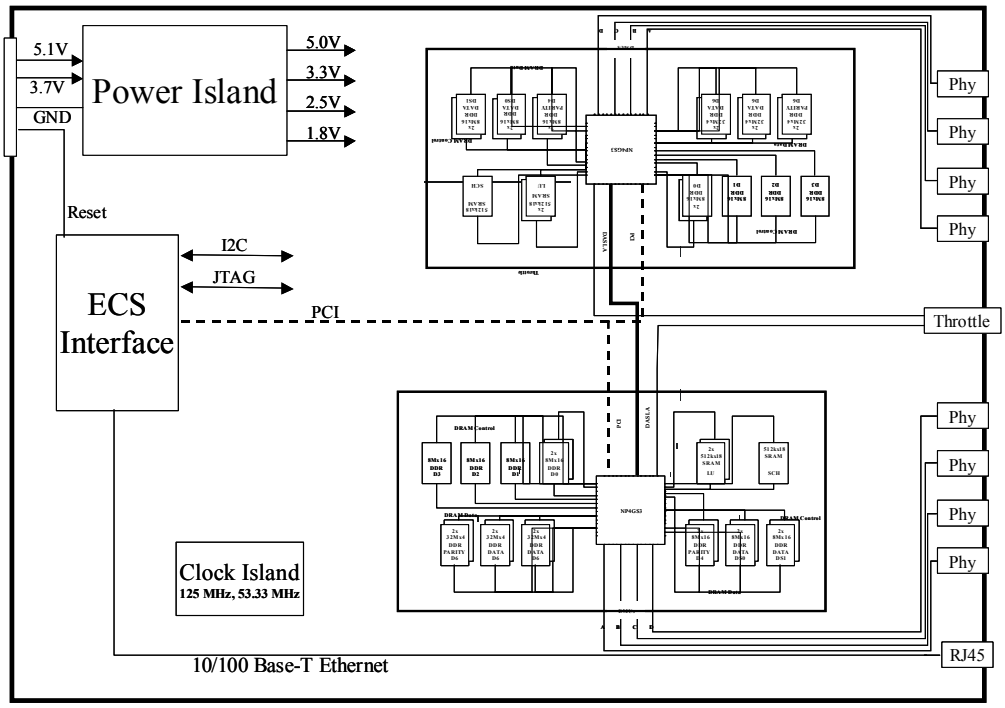

Figure 19 Block diagram of a FEM/RU module with two basic mezzanine cards.

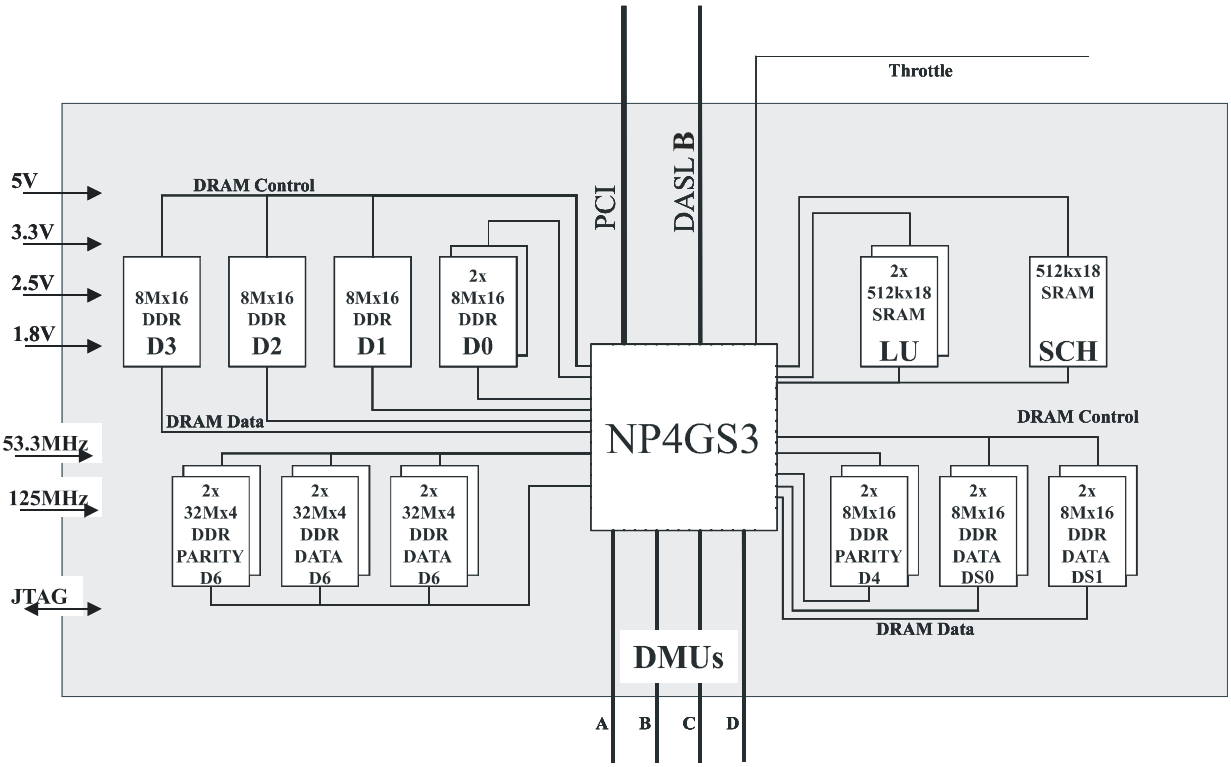

Figure 20 Block diagram of a NP mezzanine card.

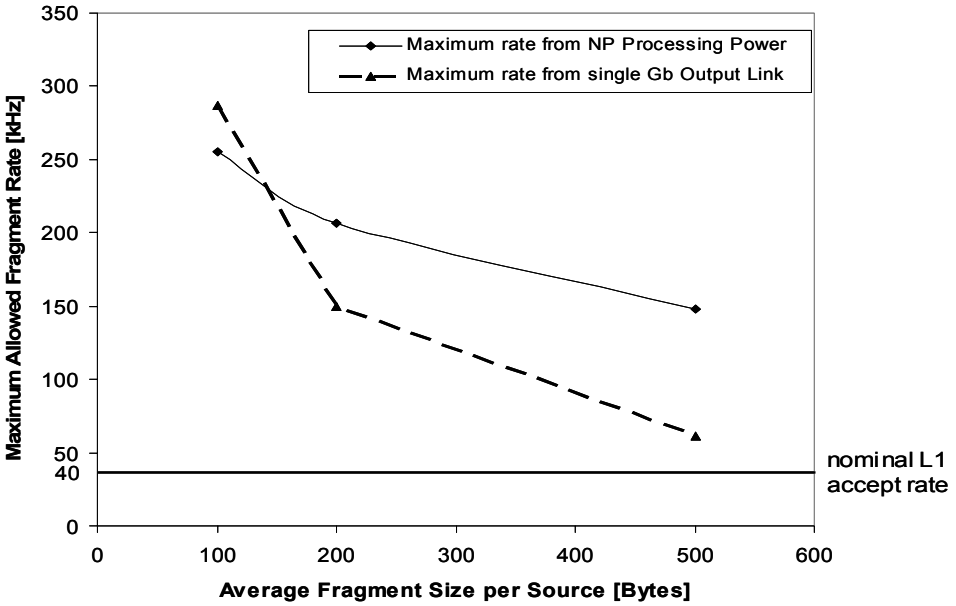

Figure 21 Performance of the NP based event-building.

Only for very small fragments of a few tens of bytes the limit in performance comes from the processing power available for packet merging. This holds true for any multiplexing factor from two up to the maximum seven.

A hardware evaluation kit from IBM (IBM PowerNP Reference Platform [37]) has been used to verify the simulation results. This platform consists of 2 Network Processors, each with four Gigabit Ethernet ports and a control processor connected to the NPs. It is functionally fully equivalent to the readout unit to be used in LHCb.

Using a dedicated test set-up with the programmable NICs described in Section B.3 operating as data sources and sinks, the results of the simulation could be confirmed with very good accuracy. The details of the test set-up, the measurement procedure and more detailed results can be found in [34].

4.3.2. Baseline Implementation

The FPGA-based and NP-based implementations were brought to a state of prototyping that proved the viability of the concept. After an internal review it was decided to adopt the Network Processor-based approach as baseline solution and keep the FPGA-based solution as a backup. The main arguments for this decision are as follows:

- The NP-based readout unit is more flexible and more versatile, because its functionality is to a very large part governed by software only. This fits also well with the core competences of the LHCb online group.
- As a switch- building block, it seems to be competitive with ‘monolithic’ switches and has the great advantage to be software customisable to the needs of the LHCb event-building protocol.
- It has a very elaborate development environment that allows rapid simulation and testing of new versions of the software.
- The cost of the NP-based solution is approximately the same as that of the FPGA-based module.

This approach was also endorsed by an extensive review of the whole data-flow system with the participation of external experts.

4.4 Event Building

The event-building sub-system consists of

- the output stage of the RUs
- the input stage of the SFCs
- the Readout Network

As already mentioned in Section 4.2, Gigabit Ethernet (GbE) has been selected as link technology from the output of the Level-1 front-end electronics boards to the input of the Sub-farm Controllers. We have also investigated other technologies for implementing the Readout Network, such as Myrinet (see Section B.1). ATM has been studied as part of CERN's R&D programme (RD31, [35]). This technology has, however, not gained the market acceptance expected, due to the advent of 100 Mb Ethernet.

With the choice of Network Processors for the implementation of the RUs, the output stage of the RUs consists of the built-in output of the Network Processor, and therefore no special Network Interface Card (NIC) is required. The output stage of the RUs must implement the function of destination assignment. Flow control or traffic shaping may be required if the size of the internal buffers in the readout network is not sufficient. Network processors implement support for traffic shaping in hardware, hence with a NP-based RU this should be easily achievable, if it is required.

The input stage of the SFCs must link the event fragments that arrive in an arbitrary order, sort out the fragments belonging to different events, detect and signal missing data. No flow control action is required at this stage, as long as the SFC is operational and the buffer space sufficient. Several possibilities exist for the implementation of the SFC input stage. The preferred solution, for the time being, is the use of "Smart NICs" ([60] and [61] and Section B.3). Should the market trend run against the existence of Smart NICs within the time scale of our decision (beginning 2003), alternatives exist either using a NP-based solution or performing the event-building in the CPU of the SFC.

The remaining part of this section will be devoted to a discussion of the implementation issues of the readout network. An introduction to the general concepts of load, switching networks and data flow control are presented in Section B.4 of Appendix B .

The size of the readout network results from a compromise to satisfy several constraints:

- One port link bandwidth is 1 Gb/s.
- The load on the network should be significantly lower than 100% in order to keep the probability of data loss sufficiently low.
- The cost of the network grows like NlogN with the number of ports, the cost of port devices like 2N.
- The requirements for partitioning forbid the aggregation of data from different detectors into the same link (event fragment).

The optimal solution requires 60 RUs (Appendix B , Section B.4). The number of SFCs must be at least 60 but may be larger if needed. The fragment size is around 2 kB, which corresponds to a load of 66%. The total event size is 100 kB. Those are average values, actual events will vary in size, both for the fragments and for the full event size.

The implementation of the readout network can be based on commercial products: large switches offering up to 120 GbE ports are available now. Another possibility would be to use the 8 ports provided by the twin mezzanine boards foreseen for the implementation of the RUs. We now discuss both solutions.

4.4.1. Solutions Based on Commercial Switches

A switch offering 120 ports would be just sufficient to implement the required readout network.

A difficulty when evaluating solutions based on commercial switches is the secrecy maintained by the manufacturers on the details of the architecture and functioning of their switch. Some characteristics can be inferred from direct traffic measurements on the switch but we will probably lack the detailed information necessary to build a trustworthy model of the network. Switches from different vendors are likely to have different architectures and/or switching strategies.

An example of an architecture is given in Figure 22 that presents a schematic layout of the Fast Iron Switch proposed by the company Foundry Networks [58]. Presently, the highest bandwidth available for the backplane is 240 Gbps, allowing up to 15 modules of 8 ports to be connected, namely 120 ports in total.

Measurements that we have performed on the switch [59] show that the data packets submitted to a port of the switch are segmented in fixed length packets of 64 bytes (including possible internal network protocol). This is an indication that time division multiplexing is in operation on the backplane. The local switches probably implement central queuing. Our measurements indicate that the shared memory size of each local switch is 2 MB, with a limit of 1024 frames (see Section B.4 for explanations of time division, central queuing, etc).

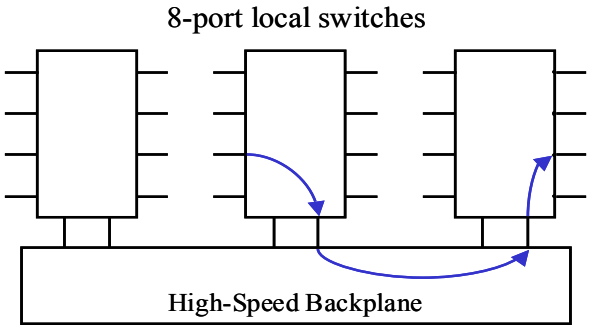

Figure 22 Architecture of the Foundry Fast Iron switch.

To implement the event-builder, one single box is sufficient. To upgrade to higher numbers of ports, more than one box is required.

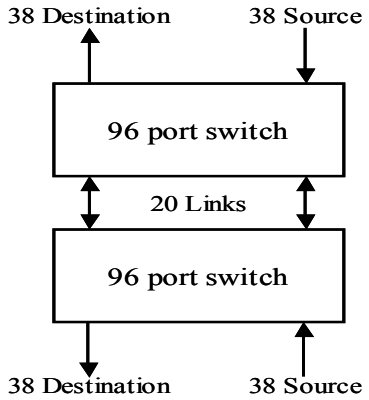

Figure 23 Interconnection of 2 switches, 96 ports each, providing a 76X76 event-building network.

A possibility is to interconnect 2 crates not completely equipped in order to increment the event-builder size in steps. Figure 23 shows an implementation based on the Foundry Switch with 12 modules of 8 ports per crate, implementing a 76x76 event-building network.

Note that sources and destinations must be mixed on each box in order to use both directions of the interconnection links.

4.4.2. Solutions Based on Network Processors

The Network Processor boards foreseen for the implementation of RUs can be programmed to perform switching between the 8 GbE ports with central queuing (B.4.2). The behaviour of a network based on those processors can be simulated. We have used the following assumptions to produce the results shown in this section:

- The fragment size has an inverse exponential distribution with an average of 2 Kbytes. It is limited to a minimum of 700 Bytes and a maximum of 9 Kbytes.
- The inter event arrival time is Poissonian with a frequency depending on the load generated.
- The destination assignment is random, instead of sequential, and is generated at the start of run.

As a first example of implementation, let us consider a Banyan network of 4x4 switching modules that provide 64 input and 64 output ports (Figure 24).

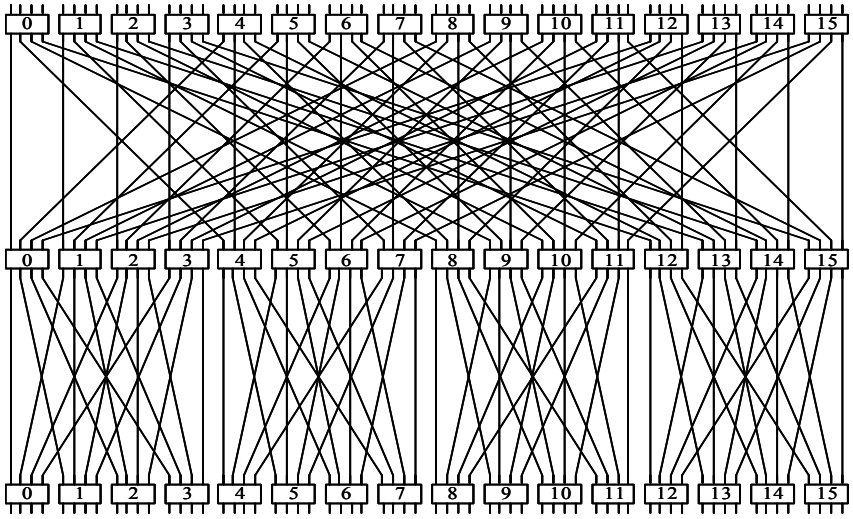

Figure 24 A 64x64 event-builder based on 4x4 switching modules inter-connected in a Banyan topology.

This network requires 3 stages of 16 modules and 256 inter-connection links (128 bi-directional). The links are used in one direction only for data transfer.

Results from simulation are shown in Figure 25. The maximum buffer occupancy is shown as a function of the load on the network. The working point corresponding to 40 kHz (66% load) is in a very stable domain for loads up to 90% at least.

The main drawback of a Banyan network is the unidirectionality of the data flow such that half of the installed bandwidth is unused while the other half can be loaded up to 100%. Another drawback is the poor scalability for numbers of ports that are not powers of the basic module size (or one of its factors). For example, the next well-balanced network above 64x64 is 128x128.

Another scheme is proposed which mixes sources and destinations in terminal modules and uses the interconnections in both directions. For that purpose, full interconnectivity of end ports must be implemented. Figure 26a shows one basic interconnection pattern where 3 modules carry the sources and destinations and 3 modules ensure the full connectivity of the network.

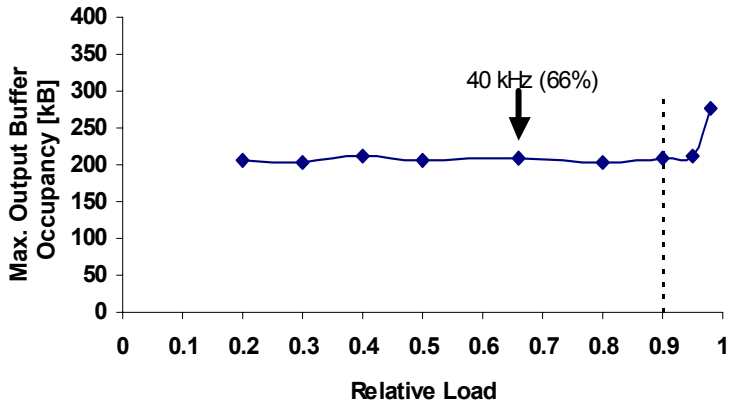

Figure 25 Simulation results for the 64 x 64 Banyan network based on 8 port modules for event-builder traffic.

A maximum of 16 basic patterns can be inter-connected, implementing up to 112 sources and 128 destinations. Figure 26b shows 9 such basic patterns that implement 63 sources and up to 72 destinations.

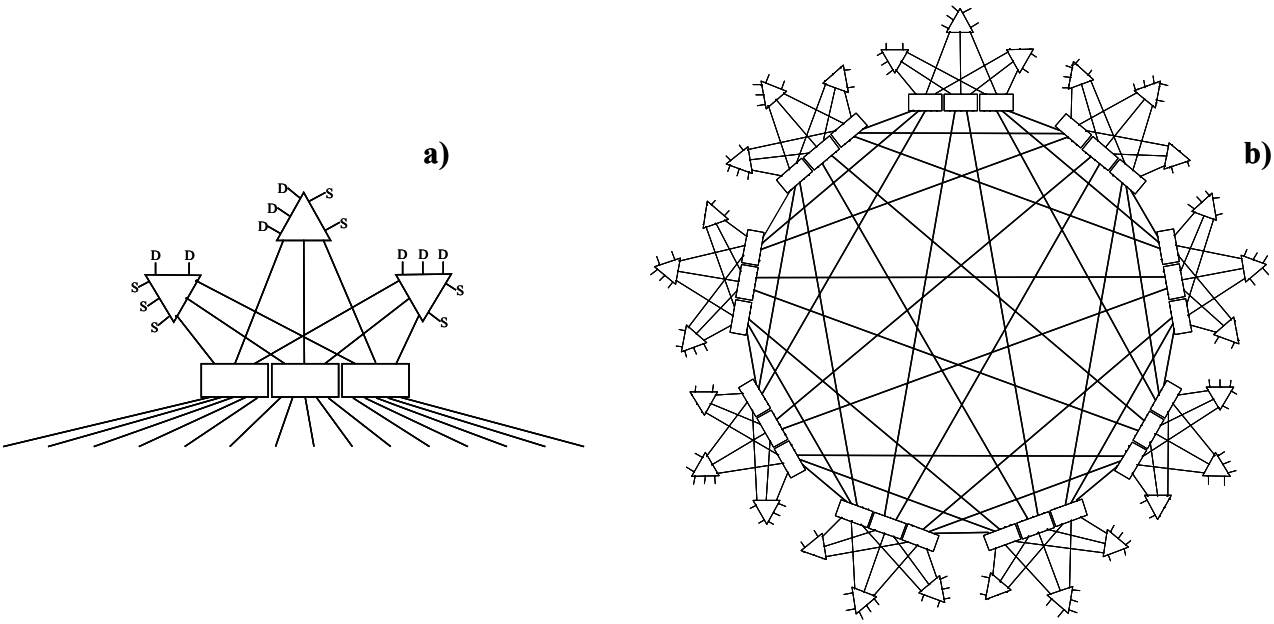

Figure 26 a) Basic building block for a readout network. b) 9 basic blocks connected to a readout network sufficient for LHCb.

The maximum load on the inter-connection links does not exceed 80% under the hypothesis that the sources would load their input link at 100%. The simulation results for this network topology are shown in Figure 27 where the maximum of the shared output buffer occupancies over all modules is plotted as a function of the load. One can see that the network is stable up to a load of at least 90%, the working point (40 kHz) being at 66% load. Compared to the Banyan topology, this layout only offers lower output buffer occupancy, but the number of components is higher (54 instead of 48).

The advantages of this type of interconnection do not show up for such a small network. It has, however, the advantage to scale more smoothly than the Banyan network. For larger configurations, it requires fewer components. For example, in its largest configuration (112 x 128) it requires 96 modules of 8 ports and 528 links, to be compared to 128 modules and 768 links for a Banyan network of 128 x 128 ports.

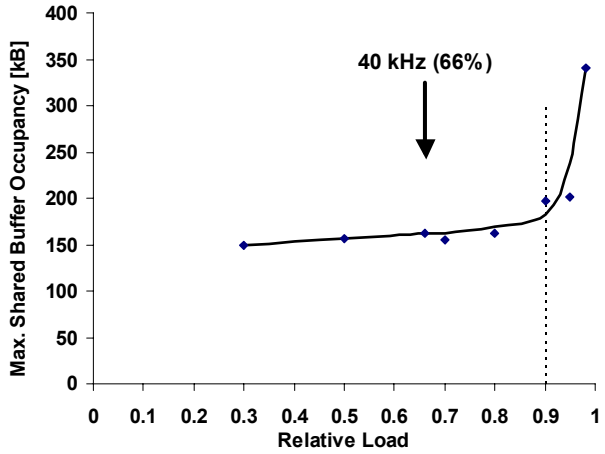

Figure 27 Simulation results for the 63x72 event-builder using 8 port modules and bi-directionality in the inter-connections

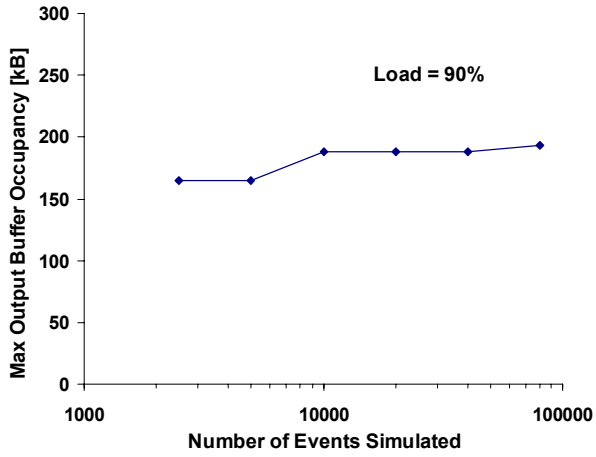

Figure 28 Maximum Buffer occupancy as a function of the number of events simulated for a load of 90%

The maximum occupancy has been determined as a function of the number of events simulated. Figure 28 shows this dependency for a load of 90%. Based on this result, one can estimate the probability to lose 1 cell in an event to be less than 10^{-8} (~ once per hour) for an output buffer of 1 Mbyte at 90% load.

4.4.3. Effect of large events

The simulations presented so far have been made assuming "normal" events with an average size of 100 kB (fragments of 2 kB). Under normal running conditions, it is expected that a small fraction of the events will have a much larger size, as they will carry calibration data. For the time being, we have no information on the characteristics of those large events. So, their effect has been evaluated for a wide range of values, both for the event size and for the event frequency.

The nice feature of output queuing switches is that they carry quickly the data to the output ports, storing them momentarily in shared output memory when contention occurs. This allows sustaining high loads through the network. In the case of event-building traffic however, this feature may lead to high buffer occupancies when the event size is higher than some value. For the "normal traffic", we are far below this limit.

Figure 29 shows the effect of events 10 times larger than normal events that are inter-mixed with the normal traffic. For a frequency low enough (depending on the size of those big events), the maximum buffer occupancy is independent of the frequency. Figure 30 shows the maximum occupancy as a function of the size of those large events (up to 100 times the size of a normal events). If the output buffer is large enough, as it is the case for the Network Processor based switch, no special care is needed, however in commercial switches the output buffer size is not sufficient (~2MB) and traffic shaping is required.

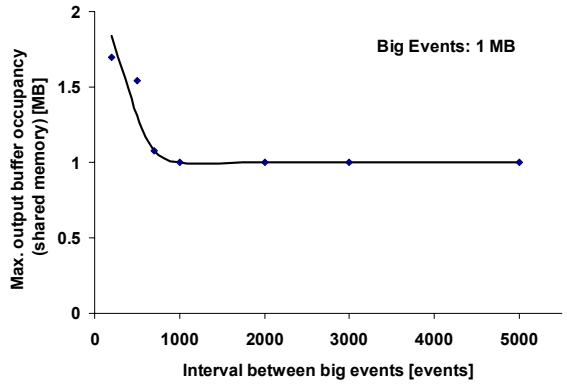

Figure 29 Maximum output buffer occupancy for 1 MB events superimposed to normal traffic as a function of the interval (in events) of the large events.

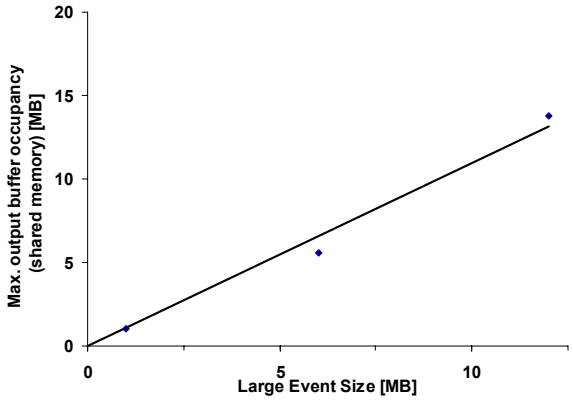

Figure 30 Maximum output buffer occupancy for large events superimposed to normal traffic, as a function of the large event size.

4.4.4. Implementation of the Readout Network

In the previous sections, the criteria and the different options for implementing a Readout Network satisfying LHCb's bandwidth requirements[1] were described. The crucial parameter that will determine which of the options to use is the output buffer size available. While there should not be a problem for events of normal size (see Figure 27 and Figure 28), which are very modest, the situation is much less clear for large events. These events will occur and it is unacceptable if all those events were tagged as error-prone.

Clearly, we would prefer to use commercial switches to implement the Readout Network, mainly for cost and convenience reasons. It is, however, not obvious that these switches will provide the buffering capabilities necessary for our purpose, since for their original designation, buffers of the size we require are not needed and the memory installed is a cost factor. Hence, currently we are sure that an implementation with Network Processor-based Readout Units is a viable solution. We will watch the switch market very carefully in the future and will also do tests with different commercial switches, and re-assess the situation in mid 2003 when we will have to decide on the implementation.

4.5 Event Filter Farm

The Event Filter Farm will consist of many CPUs organised into ~60 sub-farms, interfaced to the Readout Network via the Sub-Farm Controllers (SFC). A sub-farm is a collection of PCs, which are fed by the SFC with events. When an event is selected, its raw and reconstructed data will be transferred back via the SFC to the Storage Controller for final archival to tape.

A sketch of this system is shown in Figure 31. The SFC is connected to the Sub-farm nodes via an aggregation switch (data switch). The SFC and all nodes of a sub-farm are connected via another aggregation switch (controls switch) to a Controls PC and to the controls network. The SFC will have two Gigabit Ethernet interfaces, one towards the Readout Network and one towards the switch connecting to the farm nodes. In our base-line solution, the interface to the Readout Network will be a "Smart NIC", discussed in Section B.3 And in [61], which performs the final event-building on the fly. In addition, it will have a separate 100 Mbit, i.e. Fast Ethernet, interface to the controls

[1] The bandwidth requirements almost directly transform into requirements on the number of ports of the switch. This is discussed in detail in Section 4.7.2.

network. Likewise all farm nodes will have two network interfaces, one connecting to the data switch and one to the controls network. The controls network is ultimately connected to a controls PC responsible for configuring, monitoring and also booting the farm nodes as well as the SFC. The final number of sub-farms controlled by one controls PC will be determined later, based on a detailed understanding of the performance requirements.

The sub-farm aggregation switches are already commodity items today. They are commonly called "connectivity switches" and provide a non-blocking fan-out from one or two Gigabit "up-links" to several (~ 20) 100 Mbit Ethernet ports. With ~900 farm-nodes in total the data rate into each node will be 4 MB/s so 100 Mbit will be sufficient.

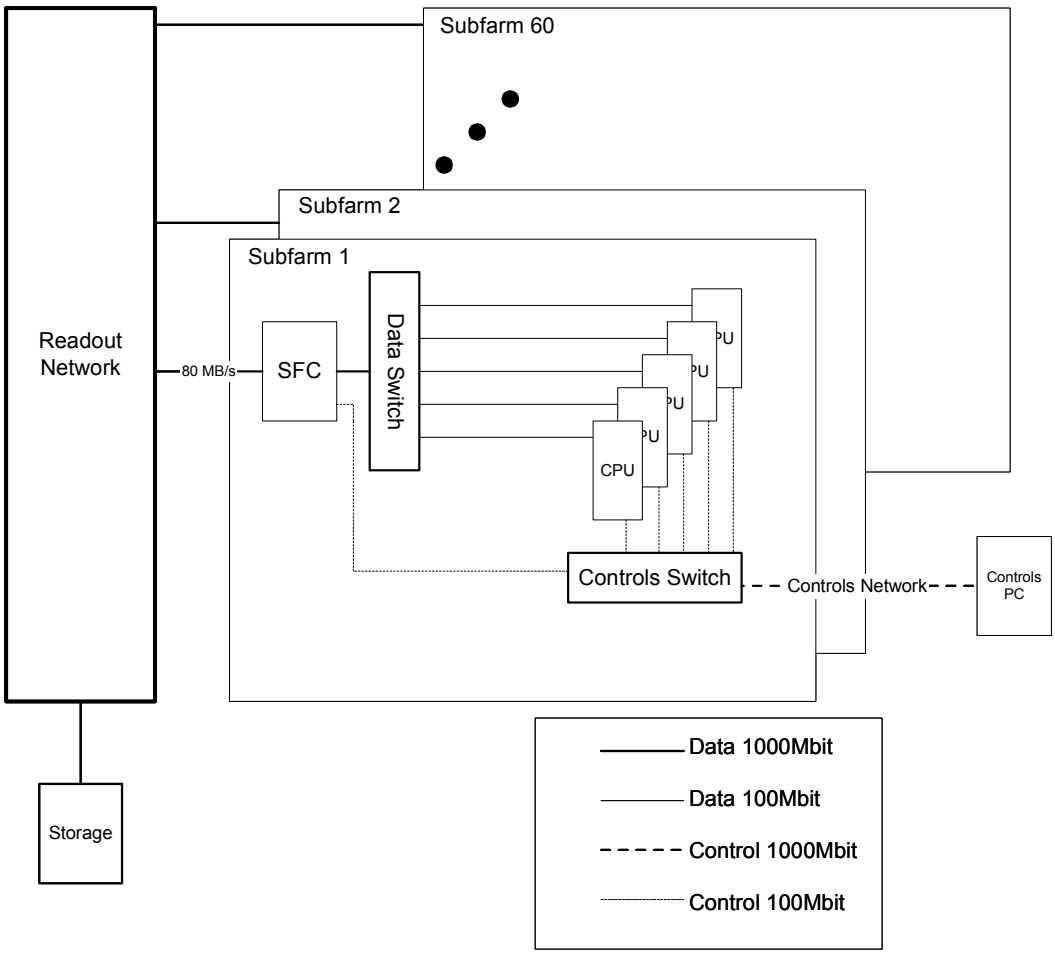

Figure 31 Schematic view of a sub-farm.

The SFC will be a PC optimised for I/O performance. A sketch of the internal architecture is shown in Figure 32. On the right-hand side the three network interfaces are shown. Also indicated are the average expected data rates. The numbers for the local bridges are for a PCI architecture and may differ for other, faster, local buses like Infinibus. It will need a large amount of memory, 2 GB RAM or more. The CPU requirements depend on the way the final event-building is done. We are considering two options. The baseline option is to use "Smart NICs", which are part of the SFCs. The CPU requirements on the SFC will then be rather modest. In the case that "Smart NICs" are not available, the final event-building including stripping of headers, has to be done by brute-force memory copying in the SFC. This requires more CPU power in the SFC.

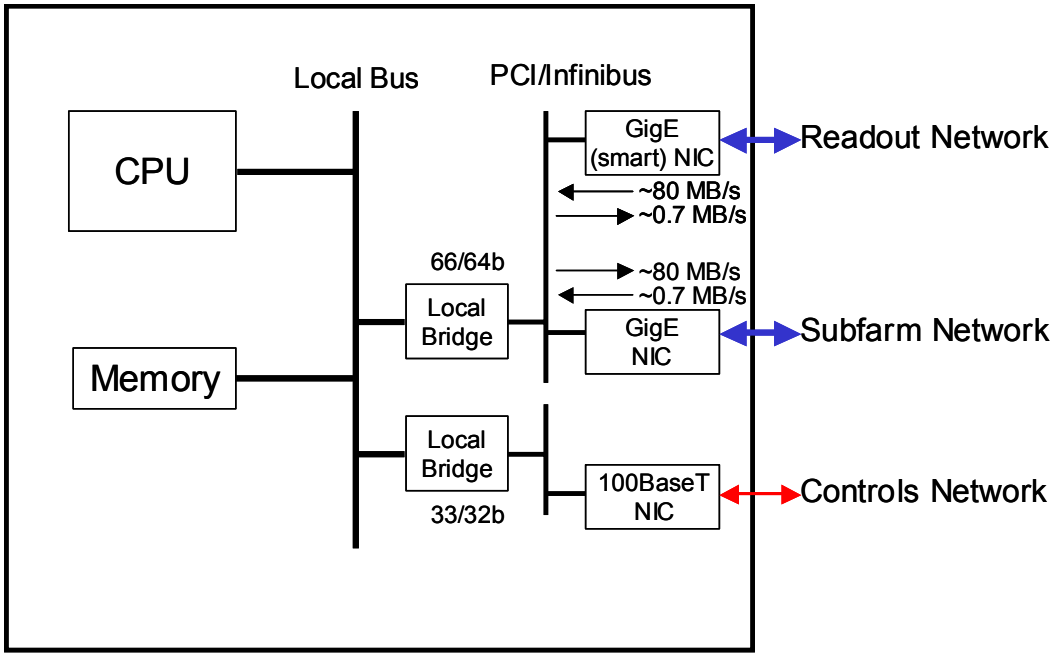

Figure 32 Sketch of the architecture of a SFC.

The farm nodes will be commercial PCs. In particular, we will follow closely any common purchasing strategy adopted by CERN and/or the LHC experiments to leverage on common effort and quantity rebates. More details on possible farm node implementations can be found in [62].

Possible specific implementations of the farm-node include:

- Standard PC boxes. They are widely available, but are not very effective in terms of floor space.
- Rack-mounted (1U) servers are available from many companies, but are usually more expensive in terms of MIPS/CHF.
- Micro-server blades are a relatively new development rapidly catching on in the market [63]. These offer the highest CPU density. They are usually operated using low power consumption CPUs, which are less performing. Whether they will be a cost effective solution depends very much on the development of the market.

The specific configuration of a commercial device might not be optimal for our purposes, since we prefer to operate the nodes in disk-less mode. The trade-off between paying extra for a custom configuration and paying for unused components has to be evaluated.

The criteria shown in Table 5 will be used to prepare the arguments for making the final decision. The figures assume 900 nodes and today's prices.

Table 5 CRITERIA FOR THE DECISION FOR THE IMPLEMENTATION OF THE EVENT FILTER FARM OF 900 NODES

	Cost [kCHF]	Space sqm	Cooling [kW]	Vendor dependence
Mini-tower	1260	45	180	No
1U server	1530	15	180	No
Micro-blade	1440	2.25	63	Yes

The cost in Table 5 is for the raw CPU only. For micro-blades it should be kept in mind that they usually don't use the latest CPUs and so the number can be scaled up by ~ 20%.

Allowing for clearance and assuming 1.8 m (19" rack size) high racking, densities are 40/m^2 for mini-towers, 120/m^2 for 1U systems and over 800/m^2 for micro-blades. These are node densities, so

the CPU density can be doubled by using dual CPU systems, which are increasingly becoming available.

For a thousand-node system, the total amount of power to be cooled away will be approximately 200 kW for mini-towers and 1U systems, and 70 kW for low power blade systems. The required cooling power per m^2 is quite big and solutions must be found, which will possibly require additional hardware such as closed shelves with separate heat exchangers.

While there are fewer vendors for 1U systems than for standard PCs, there is still a number of vendors that offer them. In addition, they are exchangeable, because they have the same-form factor, although a slightly different layout of the front-panel. Micro-blades come in special crates, and imply using a single vendor, at least for complete crates.

4.6 Experiment Control System

LHCb's Experiment Control System (ECS) is in charge of the control and monitoring of all experimental equipment. As such, it has to provide interfaces to all types of devices in the experiment and a framework for the integration of these various devices into a coherent complete system. In the following paragraphs, we will first describe the control framework and then the interfaces proposed for the different control areas.

4.6.1. Control Framework and Tools

The LHCb Control Framework will be a specialisation of the JCOP framework. It will provide for the integration of the various components (devices) in a coherent and uniform manner. JCOP defines the framework as:

> *"An integrated set of guidelines and software tools used by detector developers to realize their specific control system application. The framework will include, as far as possible all templates, standard elements and functions required to achieve a homogeneous control system and to reduce the development effort as much as possible for the developers".*

The control framework was developed following the specifications provided by the JCOP Architecture Working Group (AWG) [18]. The framework is based on the PVSS II SCADA system and addresses the following issues:

Device Orientation

Device orientation is a high-level abstraction allowing the description of complex equipment in simple terms. The device description contains all the data and the high level commands that are needed in order to operate the equipment, even though the equipment could be composed of many channels. In comparison, tag-based systems would describe and operate channels individually and independently. The framework will provide a device-oriented interface to the different hardware components. PVSS II allows device-oriented modelling. Support of this feature is considered crucial and was one of the main criteria for choosing this product. PVSS II has the concept of "data point" types, which can be complex data structures, from which "data points" are instantiated. The protocol "drivers" used should also allow for this access mechanism. This is true for the OPC protocol, which is widely used in industry as a standard interface to commercial components, and for DIM (Distributed Information Management) [38], which is the recommended interface for components not providing OPC servers.

Hierarchical Control

The framework offers tools to implement a hierarchical control system [18]. The hierarchical control tree is composed of two types of nodes: “Device Units” which are capable of monitoring and controlling the equipment to which they correspond and "Control Units" which can model and control the sub-tree below them. This is illustrated in Figure 33. In this hierarchy "commands" flow down and "status and alarm information" flow up.

Control units are typically implemented using Finite State Machines (FSM), which is a technique for modelling the behaviour of a component using the states that it can occupy and the transitions that can take place between those states. PVSS II does not provide for FSM modelling and therefore another tool – SMI++ [39] has been integrated with PVSS for this purpose. SMI++ allows for the design and implementation of hierarchies of Finite State Machines working in parallel. SMI++ also provides for rule-based automation and error-recovery.

Each component in the tree (Device or Control Unit) provides information and can receive commands. From the point of view of hierarchical control, the interface between components and between components and operators is "state" flowing up and "command" flowing down, i.e. a “state/command” interface.

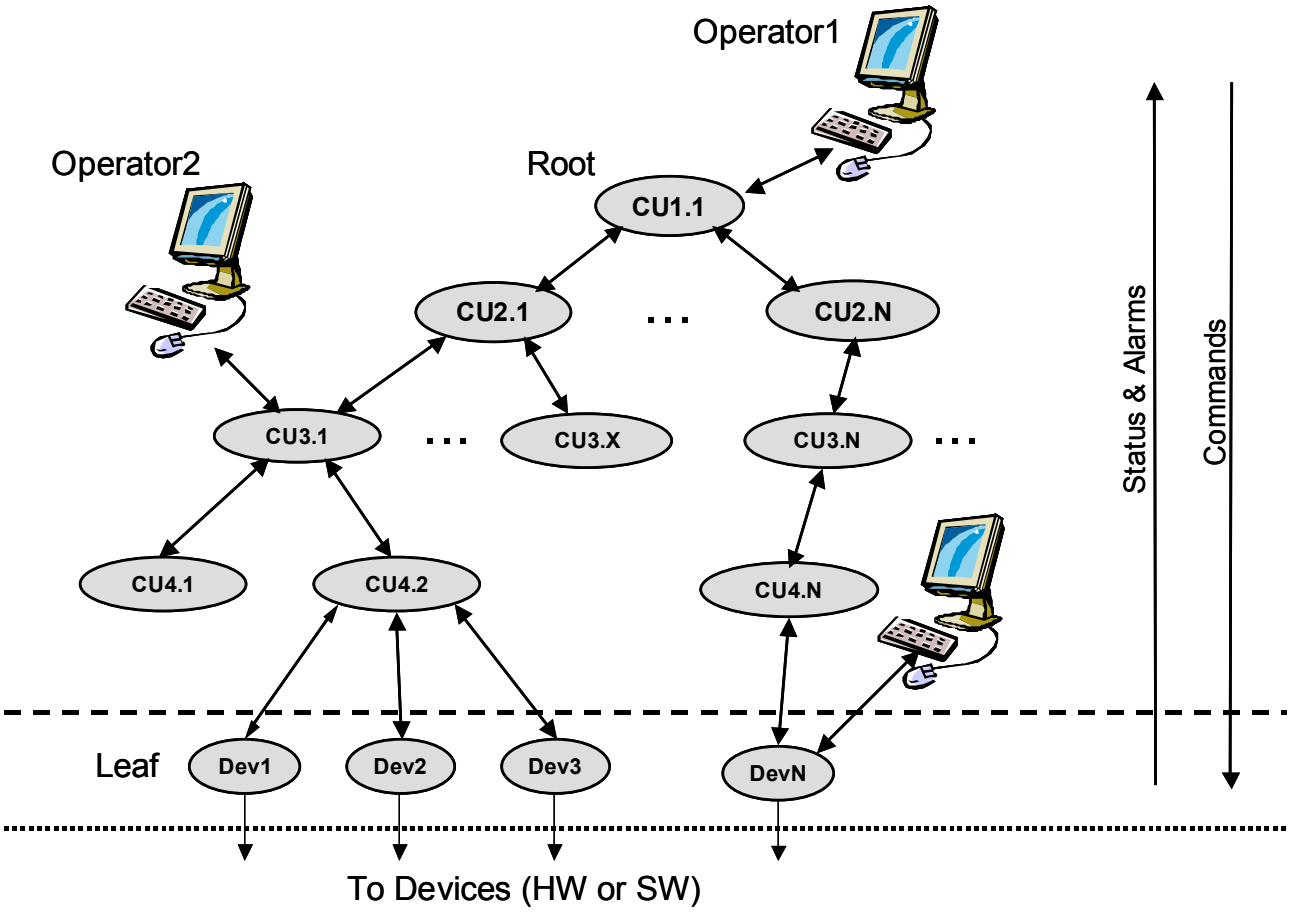

Figure 33 Hierarchical Control Architecture

Control Units are logical decision units. The logic behaviour of a Control Unit is expressed in terms of Finite State Machines. State transitions can be triggered by:

- Command reception (either from its parent or from an operator)
- State changes of its “children”

State transitions cause the evaluation of logical conditions and possibly commands to be sent to the "children". Control units can act on their "children" on request from their parents or they can behave autonomously since they can take decisions internally based on the states of their "children".

This mechanism can be used to propagate actions down the tree, to automate operations and to recover from error situations. An expanded view of a Control Unit showing the functional components and the interaction with the external world can be seen in Figure 34.

Device Units implement the interface with the lower level components. They always represent a "leaf" in the control hierarchy tree, i.e. they have no children. They do not implement logic behaviour. They receive:

- Commands and act on the device
- Device data and translate it into States.

The detailed view of a Device Unit can be seen in Figure 35.

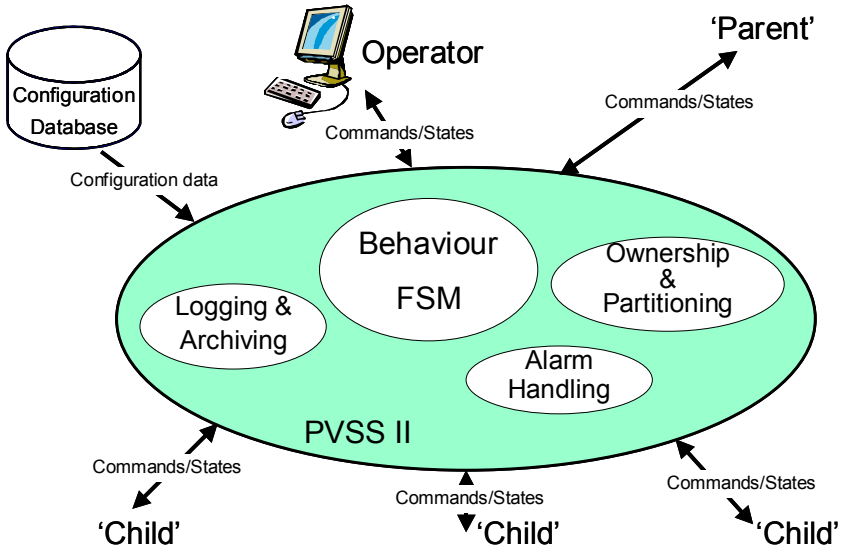

Figure 34 Schematic representation of a Control Unit.

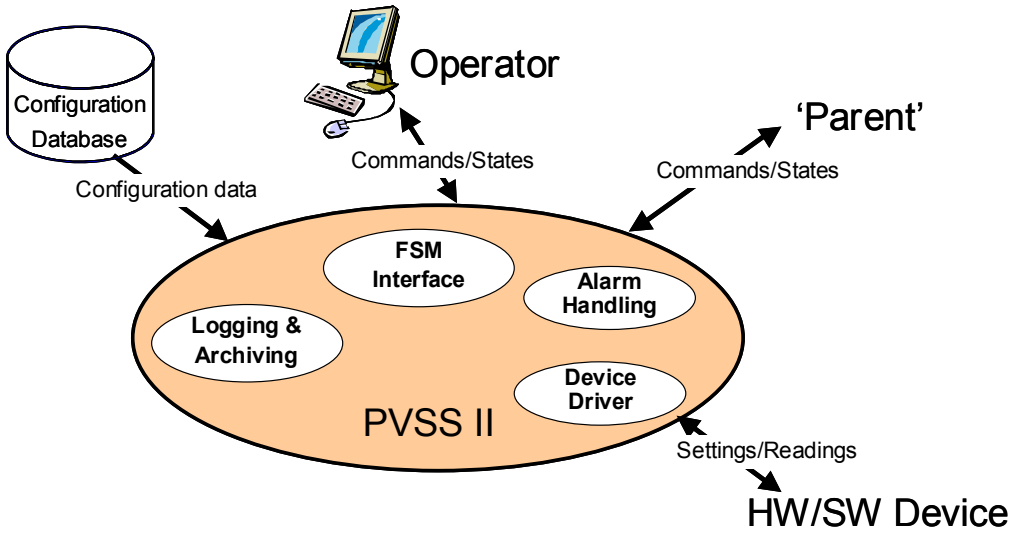

Figure 35 Schematic representation of a Device Unit.

Partitioning

Partitioning is the capability of monitoring and/or controlling a part of the system, a sub-system, independently and concurrently with the others in order to allow for tests, calibration, etc.

Each Control Unit knows how to partition "out " or "in" its children. Excluding a child from the hierarchy implies that its state is not taken into account any more by the parent in its decision process, that the parent will not send commands to it and that the owner operator releases ownership so that another operator can work with it (Figure 36).

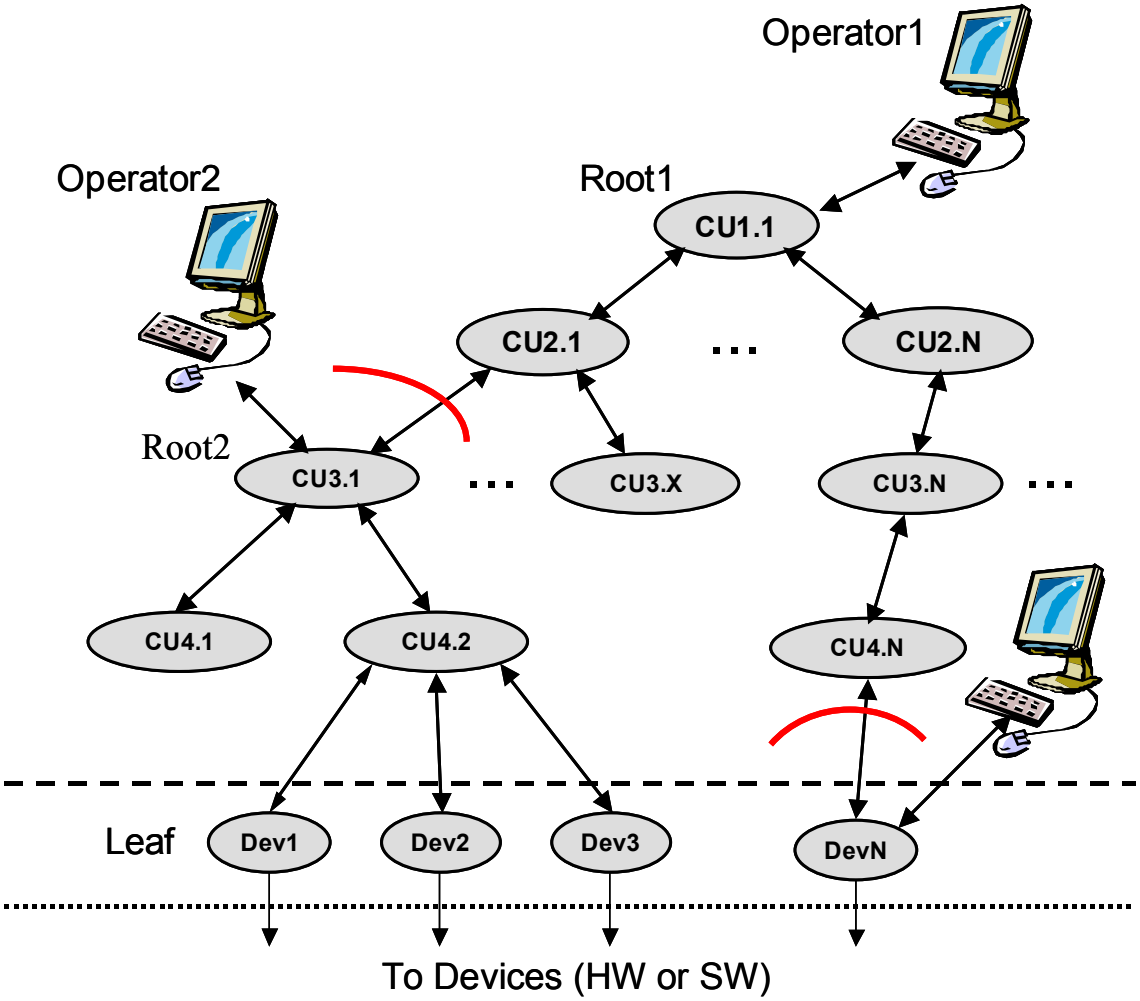

Figure 36 Partitioning the hierarchy.

It was felt that excluding completely a part of the tree was not flexible enough, so the following partitioning modes were defined and implemented in the Framework:

- **Included** - A component is included in the control hierarchy; it receives commands from and sends its state to its parent.
- **Excluded** - A component is excluded from the hierarchy, it does not receive commands and its state is not taken into account by its parent. This mode can be used when the component is either faulty or ready to work in stand-alone mode.
- **Manual** - A component is partially excluded from the hierarchy in that it does not receive commands but its state is still taken into account by its parent. This mode can be used to make sure the system will not send commands to a component while an expert is working on it. Since the component's state is still being taken into account, as soon as the component is fixed the operations will proceed.

- **Ignored** - A component can be ignored, meaning that its state is not taken into account by the parent but it still receives commands. This mode can be useful if a component is reporting the wrong state or if it is only partially faulty and the operator wants to proceed nevertheless.

The partitioning mechanism has also been implemented using PVSSII and SMI++ integrated tools.

Error handling

Error handling is the capability of the control system to detect errors and to attempt recovery from them. It should also inform and guide the operators and to record/archive the information about problems for maintaining statistics and for further analysis offline.

Since SMI++ is also a rule-based system, errors can be handled and recovered using the same mechanism used for "standard" system behaviour. There is no basic difference between implementing rules like "when system configured start run" and "when system in error reset it". The recovery from known error conditions can be automated using the hierarchical control tools based on sub-system's states. In conjunction with the error recovery provided by SMI++ full use will be made of the powerful alarm handling tools provided by PVSS II for allowing equipment to generate alarms (possibly using the same conditions that generate states), for archiving, filtering, summarizing and displaying alarms to users and to allow users to mask and/or acknowledge alarms.

Distributed systems

Both PVSSII and SMI++ allow for the implementation of large distributed and decentralized systems. There is no rule for the mapping of Control Units and Device Units into machines, i.e. there can be one or more of these units per machine depending on their complexity, or other factors such as development teams they "belong" to. The framework will allow users to describe their system and run it transparently across several computers. Since both PVSS II and SMI++ can run on mixed environments comprising Linux and Windows machines, the user can also choose the best platform for each specific task.

System configuration

Each component of the system, be it a front-end electronics board, a high voltage channel or a physics algorithm in the PC farm, will have to be initialised, configured and monitored for different activities or running modes. This can involve the management and transfer of large amounts of data. Even though the control, including the downloading of configuration data, of each component is done through the SCADA system (this is the only interface to the device), the data required for this operation will not reside at all times in PVSS for two reasons:

- Performance: Currently the PVSS database is not made to store large amounts of static data. The PVSS database is optimised for dynamic data, i.e. all data are loaded into memory for efficiency.
- Flexibility: The configuration of the control system itself has also to be stored, if one of the machines fails and has to be replaced its configuration parameters have to be available.

The configuration data will reside in the Configuration Database. This database will contain the information necessary to locate, initialise and configure all components. Some of the of data stored in the configuration database includes:

- Activity classifications (running modes)
- Device type description: decomposition in components, addressing protocols, etc.
- Device description: name, serial number, description, address, connections to other modules, etc.

- Device parameters by activity

The configuration data relevant to each PVSS sub-system (for PVSS itself and for the devices connected to it) will be obtained by each sub-system whenever necessary, for example: at power up, on change or on user request. The tools to edit the configuration database by the users will be integrated in the control framework, i.e. the user will see a single configuration tool based on PVSS tools which will trigger the import/export mechanisms between the configuration database and the PVSS sub-system whenever appropriate. The implementation of the configuration database will follow the main-stream database technology, also applied in the offline computing environment.

Interface to external systems

There are several external systems to the LHCb control system with which information has to be exchanged:

- the LHC machine
- the CERN Technical Services
- the CERN Safety System

For these systems, a protocol, the Data Interchange Protocol (DIP), is being agreed upon by all parties involved. This is the responsibility of the LHC Data Interchange Working Group (LDIWG, [41]). Once this protocol is defined, the framework will provide access to the data coming from these systems.

System operation

The framework will provide configurable operation panels. These panels will have predefined areas showing the states of the hierarchical components, their partitioning modes, their alarm states, etc. and user defined areas that are specific to the task of that particular component. The user can navigate through the hierarchy by clicking on the different components.

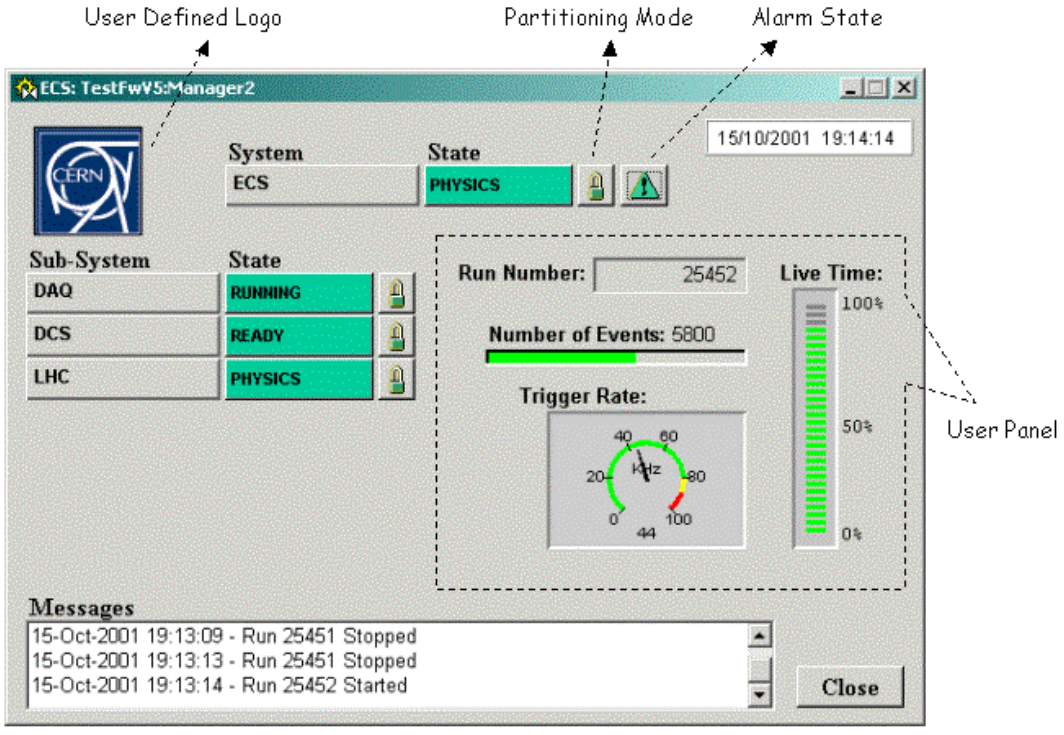

Figure 37 Prototype Run Control interface.

The panel showing the component at the top of the hierarchy provides a high-level view of the complete experiment and allows the user to interact with the different sub-systems, the DCS, the DAQ, etc. The main interface to the experiment is normally called the "Run Control". The Run Control panel of the first prototype is shown in Figure 37. The operation of sub-systems or complete sub-detectors, when working in stand-alone mode, is based on the same tools and will provide similar interfaces.

4.6.2. Data Acquisition Control

LHCb's Data Acquisition system, including the timing and fast control system, the front-end electronics, the readout chain and the event-building network, will be composed of thousands of electronics boards or chips. These electronics have to be initialised, configured, monitored and operated. There are two basic categories of electronics:

- Electronics boards or chips close to the detector in the radiation area. This electronics has been designed with the radiation constraints in mind and require only the I2C and JTAG protocols to access chips.
- Boards in counting rooms (no radiation), these boards can make use of large memory chips or processors and they require I2C, JTAG and a simple parallel bus to access the board components.

The architecture devised for the control of electronics is represented in Figure 38. All electronics equipment will contain a slave interface (S) providing the necessary protocols: I2C, JTAG and a simple parallel bus. When there is a need to control electronics located directly on the detectors, where radiation levels can be high, I2C and JTAG are driven over approximately 10 meters, from the board containing the slave interface to the chips on the detector. This avoids the necessity of radiation-hard slave interfaces, since they only have to be radiation tolerant. The slave interfaces are then connected via a master PCI board (M) to a PC. Depending on the protocol there might be the need for an Intermediate (I) board to transform the long-distance protocol into the short-distance protocol.

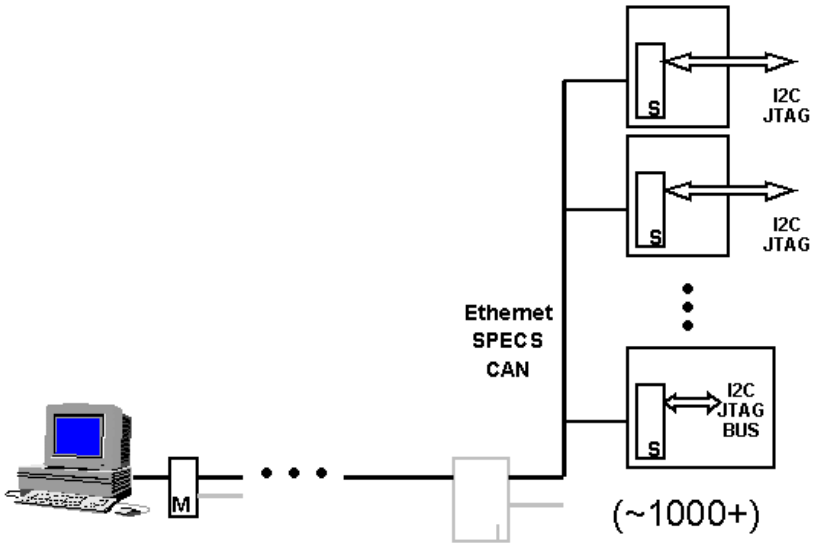

Figure 38 Schematic view of the control path into electronics boards.

One important requirement for the slave interface is that resetting the slave part on the board should not perturb data-taking activities, i.e. it should not induce signal variations that might disturb the rest of the board's components.

Three solutions have been agreed by the collaboration for interfacing electronics to the control system, the SPECS or the ATLAS ELMB for the radiation areas and Credit Card sized PCs for non-radiation areas:

- The Serial Protocol for Experiment Control System (SPECS) [42], is an evolution of the ATLAS SPAC (Serial Protocol for the Atlas Calorimeter). The SPECS slave has been improved for radiation tolerance and the SPECS Master for increased functionality. The SPECS protocol can transfer data at rates up to 10 Mbit/s. The SPECS slave is made radiation tolerant and single event upset (SEU) tolerant by using an anti-fuse FPGA and implementing triple voting on all necessary registers. The SPECS Master card is a PCI card implementing four SPECS interfaces (i.e. it can drive four SPECS buses). The SPECS specifies the use of an intermediate board to translate the long-distance protocol (~100 meters, from the counting room where the PC is to the other side of the wall) into the short-distance protocol (a few meters) to the SPECS slaves.
- The ATLAS Embedded Local Monitoring Box (ELMB) [43] is based on micro-controllers and uses the CAN bus as an interface. The ELMB contains 64 multiplexed ADC channels and was originally designed as an I/O device for analogue and digital values. Since it outputs I2C and JTAG it can also be used to control electronics. The CAN bus has a bandwidth of 500 kbit/s for the envisaged length of the bus (~100m). The ELMB's mechanism for coping with small doses of radiation is to have two micro-controllers, which can reset each other in case of problems. Any commercial CAN Master PCI card can be used to control the CAN branch. The ELMB has some degree of intelligence. Its micro-controller can be programmed to execute user code, for example to monitor FPGA code against SEUs. This feature will be used with moderation for two reasons: the development environment is complex and the micro-controller program can suffer itself from SEUs.
- Credit-Card PCs (CC-PC) [44] will be used to control electronics in counting rooms. The electronics in the counting rooms are normally VME sized boards (9Ux400mm). It was decided not to use the VME bus for control as there is always a danger that one failing board will block the whole bus segment. The solution adopted is to have point-to-point links to each board via Ethernet and to install on each board a commercial credit-card sized (66x85x12 mm^3) PC. The CC-PC (Figure 39) contains an Intel Pentium compatible CPU and up to 64 MB of memory. It outputs I2C, JTAG, via a special card, and the PCI bus, which can be easily converted into a simpler parallel bus. These CC-PCs will probably run Linux and will be booted remotely via the network.

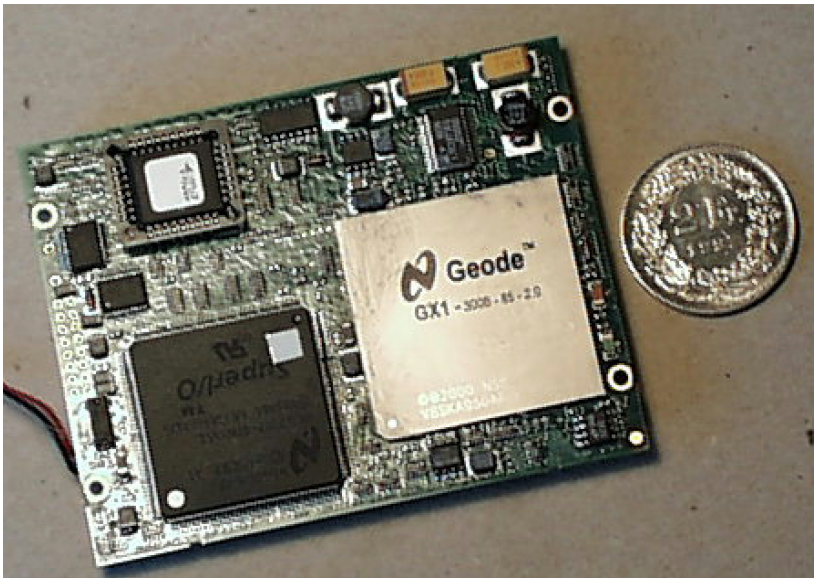

Figure 39 Photograph of a Credit-Card PC. A two-franc coin is shown for size comparison.

The Event Filter Farm (EFF) will make use of commodity items, it comprises several hundreds standard PCs and its control does not need dedicated hardware developments. Each CPU in the farm, including the Sub-Farm Controllers (SFC) will have an independent Ethernet connection for control purposes separated from the data path. The architecture of the EFF control is represented in Figure 40. The Control PCs connected to each branch of the EFF will be responsible for downloading the correct software into each CPU and for monitoring their operation, including the monitoring and control of the physics/trigger algorithms. The control of the EFF will be completely integrated in the ECS. Some R&D has been done on using the ECS SCADA tool to control and monitor a farm of CPUs with success [47]. This approach is now being followed by the IT-PDP group. LHCb's EFF control will benefit from the developments made by this group.

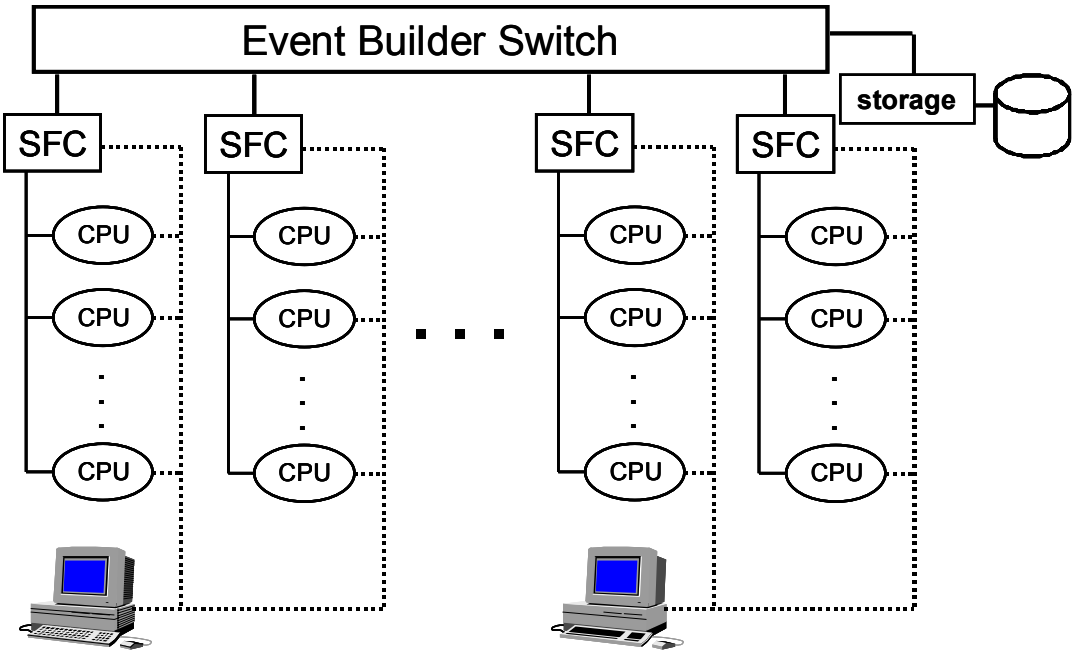

Figure 40 Schematic view of the control of the Event Filter Farm

4.6.3. Detector Control

A very large part of LHCb's control system is the interface to all the equipment involved in the Detector Control System (DCS). These include high voltage and low voltage power supplies, temperature and humidity sensors, and many other I/O devices used for calibration, alignment, mechanics, etc.

These devices are integrated into the control system via a PCI card on a PC, either directly, via a fieldbus or using a Programmable Logic Controller (PLC). The generic architectural options for the control of detector equipment are shown in Figure 41.

The choice of this equipment is largely the responsibility of the sub-detector teams due to their specific requirements. This process is still at a very early stage and most groups have not yet decided on any equipment.

Aiming for standardisation the following guidelines have been adopted by all LHCb detector groups for the control of this type of equipment:

- Commercial equipment will be used as much as possible.
- The HW interface to the equipment should be one of the CERN recommended fieldbuses: Profibus, CAN, WorldFip or Ethernet. Devices should be accessible via a PCI card on a PC, not via VME.

- The SW interface to the equipment should be an OPC (OLE for Process Control) server [40], preferably delivered by the HW manufacturer.
- PLCs can be used whenever fast control loops are needed or whenever the safety of the system requires it. The CERN recommended manufacturers are Schneider and Siemens.

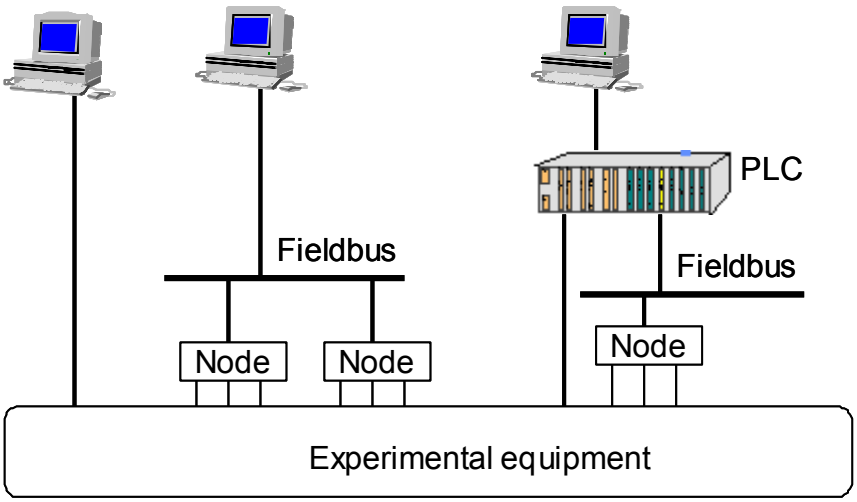

Figure 41 Schematic view of the connection to DCS type devices.

In anticipation of the choice of the sub-detectors some equipment is already being integrated in the framework as ready-to-use components: this is the case of CAEN high voltage power supplies, ISEG and WIENER low voltage supplies and the Atlas ELMB for analogue and digital I/O. In general, any device providing an OPC server can be easily integrated since PVSS provides an OPC client: this is the case of WAGO I/O modules, the CERN recommended PLCs, and many other industrial devices.

The gas systems of the different sub-detectors (the Outer Tracker, both RICHes and the Muon systems) are being developed in the framework of the Gas Working Group (GWG) in common with the gas systems of the other LHC experiments. The GWG will provide for a common control room and a single operation and maintenance “piquet” service for all the gas systems of the four LHC experiments. This working group includes a control team, which is under the supervision of JCOP. Although the control of the gas systems will be very specialized and largely PLC based, JCOP tools including PVSS and the JCOP framework will be used for the supervisory levels. As a result the integration of the gas system monitoring in the overall experiment control system should be straight forward.

4.6.4. Infrastructure Control

The experimental infrastructure and environment has also to be monitored and when possible controlled, this includes:

- Monitoring environmental parameters in the counting rooms and experimental halls (temperatures, humidity, radiation levels, etc.).
- Monitoring and controlling the racks and the crates containing the electronics. Each rack will have sensors for temperature, humidity, water leaks and a thermo switch, which can cut the power to the rack if it overheats. Each crate and each rack can be operated independently. The control of racks is being handled in common for the four LHC experiments by the Rack Control Working Group [45]. There is also a common activity to standardise on VME mechanics and its control.

- Monitoring and control of cooling and ventilation both centrally (for example for the racks) and inside the sub-detectors. This is the task of the Joint Cooling and Ventilation (JCOV) working group and its control team [46].
- Monitoring of the electricity distribution. This information will be provided by the CERN Technical Service group via the Data Interchange Protocol (DIP).
- Monitoring of the LHCb Magnet. The experimental magnets control is also being done in common by the magnet control group. The information will be available to LHCb either directly if this system is implemented using PVSS or via the DIP protocol.
- The information gathered by the Infrastructure and Environment control sub-system has to be stored and will be used to take decisions in case of problems, for example cutting the power to crates or racks (in an orderly and organised manner) if the temperature increases or the cooling stops, etc.
- The architecture of the infrastructure monitoring and control sub-system is very similar to that of the DCS sub-system (Figure 41).

4.6.5. Detector Safety System

The CERN Safety Alarm Monitoring System (CSAM) will provide the LHC experiments and their experimental areas with a safety system for level 3 alarms, i.e., for accidents or serious abnormal situations where people's lives may be in danger. The main action taken by this system in such situations is to alert the fire brigade.

LHCb's Experiment Control System is mainly a software-based system and even though it is expected to be robust and available most of the time (>95%) it was not designed with safety constraints in mind, hence there is a need for an independent system that can run stand-alone and handle equipment safety. The aim of LHCb's Detector Safety System (DSS) is to protect the experiment's equipment and to prevent situations leading to level 3 alarms.

LHCb's DSS is being developed in the framework of the LHC experiment's common detector safety system project. The DSS will be a complementary hardwired system to the ECS. The ECS (whenever available) will have all the information to react sequentially and with high granularity, for example if a rack overheats it can cut each crate in the rack one after the other and then the rack itself. The DSS would, for instance, if a rack temperature goes above a certain limit (higher than that of ECS), cut the whole rack row or even the whole barrack. The DSS will be kept simple and small, in order to allow for a high degree of reliability.

The DSS will be composed of two parts:

- The DSS front-end part can work completely stand-alone. It will be implemented using highly reliable devices, probably PLCs. It will receive information from hardwired sensors, make simple combinatory decisions and send hardwired actions.
- The DSS back-end part will gather the data from the front-end, archive it, process it and relay the information to the ECS. This part can be implemented using standard ECS tools, i.e. the JCOP framework, since its malfunctioning would not affect the front-end part. It will also be used to configure the front-end, e.g. to disable a sensor that is known to be malfunctioning.

The architecture of the DSS can be seen in Figure 42. The DSS will be implemented as a set of tools that can be used to implement the front-end and the back end parts and that can be configured and tailored for the different experiments and experimental areas.

In LHCb the DSS toolkit will be deployed not only in the underground area (UX85) but also in the surface for the gas building (SG8) and the main experiment building (SX8). The DSS geographical distribution is shown in Figure 43.

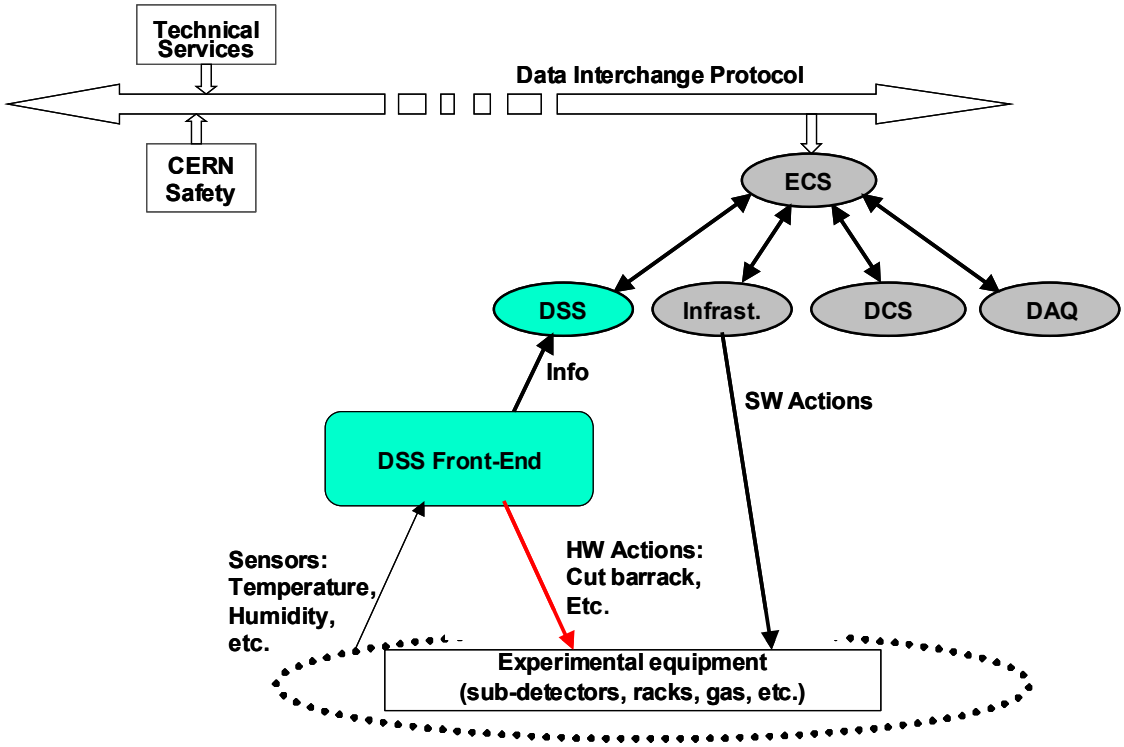

Figure 42 Architecture of the Detector Safety System.

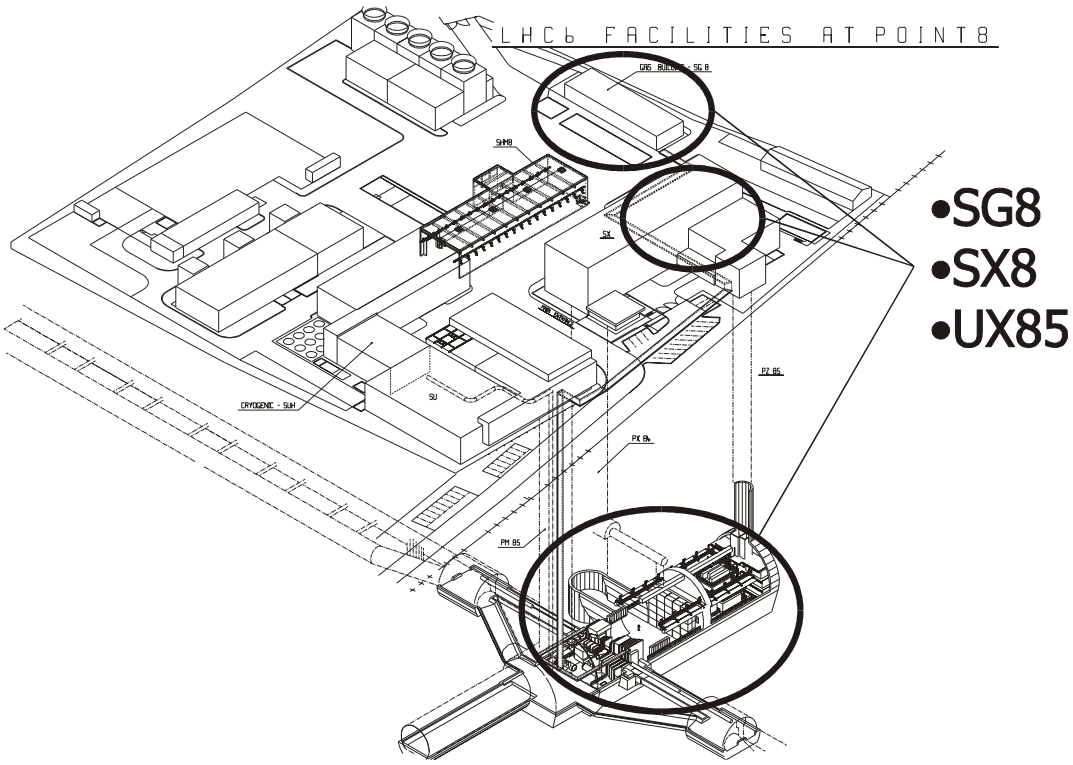

Figure 43 Geographical distribution of the Detector Safety System in LHCb.

4.6.6. Data Processing and Offline Computing

The data acquired by LHCb will be promptly reconstructed in the Event Filter Farm. This provides immediate feedback to the shift crew on the quality of the data. The experiment control system will provide for the detector performance monitoring by combining information gathered by the

SCADA system (allowing the identification of problems related to for example wrong high voltage settings or gas mixtures) and statistical information resulting from the analysis of the data.

The data quality monitoring will take place either in the individual CPUs of the Filter Farm or while the data are written to permanent storage. The results of these algorithms are statistical information, such as counters or histograms. These statistical data will be acquired through the ECS system and, in case of the algorithm running on the Filter Farm, will be combined to form a single set of histograms and counters[2].

The software performing this aggregation of statistical data will run under the control of the ECS system and will interface to the LHCb-standard histogramming sub-system to provide the necessary information.

The offline software will require bookkeeping information on the sets of data ("runs") being acquired, stored, reconstructed, reprocessed, etc It will also need information on the conditions under which these runs where acquired.

There will be a unique bookkeeping tool (the bookkeeping database) in LHCb. It will be used in order to store information about physics data-taking runs, but also test-beam runs, Monte-Carlo productions, and during reprocessing. This database will be shared between online and offline.

The detector "conditions" information will be stored in the Conditions Database. The ECS system will keep track (through the SCADA tools) of the state of the whole detector, including the online system, and will archive most of the state information permanently. A subset of this state information is of crucial importance to offline algorithms, like reconstruction or analysis. Hence, a mechanism will exist to interface the SCADA state information to the offline Conditions Database [8]. This will ensure that selected quantities are transferred from the SCADA archive to the conditions database. These quantities include: environmental parameters (such as pressures and temperatures), the details of the current configuration (i.e. high voltage settings, the current parameters downloaded into the front-end electronics, which parts of the detector are being read out) and results of online calibration and alignment activities. This transfer will be done during system initialisation and will be updated whenever new data are available.

The online calibration and alignment tasks will not have direct access to the conditions database. They will deliver their data to the ECS system, which in-turn will feed it through a standard mechanism to the conditions database.

4.7 Scale of the System

In this section, we will describe the tentative scale of the system in terms of number of elements needed to satisfy the performance requirements. In addition, some functional requirements, such as partitioning, will influence the detailed numbers. For example, partitioning will prevent the assignment of a given RU to the dataflow of two different sub-detectors or partitions.

4.7.1. Timing and Fast Control

The front-end electronics of the sub-detectors and the Level-0 and Level-1 trigger systems comprise roughly a thousand TTC receiver chips (TTCrx). The TFC Switch allows dividing the sub-detector into 16 partition elements. Following these two constraints, Table 6 presents the number of TFC modules required including spares.

[2] It should be noted that these data do not need to be synchronized among different CPUs. It is not relevant, that the different sub-histograms are read at exactly the same time. Only at the end of a data-taking activity, the total statistics has to be consistent.

Table 6 EXPECTED NUMBER OF TFC MODULES REQUIRED.

TFC component	Number of modules
TTCmi + 4 TTCcf	2
Trigger splitter	3
Readout Supervisor	12
TFC Switch	2
Throttle Switch	3
TTCtx	18
Optical coupler	50
Throttle OR	40

4.7.2. Data-Flow System

In the absence of a complete simulation of the LHCb raw data, including the front-end electronics we founded our numbers on very rudimentary estimates based on occupancies determined from Monte-Carlo simulations.

Table 7 shows the expected numbers of elements in the readout system upstream of the readout network.

It should be noted that the original event sizes from the Level-1 front-end electronics boards add up to ~73 kB. These numbers are based on occupancy estimates and some global assumptions on encoding the addresses of the hit channels after zero-suppression. There is limited accounting for electronics noise or background in these figures. We thus scaled the event sizes of all the detectors up such that the average total event size amounts to ~100 kB (see Section 2.9). The number of crates quoted in Table 7 is to house the DAQ electronics (FEMs and RUs) only. It does not denote the number of crates needed for the Level-1 front-end electronics.

The design process starts with the amount of data that is produced in one Level-1 front-end electronics board (after scaling), and the data rate per input port to the readout network that can be handled. For the latter we chose 80 MB/s, which represents a load of 66 % per link (1 Gb/s), a reasonable safety factor as simulations show (Section 4.4.2). From these two numbers, the data fragment sizes and the desired maximum bandwidth used on a link, a 'target multiplexing factor' per sub-detector — a consequence of partitioning — is calculated[3], which we try to realize by connecting Level-1 front-end electronics boards to FEMs and FEMs to RUs[4].

Due to the integer nature of the multiplexing factors, the ideal multiplexing factor cannot always be achieved. To be on the safe side, wherever possible a lower multiplexing factor has been chosen. We can conclude that the system can be implemented with 248 Network Processor mezzanine cards mounted on 127 carrier boards. The scale of the readout network will be 60 input ports and 60 output ports[5].

[3] The Readout Supervisors are handled somewhat differently. In principle, one port of the Readout Network per RS should be associated, to strictly conform to the partitioning principles. This would, however, lead to 10 more RUs and 10 more ports in the RN. This would imply financial consequences that cannot be justified. We therefore decided to connect the RSs to RUs as if they were all belonging to one sub-detector but will load special code into these FEM/RU modules such that there is no data merging performed and the destination assignment will follow the partitioning. Basically the appropriate FEM/RUs will act as partition aware simple multiplexers.

[4] FEMs and RUs are identical modules. They are just distinguished here for clarity.

[5] There are additional ports needed in the readout network, since we reuse the connectivity already provided by the readout network to connect the data path to the computing infrastructure (storage).

Table 7 NUMBER OF READOUT ELEMENTS IN THE DATAFLOW SYSTEM BASED ON AVERAGE OCCUPANCIES IN THE DIFFERENT SUB-DETECTORS[6]. WHERE MEANINGFUL, THE SUM OF THE QUANTITIES IS ALSO ADDED.

	Velo	IT	OT	RICH	Calori-metry	Muon	Level-0	Level-1	Readout Super-visor	Total
L1 Boards	100	225	102	54	26	12	6	1	10	**536**
Fragment Size/L1 Board [kB]	0.08	0.07	0.42	0.26	0.55	0.24	0.25	0.25	0.25	
Data Rate/L1 Board [MB/s]	3.2	2.7	16.8	10.4	22.0	9.7	10.0	10.0	10.0	
Total Rate [MB/s]	322	604	1713	564	573	116	60	10	10	**3973**
FEM Outputs	20	38	102	8	19	6	6	1	2	**202**
RU Outputs	4	8	26	8	9	2	1	1	1	**60**
Ouput BW/RU [MB/s]	80.6	80.6	67.2	73.1	74.4	58.0	60.0	10.0	10.0	
Average frag-ment Size [kB]	2.01	2.01	1.68	1.83	1.86	1.45	1.50	0.25	0.25	
EventSize [kB]	8.1	16.1	43.7	14.6	16.8	2.9	1.5	0.25	0.25	
Carrier Boards	24	46	26	13	9	4	1	1	3	**127**
Mezzanines	48	92	52	24	16	8	2	1	5	**248**
Crates	2	3	2	2	3	1	1	1	1	**16**

4.7.3. Event Filter Farm

The size of the farm is critically dependent on the processing time required to reach a trigger decision.

Table 8 ITEMS TO BUILD AN EVENT FILTER FARM

Item	Quantity
CPUs	900
SFC	66
Switches	66

From the SPEC web page [48], one can see that a current 1 GHz system has approximately the power of 45 SI95 units. Using Moore's law[7] [49] and assuming procurement in 2005 we estimate the size of the farm to be ~900 CPUs. The minimal number of sub-farms is derived from the minimal network compatible with the allowed link-load, that is 60. This means that the minimal system needs 60 SFCs and 60 data switches. Not counted here are spares and the controls switches.

[6] The figures for RICH and Calorimetry are sums (e.g. for data rates) or averages (e.g. for data sizes). The RICH numbers are composed of RICH1 and RICH2, whereas the Calorimetry numbers are assembled from SPD/PS, ECal and HCal.
[7] Or rather a crude corollary, stating that doubling the number of transistors, which is what Moore's law predicts, is equivalent to doubling the CPU power.

4.7.4. ECS

In order to give an idea of the scale of the Experiment Control System we will estimate the number of Control PCs that will be needed to implement the complete control system. The major control areas are: the control of electronics, the control of DCS devices and the control of the Event Filter Farm. We will base this exercise on the type of interface or the type of device connected to the control system.

Table 9 gives the decomposition of Controls PCs associated to the control of CC-PCs per sub-detector. It is assumed that one Controls PC can drive ~50 CC-PCs.

Table 9 PCs CONTROLLING ELECTRONICS INTERFACED VIA ETHERNET/CC-PCs

CC-PC Table	VELO	IT	OT	RICH1	RICH2	SPD/ PS	ECal	HCal	Muon	Trigger	Central	Total
Level 1 Boards	100	225	102	21	33	8	14	4	12	70		589
FEM/RUs	24	46	26	5	8	2	6	1	4	2	3	127
TFC	9	18	6	3	4	2	3	2	2	4	13	66
Total CC-PCs	133	289	134	29	45	12	23	7	6	76	16	770
Controls PCs	3	6	3	1	1	1	1	1	1	2	1	21

Table 10 gives the number of Controls PCs needed to drive electronics controlled via the SPECS interface. Each SPECS master board has 4 independent channels and can in principle drive up-to 128 slaves. It is assumed that up-to 2 SPECS master cards can be housed in a controls PC.

Table 10 PCs CONTROLLING ELECTRONICS INTERFACED VIA SPECS

SPECS Table	Velo	IT	OT	RICH1	RICH2	SPD/PS	ECal	HCal	Total
SPECS Slaves	28	50	40	84	136	94	188	47	667
SPECS Masters	1	2	2	4	6	8	14	4	41
Controls PCs	1	1	1	1	1	1	2	1	9

Table 11 PCs CONTROLLING ELECTRONICS INTERFACED VIA CAN/ELMB

ELMB Table	Muon
ELMBs	756
CAN Masters	34
Controls PCs	6

Table 11 lists the number of Controls PCs needed to drive the CAN buses controlling the ELMB-based interfaces [50]. It is assumed that each CAN master can control up to 32 ELMBs and that 6 CAN masters can be housed in one Controls PC.

Finally Table 12 gives the number of Controls PCs needed to control the Event Filter Farm. The assumption is that one control PC will handle two sub-farms.

Table 12 PCs CONTROLLING THE EVENT FILTER FARM

EFF Table	Quantity
CPU Nodes	900
Sub Farm Controlers	60
Controls PCs	30

Table 13 summarises the number of Controls PCs needed for the DCS equipment, i.e. the equipment controlling and monitoring the operational state of the LHCb detector and infrastructure.

Table 13 PCs CONTROLLING DCS DEVICES

DCS Table	Vertex	IT	OT	RICH (RICH1 & RICH2)	Calo-rimeters SPD/PS, Ecal,Hcal	Muon	Central Infra-structure & DSS	Total
HV Channels	104	900	1200	430	1200	2160		**5994**
LV Channels	104	900		220		160		**160**
Temperatures probes	104	360	30		50	180		**724**
Other Systems to Control	vacuum motion cooling	cooling	gas calibration alignment mechanics	gas alignment	calibration alignment	gas cooling	racks, crates environment	
PLCs	3	1					5	**9**
Controls PCs	5	3	5	6	5	5	7	**36**

Table 14 shows a summary of the previous tables in terms of Controls PCs. The total number needed is ~100, where almost 30% are attributed to the control of the Event Filter Farm.

Table 14 TOTAL NUMBER OF CONTROLS PCS

Equipment	Controls PCs
CC-PC based electronics	21
SPECS based electronics	9
ELMB based electronics	6
DCS Devices	36
Event FilterFarm	30
Central Control (Racks, Crates)	5
Total	**107**

4.8 Online Computing Infrastructure

In previous sections we have focussed more on implementation of the components of the online system. In this section, we will discuss the general infrastructure that is needed to make the online system operational. Among the items described here are the central computing infrastructure, servers and networking, but also power and cooling.

4.8.1. Computing Infrastructure

The computing infrastructure (Figure 44) can be logically split into the infrastructure for acquiring the physics data and the infrastructure required for the control system and for general purpose computing.

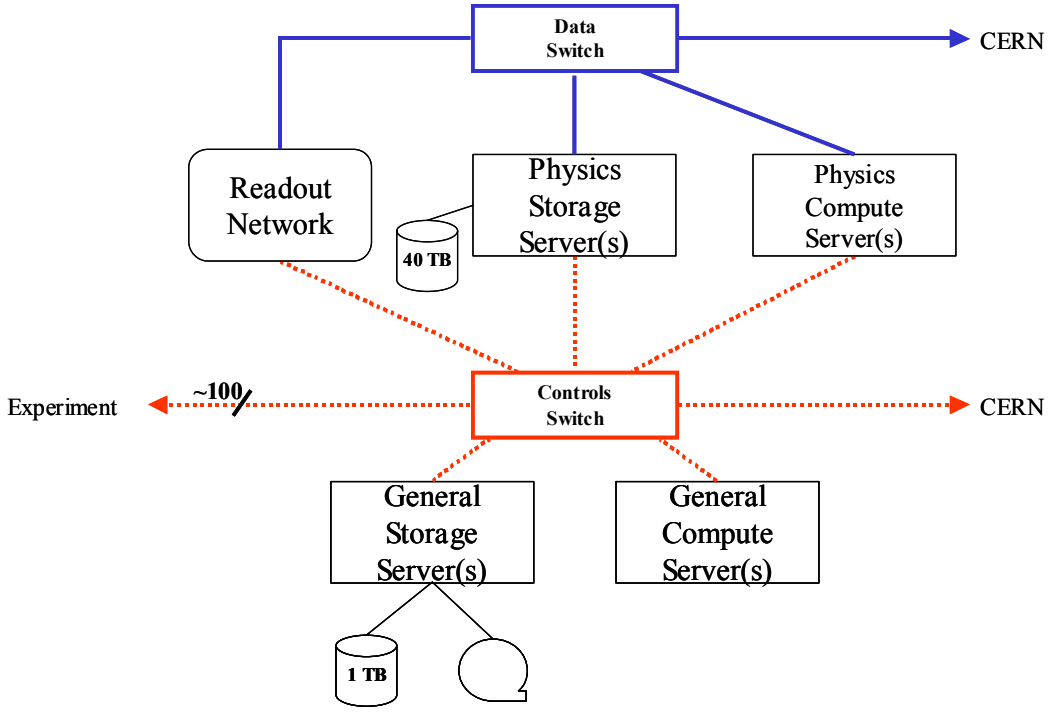

Figure 44 Architecture of the online computing infrastructure[8].

The physics storage server (Figure 44) will receive the accepted and reconstructed events from the CPU farm and will buffer them temporarily, since we intend to use the CERN Computer centre's storage system for the permanent storage of the physics data. We plan to install disk space for ~10 days worth of data, in case the links to the computer centre are interrupted. At a production rate of ~4 TB per day this implies a disk capacity of ~40 TB. This storage will have no backup to more permanent media, e.g. tapes.

There will also be a need to provide a certain amount of computing power to perform some analysis of the physics data to ensure the quality of the data. Two mid-range CPU servers will be required for this task in order to ensure there are no single points of failure.

The second part of the online computing infrastructure will consist of a storage server holding all software needed to run the system, and also the databases and SCADA permanent archives needed for system operation. This storage will be using redundancy technologies (e.g. RAID-5) and will also be connected to a backup system for safety.

The general compute servers will run the central parts of the controls system, but will also be responsible for extracting configuration data for individual modules from the configuration database. Again two mid-range CPU servers will be required.

4.8.2. Networking Infrastructure

Side-by-side with the DAQ network that transports the physics data, there will be a controls network installed in the experiment. Again, the technology for this network will be Ethernet,

[8] The picture in Figure 44 is rather logical than physical. It is a-priori not necessary that the controls and the data switches are physically different boxes as long as the performance is sufficient.

because of its abundance and its wide range of different bandwidth implementations (10/100/1000/10000 Mb/s). The controls network will be structured in a deeply hierarchical manner, with 1 or 10 Gb/s NICs in the servers and 100 Mb/s sections at the controls interfaces. Figure 45 shows the implementation of the final controls network distribution.

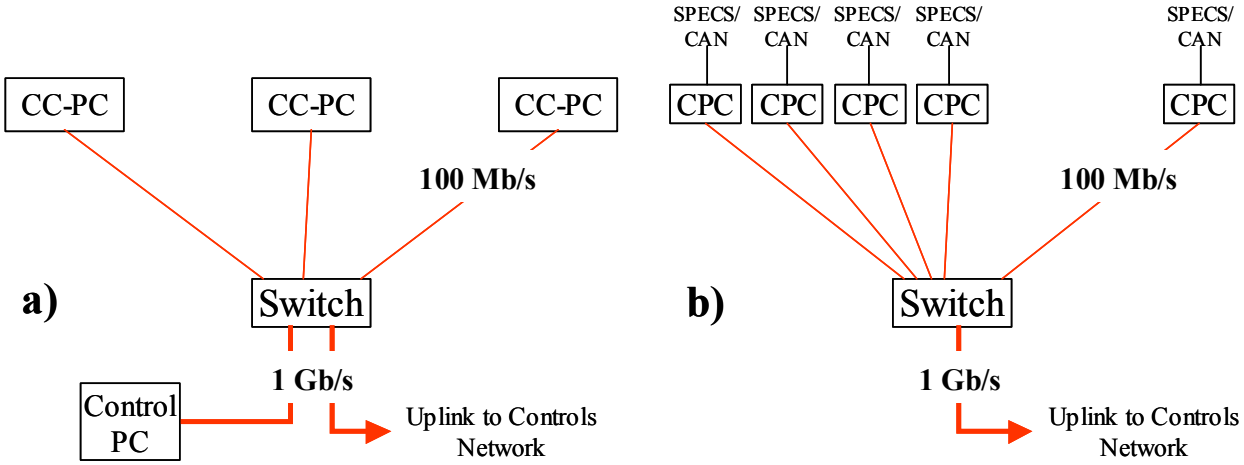

Figure 45 Implementation of the final network distribution for the case of Credit-Card PCs (a) or Control PCs as end-nodes (b).

The switches in Figure 45 will be standard so-called edge-switches. These switches will be very abundant in the future, since they will provide the basic connectivity to the individual offices in LAN environments when 10-GbEthernet will become the backbone technology. Hence, the prices for these switches will be low. For other technologies as interfaces to electronics (ELMB, SPECS), clearly some Credit-Card PCs can be replaced with controls PCs. In this case, only one Gb port is needed on the switch.

All the uplinks will be connected to a large (standard) backbone switch, which will also receive the links from the general storage controllers and the general compute servers in Figure 44[9].

4.8.3. Power and Cooling

There will be 2 MW of electric power installed at pit 8 for the LHCb experiment (excluding the magnet). This power should be sufficient for the electronics and other equipment[10].

Cooling, however, is a somewhat bigger concern, in the sense that there is about 1.5 MW of cooling power available in form of 'cold water' for cooling electronics modules and 0.5 MW of cooling power for air-conditioning. Depending on the implementation of the CPU farm (see Section 4.5), there might be a significant amount of power to be cooled away through air-cooling, i.e. though cooling the environmental air of the equipment. This is clearly much less efficient than blowing cold air, via fan trays, across the electronics and taking the heat out by means of water-cooled heat exchangers. This aspect of the implementation has to be born in mind when choosing the equipment in question[11].

The main computing infrastructure (Storage- and Compute servers and central switches) will be powered through UPS's (Uninterruptible Power Supplies) to guarantee maximum up time.

[9] This backbone switch does not necessarily need to be as highly performing as the switching network of the Readout Network. It is more to provide connectivity among all the nodes, then to provide performance.
[10] Note that the DELPHI experiment managed very well with less power and much older electronics, i.e. consuming much more power and was, to a large extent, of the same scale than LHCb.
[11] The CERN computer Centre is facing a similar problem and a solution will have to be found at a much larger scale.

4.8.4. Location of Equipment

It is obvious for a lot of the equipment where it will be located

- The front-end electronics will be located in the cavern of pit 8 of the LHC.
- All the TFC equipment will also be located in the cavern for latency reasons.
- Very many of the control PCs will be underground, close to the equipment they control.
- For floor-space and maintenance considerations, most likely the CPU farm will be located upstairs, i.e. on the surface.

The choice of Gb Ethernet as link technology allows in principle distances between 100 m (Unshielded Twisted Pair implementation) and 500 m (short haul optical). Hence, the choice for the rest of the equipment, such as FEMs and RUs is largely arbitrary[12]. For convenience and operational reasons, it would be advantageous to house as much as possible of the equipment at the surface. We will follow closely the market trend and the price evolution and decide on a cost/benefit basis where to locate the equipment.

4.8.5. Control Room

The LHCb control room will be located on the surface in the former Delphi control room. It will be the place from which the entire experiment will be controlled and monitored by the shift crew. The crew will have at its disposal several PCs or workstations to perform their task. These will run the user interfaces and panels of the control system. Other terminals or screens will be installed to permanently display information important for the understanding of the state of the experiment, such as

- State of key components of the DAQ system
- State of the high-voltage systems of the different sub-detectors
- Asserted alarms of the control system
- State of the LHC machine

Besides the infrastructure installed for the shift crew, there will be a certain number of PCs available for sub-detector use, e.g. for experts investigating problems or for ensuring the quality of the data taken. All PCs will run the LHCb control software, at least the user interface part, while the algorithmic part will run on the compute servers (see Figure 44).

4.8.6. Connection to the CERN Computer Centre

The physical connection (fibre optics links) between point 8 of the LHC and the CERN computer centre will be provided as part of the general networking infrastructure of CERN. Figure 46 shows the connection from point 8 to the general CERN networking infrastructure.

It can be seen that there is redundancy in the connectivity between point 8 and the computer centre (Building 513). These fibres will carry high-speed data protocols, such as DWDM (Dense Wave Division Multiplexing) reaching 80 Gb/s aggregated bandwidth. LHCb will need only a very small fraction of this bandwidth. The average rate to storage and hence to the CERN computing centre amounts to ~40 MB/s. Even taking into account a 50 % higher rate from the computing centre to the CPU farm during re-processing of the data when the accelerator is not running, this load should easily be handled by a 1 Gb/s link. All controls traffic will use another channel, which will also be a 1 Gb/s link.

[12] This is only true if the physical layer (optical or twisted pair) can be arbitrarily chosen, or if optical transmission is used everywhere.

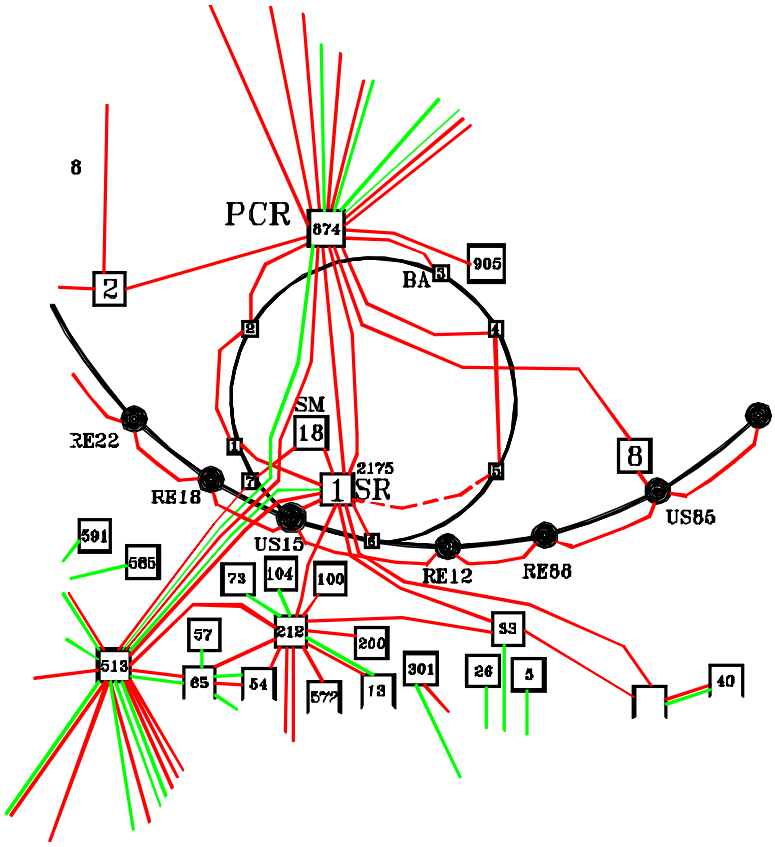

Figure 46 Network connectivity from Point 8 of LHC to the CERN Computer centre. The figure shows the SPS in the centre and a section of the LHC ring with points 8 (LHCb) and 1 (Atlas). The path from LHCb either goes via the PCR (874) to Building 513 or via US15 to the PCR and then further to building 513.

The link to the computer centre will use the transport protocol in fashion at the time, e.g. TCP/IP. CERN's central data recording software will govern the transfer to the computer centre's storage facility, where the data will be permanently archived.

Chapter 5 Cost, Planning and Responsibilities

This chapter deals with the managerial aspects of the project. Information is presented on the current estimates of the cost of the system, the planning schedule and the distribution of the responsibilities for the implementation of the system.

5.1 Costing

The costing is based on the number of modules required to build the system as discussed in Section 4.7. Provision has been made for spares, typically 10%.

Table 15 shows the cost of the different components of the TFC system. The estimates are based on component costs of standard TTC modules and on the cost of LHCb specific prototype units.

Table 15 COST BREAKDOWN FOR THE TFC SYSTEM[1].

Module	Quantity	Unit Cost [kCHF]	Item Cost [kCHF]
TTCmi + 4 TTCcf	2	0.00	0
Trigger splitter	3	3.00	9
Readout Supervisor	12	10.50	126
TFC Switch	2	4.80	10
Throttle Switch	3	4.10	12
TTXtx	18	3.90	70
Optical couplers	50	1.65	83
Throttle OR	40	4.50	180
Optical fibre (TTC)	1100	0.03	35
Total			**525**

Table 16 COST BREAKDOWN FOR THE FEM/RU SUB-SYSTEM[2].

Item	Quantity	Unit Cost [kCHF]	Item Cost [kCHF]
Mezzanines	218	4.2	921
Carrier Boards	109	3.8	419
Cables/Fibres	738	0.04	30
Crates	18	10.0	180
Total			**1549**

The costing of the FEM/RU is based on the Network Processor based module. The cost of this will depend very strongly on the cost of the Network Processor chip itself in mid 2003, when we will start mass-production. Table 16 shows the breakdown of the cost for the FEMs and the RUs. The cost per module has been based on information supplied from the commercial vendor.

[1] The TTCmi and the TTCcf modules are paid as part of the contribution of LHCb to their development in the framework of the RD12 project.
[2] The cost of 31 Carrier Boards and 62 Mezzanine Cards that have already been included in the RICH and VELO TDRs has been subtracted.

The cost of the Readout Network has been made assuming an implementation using the NP based module, since this is known accurately. It is very likely that the cost of an implementation using commercial switches will be lower than the figure quoted in Table 17, so this estimate is considered to be conservative.

Table 17 COST OF THE READOUT NETWORK IMPLEMENTED WITH NP-BASED MODULES.

Item	Quantity	Unit Cost [kCHF]	Item Cost [kCHF]
NP Boards	60	12.29	737
Cables/Fibres	480	0.04	19
Crates	3	10	30
Total			**786**

The cost of the online farm consists of three main parts:

- The cost of the CPU power
- The cost of the Sub-Farm Controllers
- The cost of the switches connecting the SFC to the farm nodes

The cost of the CPU power is determined from an estimation of the cost per SI95 at the time of purchase of the equipment. The purchasing profile will be chosen such as to minimize the overall cost, providing that the requirements for testing, commissioning of sub-detectors are always satisfied. To this effect, it is planned to purchase some 5% of the CPUs during 2004 and the rest towards the end of 2005 in order to be installed in time for data-taking in April 2006. Any change in the machine schedule will cause us to adjust the acquisition profile accordingly. Of course, the time of the acquisition will have, through Moore's law, a big impact on the cost of the CPU farm, since more than 70% of the cost is attributable to the CPUs of the farm. Some 20 CPUs have been added to the basic requirement to be used as "hot spares" in order to replace failing nodes.

The costing shown in Table 18 is based on the figures quoted in the PASTA report [51] and includes overhead costs (cabling, console access, power etc.). It is also assumed that additional CPU power is needed to cope with operating system overheads, which have to be added to the pure CPU power requirements presented in Table 4.

The SFCs will be high-end PCs having 2 Gb Ethernet interfaces and redundant power supplies. The unit price has been estimated at 5 kCHF.

For the cost of the switches, the cost predictions have been based on reference [52]. All figures assume the minimal number of 60 sub-farms determined in Section 4.7.2.

Table 18 COST BREAKDOWN OF THE EVENT FILTER FARM.

Item	Quantity	Unit Cost [kCHF]	Item Cost [kCHF]
CPUs	920	1.4	1417
SFC	66	5.0	330
Switches	66	2.4	161
Total			**1909**

The cost for the ECS system is summarised in Table 19. It is based on information from prototyping (rack control), from commercial suppliers (Credit-Card PCs), from reasonable estimates (control PCs) and from list prices (switches).

Table 19 COST SUMMARY FOR THE ECS SYSTEM[3]

Item	Quantity	Unit Cost [kCHF]	Item Cost [kCHF]
Rack Control	150	1.5	225
PLCs for DSS	4	15	60
Control PCs	120	2	240
Credit-Card-PCs	860	0.4	344
Switches	62	3	186
Total			**1055**

The cost of the central computing infrastructure is summarized in Table 20. The prices quoted are conservative estimates of those expected end of 2003, when we will start to order the equipment.

Table 20 COST BREAKDOWN OF THE GENERAL COMPUTING INFRASTRUCTURE.

Item	Unit	Quantity	Unit Cost [kCHF]	Item Cost [kCHF]
Disk Servers	Pce	4	20	80
Disks	TB	45	5	225
Backup Tape	Pce	1	20	20
Compute Servers	Pce	4	20	80
Switches	GbE Ports	100	2	200
Control Room				100
Total				**705**

The total cost for the system is estimated to be ~6530 kCHF. Its breakdown in the different sub-systems is given in Table 21.

Table 21 COMPILATION OF ALL THE COST COMPONENTS OUTLINED IN PREVIOUS TABLES.

Sub-System	Cost [kCHF]
TFC System	525
FEM/RUs	1549
Readout Network	786
CPU Farm	1909
ECS	1055
General Computing Infrastructure	705
Total	**6529**

We are confident that we can build a reliable, scalable and high performance online system within the allocated budget.

5.2 Planning

The project schedule shown in Figure 47 has been developed based on the current planning of the start-up of the LHC accelerator in April 2006.

[3] The cost of the PVSS licence is already paid.

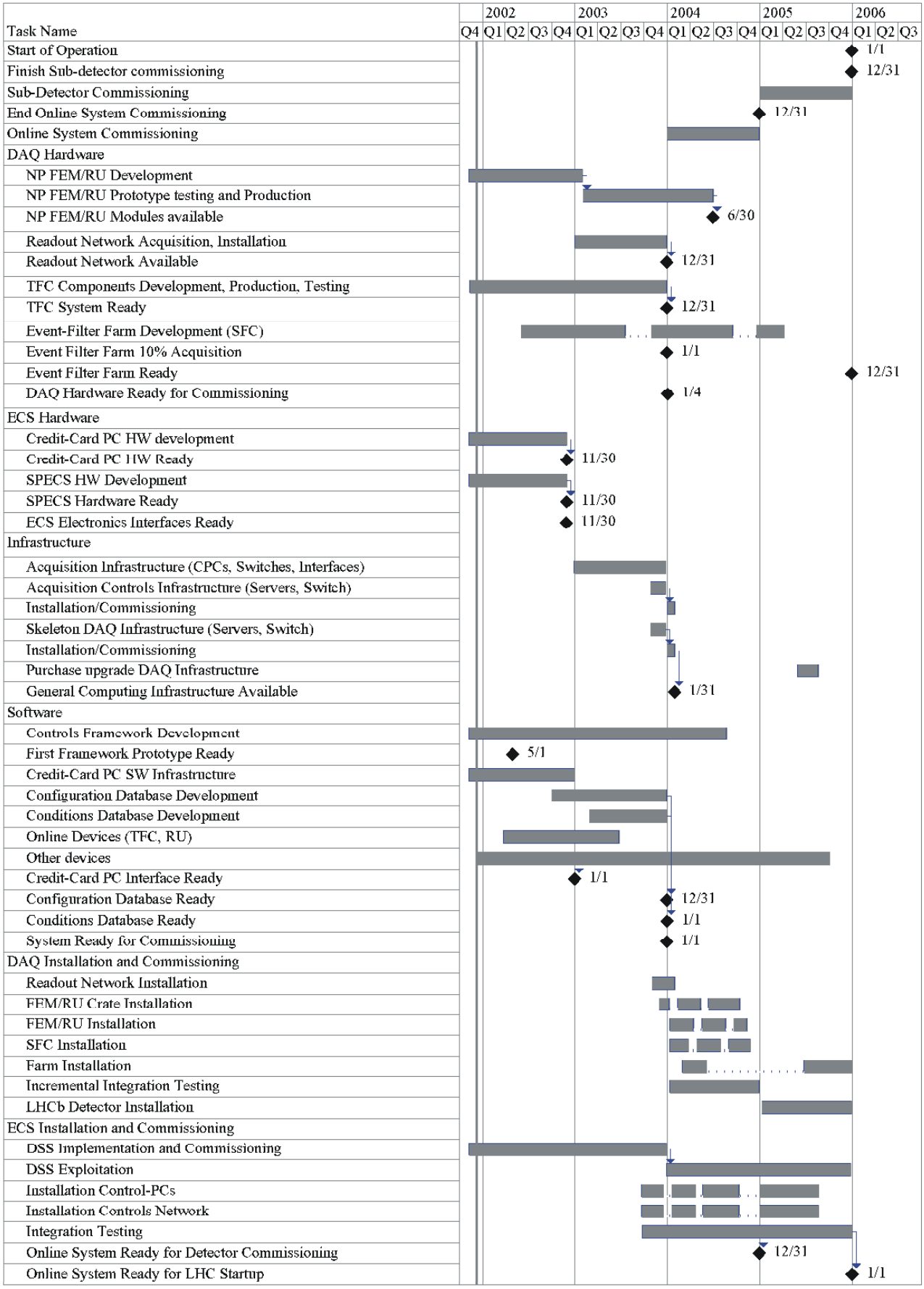

Figure 47 Project schedule for the implementation, installation and commissioning of the LHCb online system.

According to the planning, the online system is ready for being commissioned by the beginning of 2004, which means that most of the components have to be available by that time. For the Event-Filter Farm only such a part of the CPUs will be acquired as is necessary to test the principles. The bulk of the CPUs will be installed as late as possible, i.e. towards the end of 2005. Similarly, not all the RUs will be available in the beginning of 2004, but probably the final production batch will arrive by the middle of 2004. There will be, however, sufficient RUs and FEMs available at any one time for testing the system.

From the detailed planning of Figure 47 a set of major milestones has been extracted. These are listed in Table 22.

Table 22 LIST OF MAJOR MILESTONES

Milestone	Date
ECS electronics interfaces prototypes ready	1-Apr-2002
ECS software framework first release	1-Jun-2002
TFC prototypes ready	1-Jan-2003
NP-based RU prototype ready	1-Feb-2003
Readout Network implementation decision	1-Jun-2003
NP-based RU production start	1-Jul-2003
Start installation in Pit 8	1-Oct-2003
Start commissioning	1-Jan-2004

5.3 Responsibilities

In this section, we give an overview of the distribution of responsibilities for the implementation and construction of the LHCb online system. Wherever CERN/EP is mentioned without qualification it is implied that the responsibility lies within the CERN-LHCb computing group.

5.3.1. Software

The software project of the LHCb online system can be split into two broad categories, namely

- Embedded software, such as the software running on the Network Processors or in the SFCs, but also frameworks for the High Level Trigger algorithms and data monitoring software. This will be provided by the LHCb computing group.
- Software associated and making-up the ECS system. This is built using the commercial SCADA software, which is supported by CERN/IT through the JCOP project. In this section, we give an overview of the distribution of responsibilities for the implementation and construction of the LHCb online system. Wherever CERN/EP is mentioned without qualification it is implied that the responsibility lies within the CERN-LHCb computing group.

The ECS framework and many of its components are a deliverable of the JCOP project in which LHCb is collaborating. Other basic utility packages, distributed as part of the ECS framework, such as FSM toolkits are ultimately a JCOP responsibility, even though the original author/institute might keep the responsibility, such as is the case for RAL and SMI++. The software around the SCADA system and the ECS framework, such as configuration software, data-taking control, are under the responsibility of CERN/EP. Applications based on these toolkits will be the responsibility of the interested parties (sub-detectors, online team).

For the low-level software, such as drivers for equipment, in principle the rule prevails that whoever builds hardware, also has to provide the software controlling it. For a lot of equipment, the manufacturer will provide this software.

For the low-level software, such as drivers for equipment, in principle the rule prevails that whoever builds hardware, also has to provide the software controlling it. For a lot of equipment, the manufacturer will provide this software.

All software for the support of the HLT, calibration database, etc. will make use of the data processing framework Gaudi [53] and tools around it. These are the responsibility of the LHCb computing group and will be described in the Computing TDR.

5.3.2. Hardware

Table 23 summarizes the responsibilities for the different hardware components. The following should be noted:

- CERN will evaluate the different implementations of the CC-PCs and will do the procurement.
- The SPECs hardware is entirely a LAL/Orsay responsibility and is provided as a service to LHCb.
- The ELMB is an Atlas product and the LHCb muon group is a customer.
- CAN Master interfaces will be chosen in collaboration with the JCOP project and CERN-IT/CO.

Table 23 BREAKDOWN OF RESPONSIBILITIES FOR THE PROVISION OF THE ONLINE HARDWARE.

Component	Design/ Production	Coordination/ Supervision	Installation/ Commissioning
Readout Supervisor, TFC Switches, ORs	Warsaw	CERN-LBC	CERN-LBC
TTC Equipment	RD12	CERN-LBC	CERN-LBC
FEM/RU, Readout Network, SFC	Industry	CERN-LBC	CERN-LBC
Farm CPU	Industry	CERN-LBC CERN/IT	CERN-LBC
Computing Infrastructure	Industry	CERN-LBC	CERN-LBC
Credit-Card PC	Industry/Genoa	CERN-LBC	CERN-LBC Sub-detectors
SPECS Master & Slaves	LAL/Orsay	LAL/Orsay	CERN-LBC Sub-detectors
ELMB	Atlas/ LHCb-Muon	LHCb-Muon	CERN-LBC LHCb-Muon
Controls Switches	Industry	-	CERN-LBC
Controls PCs	Industry	-	CERN-LBC
Other (DCS) equipment	Industry	CERN-LBC Sub-detectors	CERN-LBC Sub-detectors

Appendix A FPGA-Based FEM/RU R&D

The FPGA-based Readout Unit (RU) [24] was designed as the input stage to the readout network of the LHCb data acquisition [25] and L1-VELO topology trigger [26] systems, respectively. Figure 48 shows a sketch of the hardware architecture of the FPGA-based Readout Unit.

It performs sub-vent building from up to 16 custom S-link inputs towards a commercial readout network via a PCI interface card. For output to custom links, as required in datalink multiplexer applications, an output S-link transmitter interface is alternatively available. The baseline readout network for the RU is Gbit-Ethernet for the DAQ system [27] and SCI shared memory network for the L1-VELO system [28]. New technologies, such as 10Gbit Ethernet or Infiniband may be used as far as suitable PCI interfaces and Linux device drivers will become available. The two baseline RU modes of operation are:

- front-end link-multiplexer with N S-Link to single-S-Link, and
- event-builder interface with quad Slink-to-PCI network interface.

Incoming event fragments belonging to the same event-tag are derandomised, buffered and assembled into single sub-events. Following a push-through scheme with intermediate buffering and sub-event assembly, new sub-events are retransmitted in the same order to the output network. Destination address allocation and synchronization protocols can be implemented either via the bi-directional network interface or via a custom link available for output traffic scheduling.

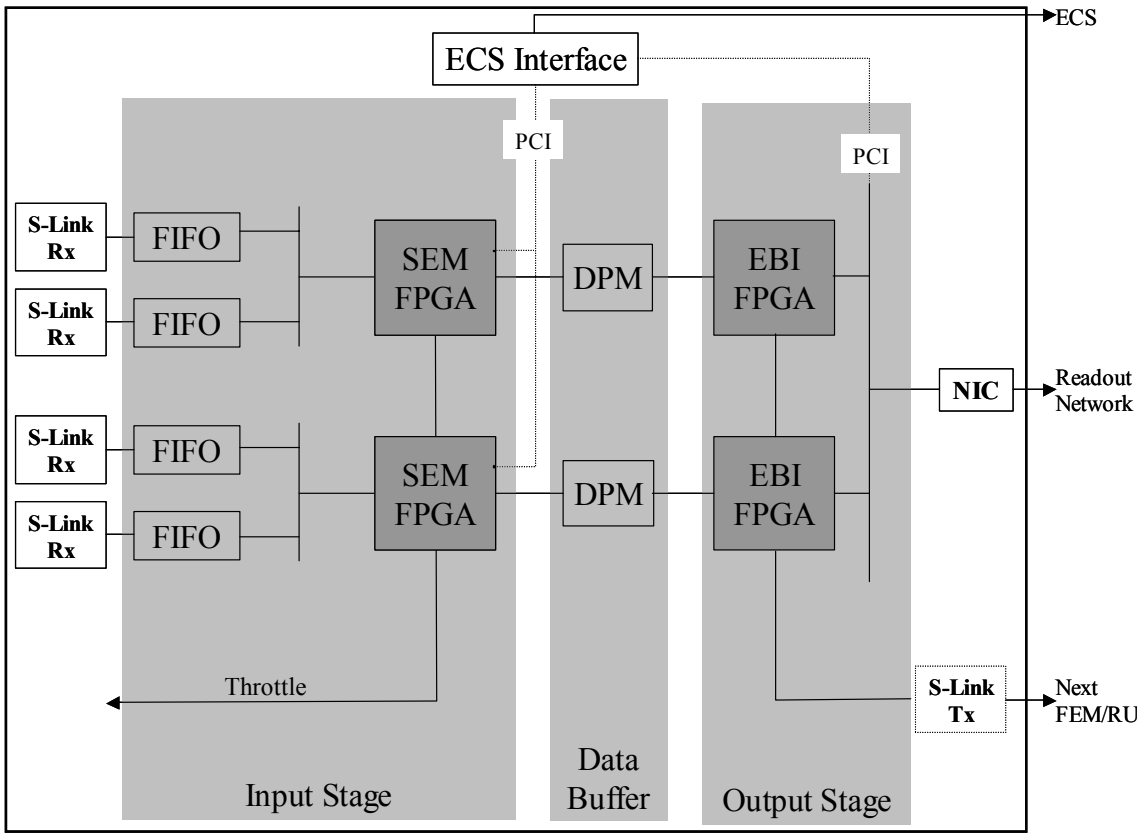

Figure 48 Hardware Architecture of the FPGA-based Readout Unit. SEM stands for Sub-Event Merging and EBI stands for Event Building Interface. The S-Link transmitter and the NIC are mutually exclusive.

The sub-event building process is based on the matching of equal event identifiers in the headers of the low-overhead Sub-event Transport Format (STF) [29]. This format was optimised for pipelined hardware state machines and failsafe transmission. Embedded in 4 words of header and trailer, STF contains fields for link parity, different data types, fast error tagging and redundancy for error detection. Input event sizes of 64 times a nominal 0.5 kByte block size can be transmitted within an 80 MByte/s bandwidth envelope. The dual-port event buffer can accumulate up to 1000 event fragments of 2 kByte. Programmable logic is used both in the input and output stages, allowing for flexibility in the scope of applications and their variants which are designed using high-level-language simulation and synthesis tools. A remote re-configuration of applications is almost instantaneously possible via the networked PCI host card resident on the RU's PCI bus.

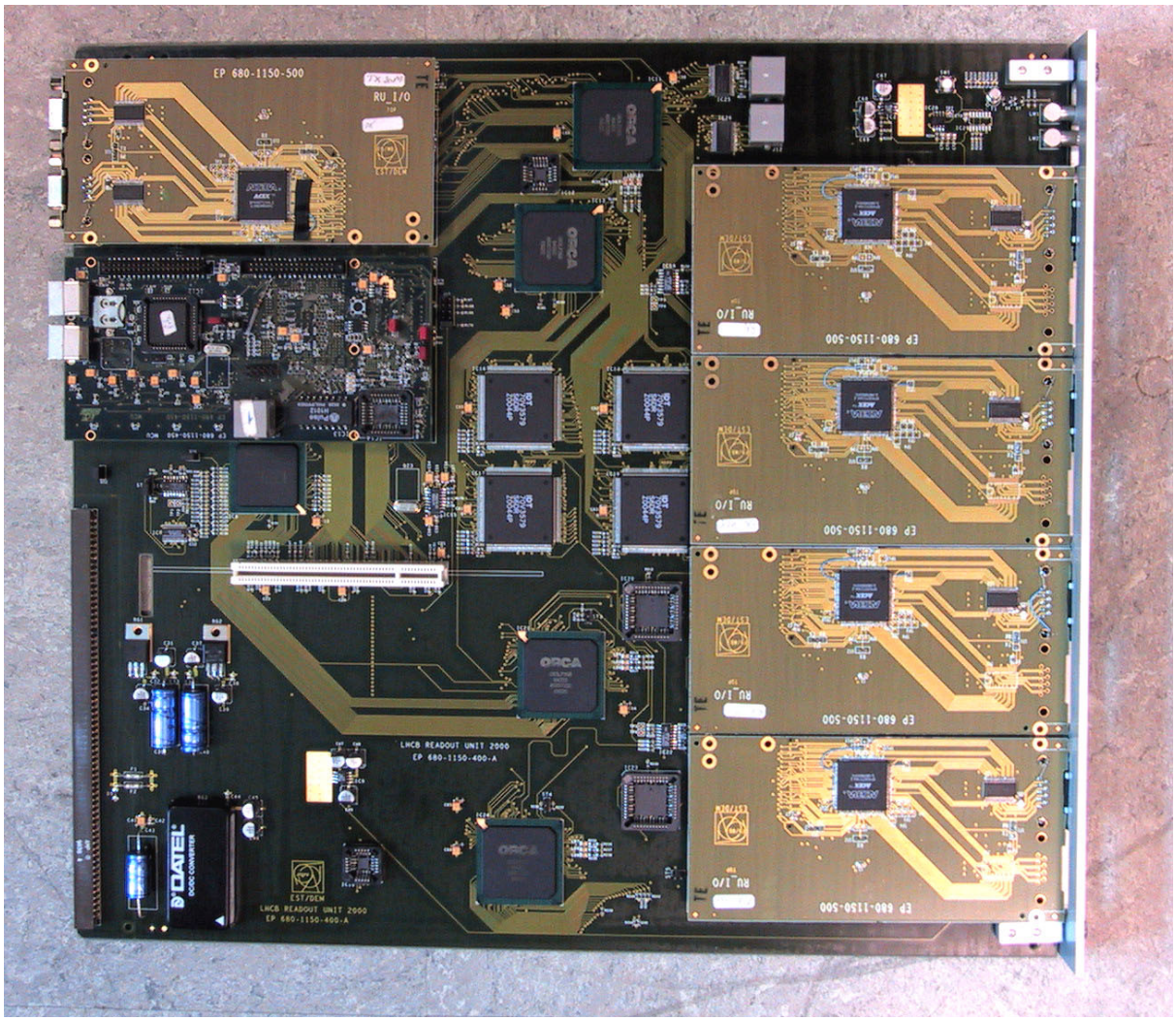

Figure 49 Photograph of a prototype board of the FPGA-based Readout Unit. The board is a 9Ux400 mm formfactor based on Fastbus mechanics. It uses only power from Fastbus, but does not use the bus itself.

All FPGA chips are interconnected via three on-board PCI bus segments, which, apart from FPGA configuration, also serve for remote access to control registers and to data buffers. The PCI segments on the RU output are 64 bit wide and mainly dedicated to make the maximum PCI bandwidth of 1/2 GByte/s available for a tandem PCI master mode [30] as required for the L1-VELO application. The root segment of the PCI bus is hosted by a networked microprocessor PMC mezzanine card that runs a diskless Linux operating system. The Monitoring and Control Unit (MCU) with a 33 MHz PCI bus master was built [31] by the RU design team, since commercial equivalents with 66 MHz PCI bus clock only became available later. Apart from the standard ECS

control tasks, the MCU is needed to initialise any specific Network Interface Cards (NIC) as a PCI device.

In DAQ or Multiplexer running mode, trigger rates up to 100 kHz are supported within an 80 MByte/s throughput envelope. Six RU modules have been produced as 9U, 10 layer PCB boards (see Figure 49), using Fastbus crates and racks from the LEP experiments as a convenient power and cooling framework. Four Slink mezzanines of any compatible link technology (with up to 4 link inputs each) can be inserted in the front panel space. On the rear side, the RU module carries two PMC mezzanines: the networked MCU and the output link card, i.e. either a NIC or Slink. One RU has also been successfully produced in halogen-free PCB technology as a first test at CERN for halogen-free PCB production, which will eventually be mandatory, by European regulations. The FPGA-based RU is a tested and reproducible 9U module and includes co-designs like the Slink I/O card for data transmission over up to 25 m of standard network cables, Slink pattern generator cards to produce STF formatted test data and a networked PMC card. A PC-based RU exerciser using PCI-to-Slink cards will complete the FPGA based Readout Unit Project, and allow for error-integrity and performance testing of Readout Unit systems with any configurable data sets and tuneable trigger rates.

Appendix B Event Building R&D Studies

This appendix gives an overview over the studies that have been carried out in the area of event building.

The first two sections will deal with candidate technologies for implementing the Readout Network, while the last two will describe the investigations concerning the network topologies and the possibilities for performing the final event building.

B.1 Myrinet Studies

Myrinet [54] is a network technology mainly used for implementing low-latency communications between computers. It features

- 1.28 Gb/s (2.0 Gb/s in Myrinet 2000) point-to-point link speed
- Full Duplex links with Xon/Xoff Flow Control
- Programmable NICs
- Non-Blocking cross-bar switch chip (up to 16 ports)

The main attraction of Myrinet is the very low cost of the individual switch port, compared to e.g. Gigabit Ethernet.

We have measured the performance of Myrinet, by connecting two PCs with Myrinet cards together and found that the specifications were met.

Subsequently we performed simulations with large network configurations (up to 128x128 ports, Banyan topology) [55]. The main outcome of these simulations is, that Myrinet, due to the lack of buffering in the switches, suffers from Head-of-Line blocking. Local congestion somewhere in the composite switching network will prevent transfers out of the NIC, even if only along the path within the network a congested internal connection is used. This leads to an unfavourable scaling behaviour as can be seen in Figure 50. The figure shows the efficiency, i.e. the maximum achievable throughput relative to the nominal installed bandwidth for different sizes of the (composite) network in a Banyan topology. The basic building block is an 8x8 switch, out of which networks up to 128x128 ports were built. The two curves represent the results with and without FIFO buffers between layers of switches.

This imperfect scaling behaviour could only be corrected by adding intermediate buffers in form of FIFOs. This would imply designing and building custom hardware. Myrinet will only be a backup solution in case an implementation of the event building sub-system with GbEthernet should face insurmountable problems.

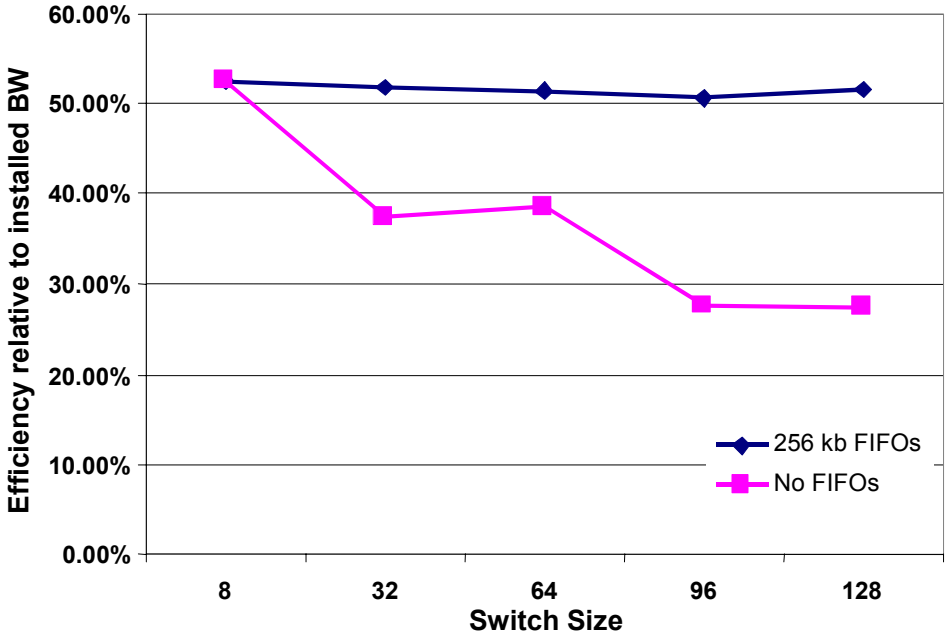

Figure 50 Simulation results of various configurations of Myrinet networks.

B.2 Gigabit Ethernet Studies

The reasons for adopting Gigabit Ethernet as the basic network technology for the Readout Network (RN) in LHCb have been explained in Section 4.2.

Gigabit Ethernet is a connectionless, full-duplex, point-to-point protocol [56] and [57]. The RN is therefore implemented as N x M fully connected switching network, where N is the number of RU (data sources) and M the number of sub-farm controllers (data sinks).

The basic architecture of the LHCb DAQ system is a pure push-through protocol. Each source sends as soon as it can. Data flows asynchronously from layer to layer. There is no lateral communication and also no central "Event Manager", acting as an orchestrating entity. The system is therefore almost perfectly scalable laterally at the top (RUs) and bottom (SFCs). The one exception is the switch itself. This is one, single, central element, whose performance and behaviour determines critically the performance of the system as a whole.

A Gigabit Ethernet switch of the required size is an expensive high-end device, on which the following requirements are put:

- It must provide a non-blocking, wire-speed switching fabric. This is fulfilled by basically all commercially available switches.
- It must be capable of coping with the specific traffic pattern imposed by our architecture. Usually all fragments of an event will arrive in a rather short time interval. Sufficient buffering or some sort of flow control must be in place to avoid packet loss.
- Packet loss must be limited to a very low rate, say 10^{-8}. If it happens at all, this must be logged in the data.

Whether a switch fulfils these specific properties is difficult to judge from the information, which is usually available for commercial products. They depend strongly on the architecture of the switching fabric (packet switching, cross bar), the speed of the back plane, the buffer-size and architecture (output-, input- or central queuing) and the firmware.

To investigate the suitability of a given switch and extract necessary parameters for input to the simulation of the Readout Network (c.f. Section 4.4.2) a test set-up was devised. A simple model for a switch was devised as follows: A switch consists of a backplane, with a certain speed and associated latency and several line-cards ("blades") equipped with a number of ports. The ports have a certain amount of possibly shared input- and output- buffer memory. Switching of packets is either done directly on the line-card with a certain latency, or via the back plane, with a different latency.

The parameters in this model are the latencies for switching a byte, the amount of memory and whether the switch is non-blocking. The measurements of these parameters were done by sending packets through a switch from NIC to NIC and back. These so-called "ping-pong" measurements use the internal clock of the sending NIC to measure the time. The time spent in producing, transferring and reflecting a frame in the NIC, can be measured by connecting the NICs back-to-back.

Measurements have been performed as a function of the packet-size for a reasonably large amount of packets (several millions per measurement point).

To measure the buffer sizes, ports were blocked by feeding them Ethernet flow-control packets, as they are described in ref. [56]. A standard compliant switch will then stop sending to that station, and hence must buffer packets directed to that station. From the amount of packets lost one can get an estimate of the buffer size available for storage. In addition, one can try to fill up the buffers completely and then try to send to another output port. This working amounts to the switch being non-blocking.

The method described is fairly general and applicable to any switch. The one switch we tested extensively up to now is the Foundry Fast Iron Gigabit Ethernet Switch [58].

The Foundry Fast Iron comes in various sizes, it has 8 port line cards, and the largest model can house 15 of them. The one at our disposal had 8 fully equipped line cards. Frames were again generated and evaluated using a dedicated firmware in the Tigon 2 based NICs.

Figure 51 shows the latency (through the same line-card) for frame sizes between 350 and 600 Bytes. The right hand scale shows the size of the residuals with respect to the model function described below. The packetisation of the switch is visible in the small steps in the latency for every 64 bytes.

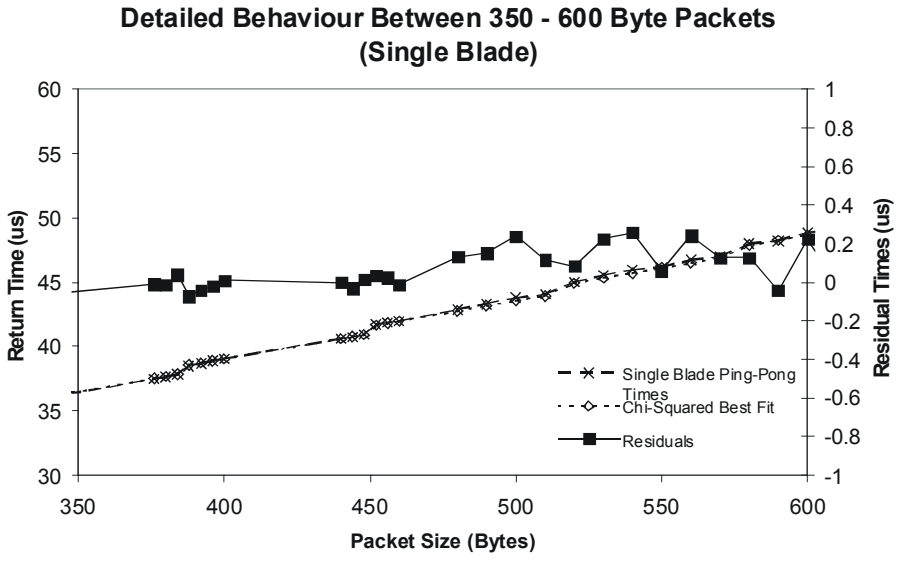

Figure 51 Close-up plot of the latency measurement across the switch.

The latency has been characterised by a function of the following form:

$$y(x) = a + b \cdot x + \mathrm{int}\left(\frac{x-1}{c}\right) \cdot d$$

Where:

x is the Frame size in Bytes

a is the Constant overhead (due to cabling, turnaround times, minimum switching time etc)

b is the Latency per Byte (overhead time for each additional "useful" data Byte within a packet)

c is the packetisation quantum

d is the additional time spent at each packetisation boundary

Our findings for this switch are summarised in Table 24.

Table 24 SUMMARY OF THE FITTED PARAMETER VALUES FOR THE CASES OF SINGLE-BLADE AND ACROSS-BLADE TRANSFERS. THE FITS WERE OBTAINED OVER THE FULL RANGE OF PACKET SIZES.

Condition	Parameter Set
Single Blade (Port-to-Port)	a = 0.537 ms b = 0.38 ms/Byte c = 64 Bytes d = 0.035 ms/packet
Across Blades (Port-to-Port)	a = 1.338 ms b = 0.041 ms/Byte c = 64 Bytes d = 0.0362 ms/packet
Buffer Memory	2 Mbytes shared between 8 ports
Flow Control	respected, but never issued

This particular switch does not seem to be suitable for our system, mostly due to its behaviour upon port blockage: Our current finding is that it first drops some frames and only then starts buffering frames, up to the maximum buffer limit. Such behaviour would not be acceptable in our system. Many more details about the method and the results can be found in ref [59]. See Section 4.4.4 for a discussion of the future strategy.

B.3 "Smart" NIC Studies

In Section 4.5, "Smart" NICs have been presented as the baseline implementation for the final event building in the Readout Network (RN). By a "smart" network interface controller, we mean here one, which is freely programmable, i.e. one that contains at least one general purposed CPU. This allows putting the bookkeeping and error-checking code involved in the event building code being implemented directly in the NIC.

The task for this smart NIC, like for any final event-builder implementation in the LHCb DAQ system consists of receiving fragments from all the sources in the system at 40 kHz. These have to be concatenated and transferred as one contiguous block.

The advantage in using the NIC in that way is that one can offload considerably the host CPU, in number of interrupts and also in memory-to-memory copying. This is so because the event building process is in principle the set-up of a cleverly chained DMA transfer of all fragments in one go.

This needs considerable hardware support and buffer memory on the NIC. Most Gigabit NICs offer some buffer memory and the possibility to coalesce packet transfers to reduce the IRQ rate on the host CPU, thus increasing overall system performance.

The algorithm used waits until either it has received all the fragments belonging to one event, or a time-out has been reached. In the latter case, an error is flagged. Then the event is shipped as soon as possible to the Sub-farm Controller. The timeout period is determined by available buffer-space only.

The actual implementation has been done on a NIC based on the Tigon 2 ASIC. This chip comprises 2 MIPS 4000 cores, with 16(8) kB of on-chip scratch pad memory (i.e. addressable memory at running at clock speed of 88 MHz), a Gigabit Ethernet MAC, and an interface to an external SDRAM buffer memory (512 kB in our case). The code was optimised to take advantage of the internal architecture, meaning, for example, to keep counters in scratch-pad memory as much as possible. The pointers are then fed to the scatter/gather-capable DMA engine for transfer over DMA.

In the end, the performance illustrated in Figure 52 could be achieved. The results are consistent with a constant overhead of ~11 μs per incoming data fragment. This is sufficient for the base-line design of 40 kHz. With some improvements, especially in the hardware of the NIC (faster, more memory) fragment rates of 100 kHz can be easily handled as well. The average load on the PCI bus will be 40 MB/sec, which should not be problematic in a server-like PC, such as the SFC.

Many more details on this and alternative algorithms, the test-bench set-up, the software tools developed and more detailed results can be found in references [60] and [61].

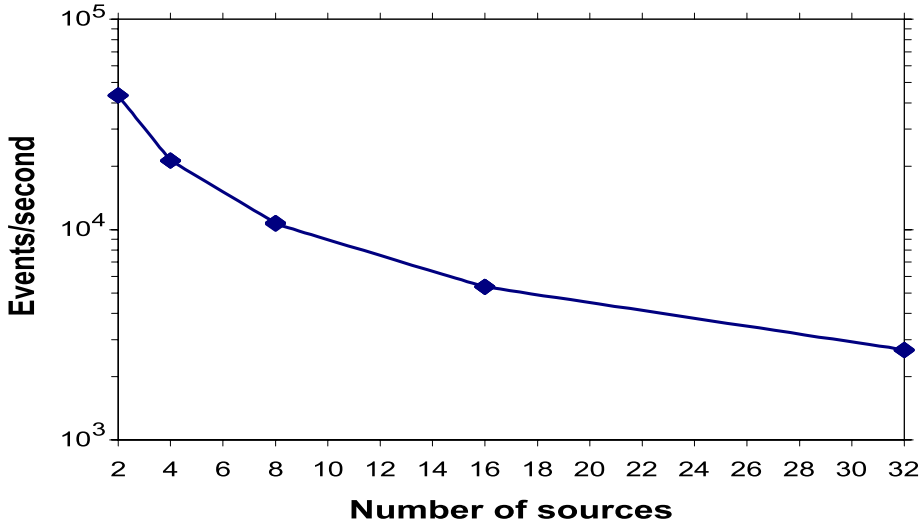

Figure 52 Performance of event building in smart NICs.

B.4 Network Topology Studies

B.4.1 Load

By load, we mean the fraction of the available installed bandwidth that is used to transfer data. We try to determine the "maximum possible load", L_{max}, which still allows a correct functioning of the system.

We distinguish the load on a single link to or from the network from the aggregate load on the network. To determine the latter, simulation is required.

Load on a single link

The maximum possible load on a single link to or from the network depends on the characteristics of the device attached to the link. In general, it is a processor that connects to the link via a NIC (Network Interface Card). The submission or reception of a packet to or from the network has a time cost that we call τ_{ov}. This overhead time is due to several operations required to handle the packet in the NIC and/or in the processor, such as protocol operations, interrupt handling, etc.

The transfer time τ_s of a packet of size s over a link of bandwidth B is B x s. Assuming that packets can be transferred concurrently with the packet handling operations, the maximum achievable load L_{max} is:

$$L_{max} = \frac{s}{Max(\tau_{ov}, \tau_s)}$$

If the overhead time τ_{ov} is independent of the packet size, the behaviour of L_{max} as a function of s is given in Figure 53.

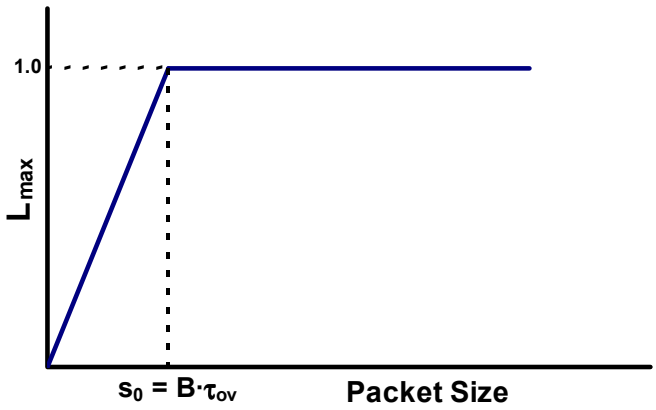

Figure 53 Maximum achievable load on a single link as a function of the packet size.

With the same assumptions as above, the maximum frequency at which packets can be transferred on a link is:

$$f_{max} = \frac{1}{Max(\tau_{ov}, \tau_s)}$$

Figure 54 shows how f_{max} varies with s

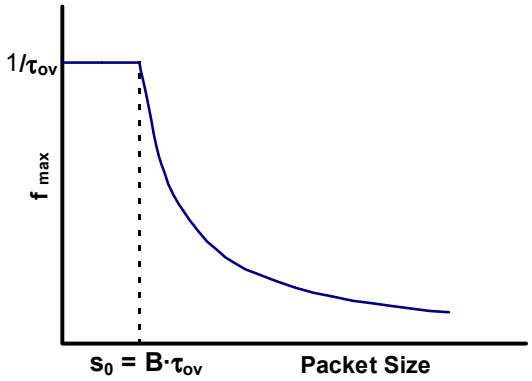

Figure 54 Maximum frequency on a single link as a function of the packet size.

The value τ_{ov} determines the packet size s_0 above which the full link bandwidth can be exploited. For highly optimised implementation of the packet handling and in the absence of a secure transport protocol, τ_{ov} is of the order of 1 or 2 μsec, corresponding to values of s_0 of 125 or 250 bytes on a 1 Gbit/s link. If a secure transport protocol like TCP/IP is required, the value of τ_{ov} can be one or two orders of magnitude higher [64], although faster implementations based on intelligent NICs are proposed [65]. For the implementation of an event-builder with a network entirely dedicated to this task, we have good reasons to believe that a non-secure transport protocol will be adequate, as long as the maximum permissible aggregate load is not exceeded.

Load on the Event Building Network

The event building network will be a switched network in order to cope with the very high bandwidth [66].

An average load factor of the network is obtained by adding all link loads and dividing by the number of links. In the case of an event-builder, the individual loads are not independent: they are determined by the event trigger frequency, being the same for all links, multiplied by the average event fragment size of the link.

The load on the network cannot be as high as the maximum load achievable on a single link. This is due to packets contending, within the network, for the same links. This contention is normally resolved by storing temporarily the packets in internal buffers. Thus, due to contention, the network does not behave as well as a fully parallel system. Event building traffic is even worse as it tends to concentrate the traffic and create more contention. However, an appropriate buffering scheme (such as "output queuing") and the fact that the destination changes for every event lead to a well-distributed load, as will be shown later.

It is wise to dimension the network with some safety factor, instead of relying on the maximum possible load, in view of possible growing demands and to avoid instabilities due to simple control systems like throttling.

The relationship between load and frequency, combining the 2 functions described previously, is useful to determine the dimension of the event building network. It is displayed in Figure 55 for several values of the packet size s.

As an example, assuming an event rate of 40 kHz, the packet size per link should be of the order of 1.5 kB if one wishes to limit the load to 50%. For an event size of 100 kB, some 65 - 70 ports are needed.

Simulation is required to analyse in detail the load issue. It requires knowledge of the strategy adopted by the switch manufacturer to cope with contention. In the next section, an overview of the basic switching strategies is presented.

Figure 56 shows qualitatively that the performance of a switching network has some maximum value of the load beyond which one can expect that data is lost or transfer is blocked.

This limit is well below 1 for networks based on switches implementing input queuing. Before reaching this cut-off limit, there is some zone in which the functioning of the network is likely to experience momentaneous blockings or data losses due to fluctuations in the traffic. Finally, there is a zone of lower load where one can expect that the switching network has a stable and safe mode of operation. One task of simulation is to determine those values.

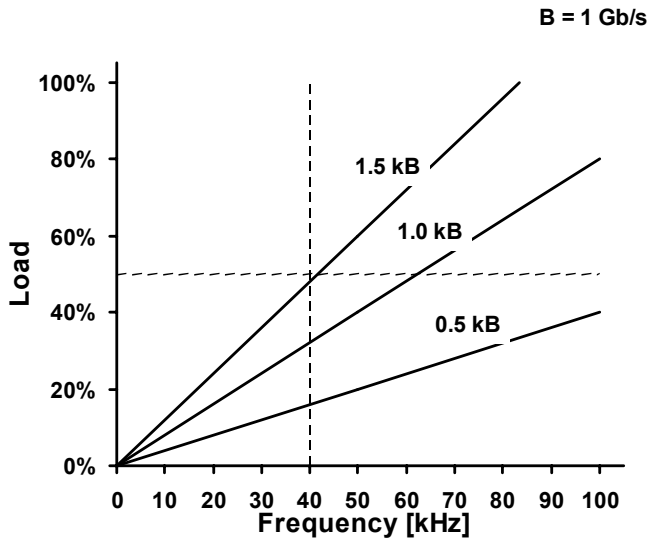

Figure 55 Load on a single link as a function of the packet frequency for several values of $s > s_0$.

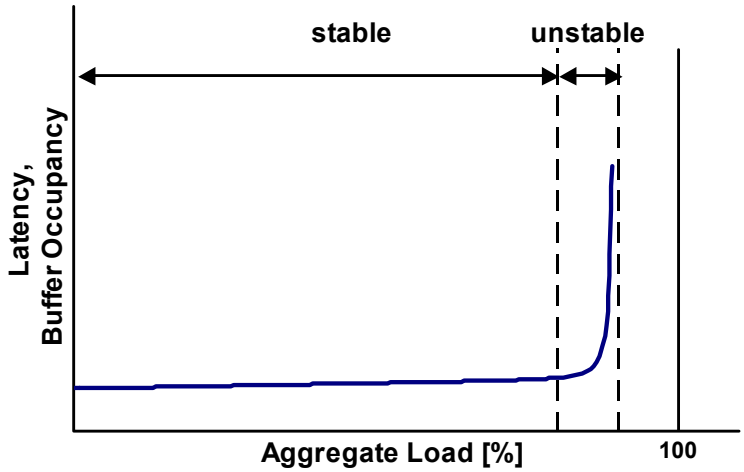

Figure 56 Qualitative sketch of the behaviour of a switching network as a function of the load.

B.4.2 Switching Strategies

A switch is said to be *non-blocking* when the path between any pair of input and output ports cannot be blocked by a transfer on a different path. This property is relatively easy to implement, however the implementation costs grow faster than linearly with the number of ports. It is said to be non-scalable.

However, *contention* is likely to occur, even in a non-blocking switch, whenever two or more input ports want to transfer data to the same output port.

Large networks can be built by interconnecting switches. There are many classical techniques (e.g. Banyan networks [67], Clos networks, etc [68]. In the simple interconnection techniques, the non-blocking property of the switching elements is not conserved. Figure 57 illustrates this fact on a simple example: even if the 2x2 switches are non-blocking, it is obvious that, for instance, a transfer between ports 1 and 5 blocks a possible simultaneous transfer between ports 2 and 6.

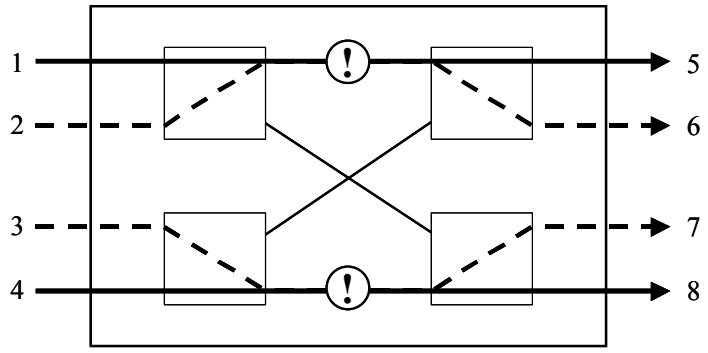

Figure 57 An 8 port switching network (4x4) obtained by interconnecting 4 non-blocking switches of 4 ports (2x2).

It can be noted that, in this case, blocking is due to contention in the non-blocking switching element.

There exist interconnection schemes that are non-blocking but they require substantially more switching elements and links than the simplest schemes (Banyan).

Contention Avoidance Schemes

It is not acceptable that, in case of contention, one or more packets are dropped. Instead, they must be stored. The location of storage can be at various places:

- at the input of the path: *input queuing*
- at the output of the path: *output queuing*
- centrally within the switching element, in a shared memory: *central queuing*

Input Queuing

An input port stores the data to be transmitted in a FIFO. If the packet at the FIFO's head cannot be transferred due to contention, the port refrains from transmitting until the path is free. This clearly lowers the link occupancy and reduces the effective load. It is well possible that other packets waiting in the FIFO could be transferred if their destination were on a free path, however the FIFO structure prevents them from bypassing the first packet. This is known as *head of line blocking* and leads to rather poor switch performances. In the case of random traffic and fixed size packets, it can be shown [69] that, for a non-blocking NxN switch, L_{max} has the asymptotic value, for large N, given by:

$$L_{\max} \to 2 - \sqrt{2} \approx 0.58$$

This load value is still lower for blocking switching networks built by interconnecting switching elements that do not implement any storage and where the input buffer is on the boundary of the network (circuit switched networks).

The Myrinet technology offers very low cost non-blocking switches that can be interconnected to build large circuit switched networks with input queuing (see B.1).

This load performance of circuit switched networks can be improved if one pays the price of implementing a traffic control system ensuring that contention never occurs. One such system is the barrel shifter and has been tested by CMS in the case of Myrinet [70].

Output Queuing

A much better solution is to let all data go through the switch, even in case of contention, and to organize the queuing at the output ports. One realizes in this case that the load can be close to 1, provided that the contention is fairly distributed over all output ports.

The cost to pay is that the bandwidth of the shared links is a multiple of the port bandwidth. In the simple example of Figure 56, all internal connections must offer twice the bandwidth of the external links. This can be implemented with multi-path connections or with faster links. Usually the output queuing systems transfer fixed size packets. Depending on the external link standard, a local packetisation with segmentation and reassembly has to be provided (e.g. Ethernet switches that segment the data in fixed size cells of 64 bytes).

Central Queuing

A drawback of the output queuing technology is that all the output buffers must be dimensioned to cope with the worst possible case, thus leading to a poor global occupancy of expensive fast memories. A better solution is to use a shared memory with dynamic allocation of space to the output ports. An example of such an implementation is the Prizma switch from IBM [71].

Switches that implement output or central queuing are preferred when high aggregate loads are expected. They are also more costly than switches with input queuing since they need faster memory.

B.4.3 Traffic Shaping

Even the best switching scheme breaks down if the contention is not fairly distributed between the output ports. This is the case for event building if very large events are generated that maintain the contention on one buffer during a time sufficient to overflow the memory.

A solution to this problem is to impose a constant bit rate on all connections between all source-destination pairs. This may be done by implementing, in every source, N queues, one per destination (N being the number of destinations). The data is segmented in fixed size packets. The source scans the queues in a round-robin fashion, sending each time 1 fixed size cell to the corresponding destination (or just stalling for the same time interval if the queue is empty). There is some loss of throughput due to the segmentation in fixed size cells. One should also take some precaution to avoid that the all sources send packets to the same destination at the same time. This can be achieved by requiring that, at initialisation, source k starts with queue k. The probability that they reach an exact synchronization due to random time shifts should be zero.

B.4.4 Transport Protocols and Safe Data Transfer

Ethernet does not provide any transport protocol that guarantees the delivery of data packets. The only mechanism offered by the Ethernet standard (IEEE 802.3x) is the so-called Xon/Xoff, a point-to-point signal that a receiver sends to a sender in case of overflow. This signalling is obeyed between the switch ports and the attached devices. However the overflow of an buffer internal to the switch will not raise an Xoff and data will be lost.

The use of a “high level” standard transport protocol (TCP/IP being the only candidate) would guarantee the delivery of data, possibly by re-transmitting lost packets. There are several arguments against this solution:

- The τ_{ov} due to TCP/IP is too high for the requirements of high rate of small packets.

- Data loss in a privately owned local network is most probably due to buffer overflow, which itself is caused by an excessive load. Adding more traffic on top by re-transmitting data would just worsen the problem.
- In the event of a faulty component, the transport protocol will be useless, unless redundant data paths are available, which will not be the case in our system.

The event building network will be designed to fulfil the following requirements:

- the load on the network shall be well within the "safe region" (Figure 56) for the specified data flow with its "normal" statistical fluctuations,
- data losses must be detected and must be signalled,
- the probability for data losses under the "normal conditions" has to be very low and unbiased,
- abnormal conditions (traffic exceeding the "normal conditions", component failures) must be detected and signalled,
- the previous conditions being fulfilled, no mechanism will be provided to recover from data losses.

Appendix C Test-Beam Activities

The activities described here cover LHCb testbeam operation in 2001 using for most of the sub-detectors a new data acquisition system. In order to improve the user interface on the run control side, the user interfaces of CASCADE [72] have been replaced by a CASCADE stage's[1] control system based on the new CERN standard PVSS[2].

C.1 The LHCb Testbeam Computing Setup

In 2001 in total four testbeam areas have been used by LHCb, spread over two experimental halls on CERN's Meyrin site. The central computing infrastructure is set up in EHW1 in a barrack assigned to the main testbeam area. This central infrastructure consists of

- a disk server running Linux and providing disk space, BOOTP, and TFTP services for the front-end processors,
- a central run control PC running Linux and the PVSS system for all testbeam activities,
- a solid "private" LAN, connecting all processors, terminals, and servers, within this specific area, and
- several terminals (Windows 2000, Windows NT, HPUX, and Linux) as well as
- several front-end processors, especially RIOs, VME boards with embedded PowerPCs.

C.2 PVSS Control for CASCADE Stages

During the shutdown 2000/2001 the change from a completely CASCADE-based system to a PVSS run control has been performed.

The main reasons for the upgrade were to provide a more flexible system and better support to the users, the use of CERN supported hardware (PCs), and to gain experience with the new SCADA[3] system PVSS used by all LHC experiments. Another reason was to use the testbeam environment as a realistic area for establishing a first proof of concept for our integrated approach to controls, which is one of the corner pieces of the LHCb online system.

This change of software was done in two steps. Firstly, the functionality provided by CASCADE was implemented and comprehensive panels for the users designed. This system is used for the present testbeam activities. In a second approach, the run control and several other devices, e.g. the control of a moveable platform for the calorimeters and a display of accelerator data, have been included inside a hierarchical structure.

First Step

The main step was the change of the receiving end of the communication between front-end and supervisor. As illustrated in Figure 58, the front-end software together with its communication package has not been changed. However, the software formerly running on HPs has been entirely replaced by PVSS and a communications package implemented as a PVSS API. Apart from that, all

[1] In the CASCADE definition, a "stage" is a process running on a front-end processor. There exist e. g. stages for controlling the DAQ front-end, disk recording, and message processing.
[2] **P**rozeß**V**isualisierungs- und **S**teuerungs**S**ystem developed by ETM
[3] **S**upervisory **C**ontrols **A**nd **D**ata **A**cquisition

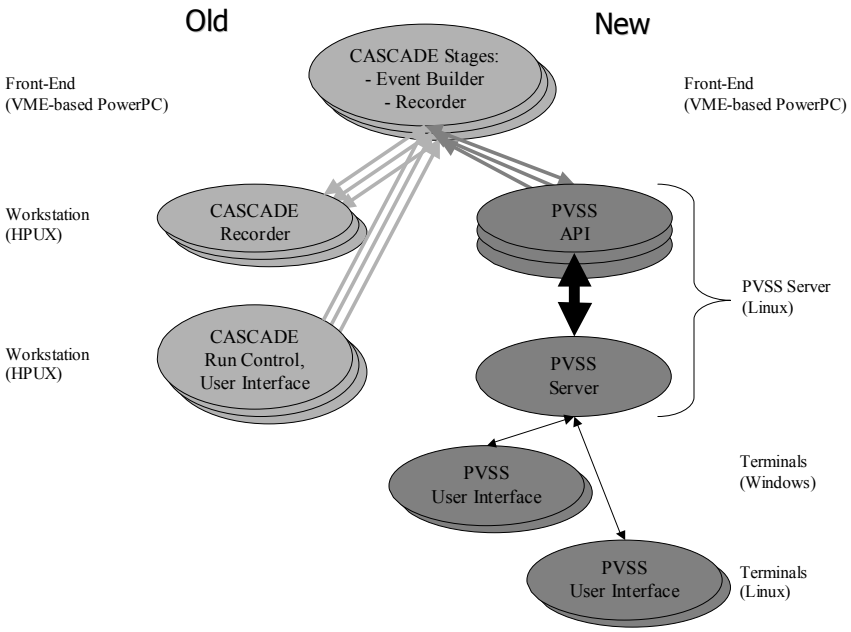

Figure 58 Changes to the Run Control System during the Shutdown 2000/2001.

users now run using a central SCADA system in contrast to the different run controllers and disk recorders of the CASCADE approach.

This step brought to the testbeam users the change from a mostly command line driven system to comprehensive panels. All user functions can now be controlled from a single main panel (see Figure 59) that allows getting an overview of the complete system. The state of the connected processes – the CASCADE stages – is shown in a common colour coding, all possible actions are implemented as buttons. For configuration, several sub-menus are available to replace the long cryptic command lines.

Experience of one year of testbeam data taking, during which the user interface has somewhat evolved, shows that the users are happy with this type of control system. Especially new untrained users tend to understand much faster, when comprehensive user guidance is available.

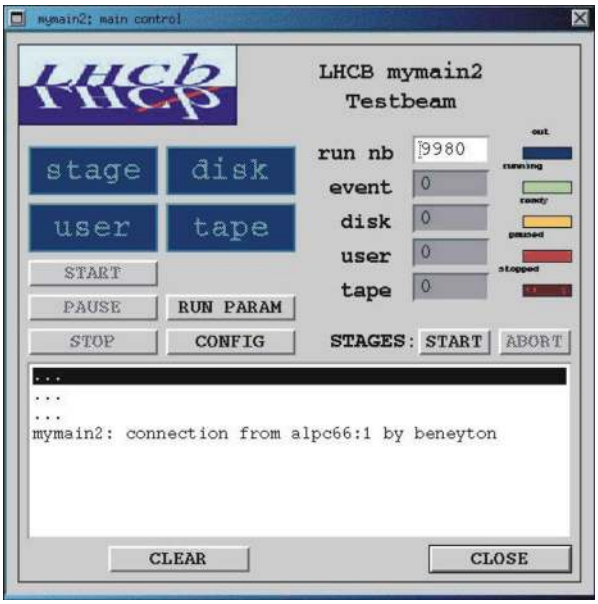

Figure 59 Main run control panel of the CASCADE stage's control.

Second Step

After having gained experience with the basic PVSS functionality and incorporating users comments, a second step towards the use of LHCb ECS tools was taken. Here, PVSS was of course left as the SCADA software, but the organisation of data inside PVSS has been changed to a hierarchical system, the LHCb framework. This resulted also in a change of the user interface – see Figure 60 for the new main panel. On the other hand, several new components have been included into the same user interface.

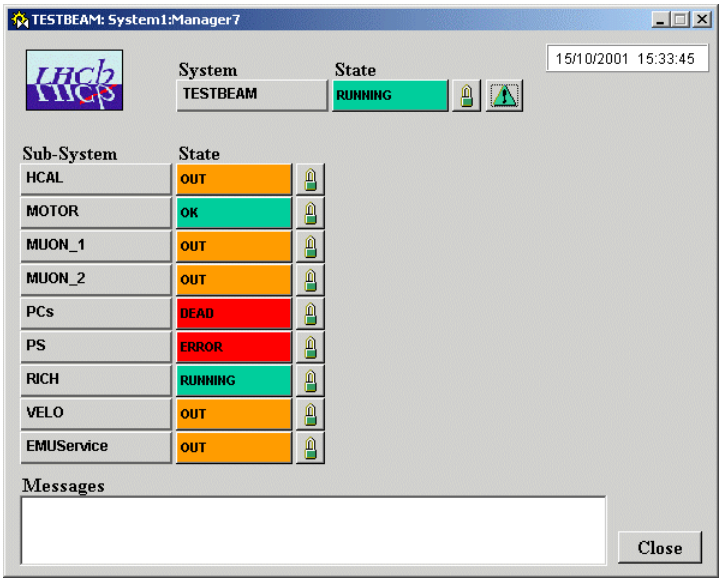

Figure 60 Main system panel of the hierarchical approach.

One of these components, the motor steering of a platform used by the calorimeter testbeam to scan complete modules without the need of personnel accessing the beamline to relocate these modules (see Figure 61). Another new device is the contact to the beam related data from the PS, had already been accessible in the prior version, but now the beamlines of the PS are fully integrated as devices of the testbeam controls (see Figure 62). Combining this functionality with the readout control makes a completely automated scan of calorimeter modules possible.

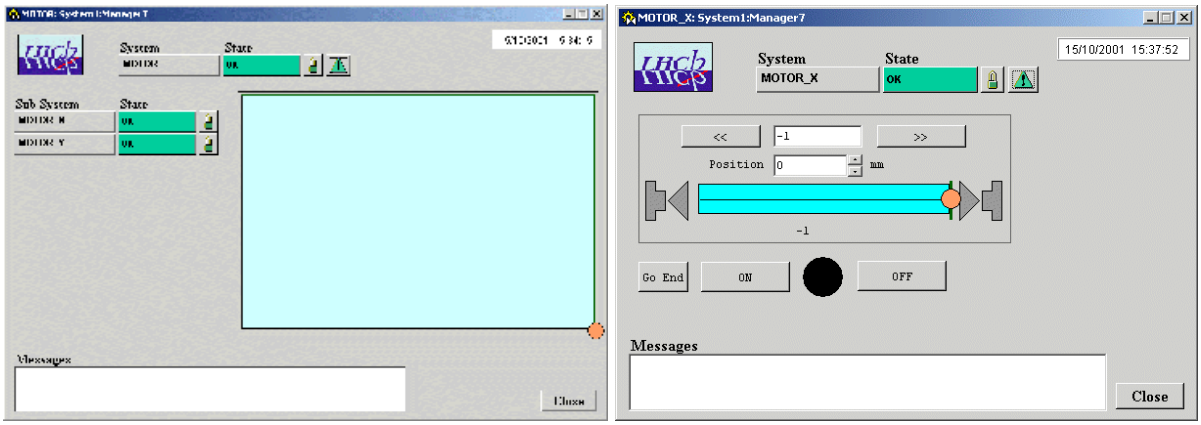

Figure 61 Panels for Controlling the Calorimeter Platform.

Central Data Recording

The recording of testbeam data has been centralized for all users. All runs taken with the new system are automatically copied to CASTOR into an area of the LHCb storage space. This area

additionally was set up to automatically create two copies of each file on separate tapes. Furthermore, all runs are entered into a run database for easy retrieval.

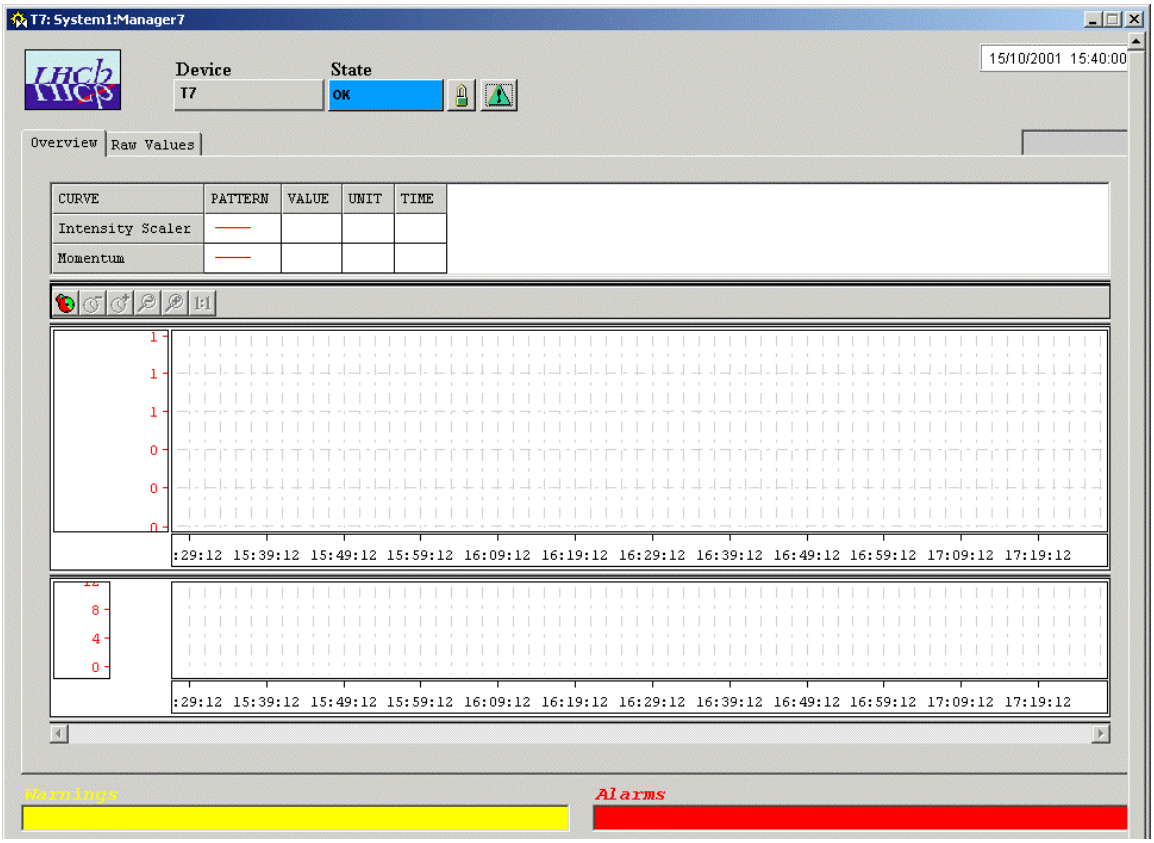

Figure 62 Integration of accelerator data into the experiment control.

Accounting for the fact that not all testbeam set-ups are realized with the supported system, a command line-interface is provided to enter files into the testbeam storage space and the run database.

User Information and Interactive System Support

Apart from the above described user interfaces to the control system, a web-based user information system with some tools has been set up [73]. These pages allow access to all log files produced by online software on the central testbeam machines, the run database, log files of beam related data from the accelerators, all parameters needed to set up a sub-detector testbeam, and the documentation of selected issues.

In addition to these presentation tools for testbeam-related information, several support procedures have been made available interactively via this website. These features include restarting services on the testbeam serves (e. g. NFS and BOOTP), automatic changes in the networking database in case of location changes of testbeam equipment directly from the IT/CS network database, and automatic backup and restore procedures together with file search capabilities.

Experience in 2001

The new controls software together with the web-based access and network configuration has been welcomed by all users. Especially the fact that a complete documentation is available online [73] and as a printable document [74] was a novelty in testbeam operations. The comprehensive user interface (see above) has been in production the full year and used by all shift personnel in the

testbeam. The system is now stable, both, the user interface part and the underlying software. For the moment, only the calorimeter group used the second approach because they needed the platform control, but it will be put into production next year.

References

[1] LHCb Collaboration, "LHCb: Technical Proposal", CERN-LHCC-98-004.

[2] O. Callot et al., "Experience with the ALEPH Online system", ALEPH PUB 2001-003, June 2001.

[3] A. Augustinus et al., "Designing a HEP Experiment Control System; lessons to be learned from 10 years evolution and operation of the DELPHI experiment", Proceedings of the International Conference on Computing in High Energy Physics, Padova, Italy, February 2000, p. 287-290.

[4] RD-12 Documentation [online] http://www.cern.ch/TTC/intro.html and references therein.

[5] JCOP [online] http://itcowww.cern.ch/jcop/subprojects/Framework/welcome.html .

[6] PVSS [online] http://www.pvss.com/english/index.htm .

[7] LHCb Front-End Architecture [online] http://lhcb-elec.web.cern.ch/lhcb-elec/html/architecture.htm .

[8] Conditions Database see e.g. http://wwwinfo.cern.ch/db/objectivity/docs/conditionsdb/ and references therein.

[9] V. Gibson and C. Shepherd-Themistocleous, "Investigation of fast particle identification in a generic event filter for LHCb", LHCb-2001-107.

[10] R. Jacobsson, "How can I run my Detector", LHCb 2001-140 DAQ.

[11] R. Jacobsson and B. Jost, "Timing and Fast Control", LHCb 2001-016 DAQ.

[12] C. Gaspar, R. Jacobsson and B. Jost, "Partitioning in LHCb", LHCb 2001-116 DAQ.

[13] J. Christiansen, "Requirements to the L0 front-end electronics", LHCb 2001-14.

[14] R. Jacobsson, B. Jost and Z. Guzik, "TFC Switch Specifications", LHCb 2001-018 DAQ.

[15] R. Jacobsson, B. Jost and Z. Guzik, "Readout Supervisor Design Specifications", LHCb 2001-012 DAQ.

[16] H. Ahmadi and W. E. Denzel, "A Survey of Modern High Performance Switching," *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 7, pp. 1091-1103, Sept. 1989.

[17] J. Harvey, "Computing Model – Baseline model of LHCb's distributed computing facilities," http://lhcb-comp.web.cern.ch/lhcb-comp/computingmodel/ComputingModelV3.pdf .

[18] W. Salter (ed.), "JCOP Architecture Working Group, "Framework Design Proposal," CERN-JCOP-2000-007, October 2001, [online] http://itcowww.cern.ch/jcop/subprojects/Architecture/Framework/AWGReport.pdf .

[19] Z. Guzik and R. Jacobsson, "Odin – LHCb Readout Supervisor, Technical Reference," Revision 1.4, Sep. 2001.

[20] I. Garcia and J. Christiansen, "Simulation of the L0 front-end electronics," LHCb 1999-047.

[21] P. Vasquez and J. Christiansen, "Simulation of the LHCb L1 front-end," LHCb 2001-126.

[22] R. Jacobsson, "Experience with the TFC Switch – Prototype 1," [online] http://lhcb-comp.web.cern.ch/lhcbcomp/DAQ/TFC/documents/switch_experience.pdf, June 2001.

[23] Nick Ellis and John Dawson, private communications.

[24] Readout Unit, FPGA version for link multipexers, DAQ and VELO trigger, Technical Note LHCb DAQ 2001-136 URL: http://hmuller.home.cern.ch/hmuller/RU/rurtieee.pdf .

[25] B. Jost et al. "The LHCb Trigger and Data Acquisition System," IEEE Real Time Conference June 14-18, 1999 Santa Fe.

[26] LHCb Level-1 Vertex Topology Trigger, LHCb 1999-31.

[27] J.-P.Dufey, M.Frank, F.Harris, J.Harvey, B.Jost, P.Mato, H.Müller, "The LHCb Trigger and Data Acquisition System", IEEE Trans. Nucl. Sci,. vol 47, no 2, pp. 86-90, Apr. 2000.

[28] A.Walsch, V.Lindenstruth, M. Schulz "Evaluation of SCI as a fabric for a computer based pattern recognition trigger running at 1.12 MHz", Proc. 12th IEEE NPSS Real Time Conference, Valencia pp. 11-15, June 2001.

[29] Readout Unit II, Presentation in LHCb DAQ/FE workshop, January 2001, URL http://hmuller.home.cern.ch/hmuller/RU/RU+Jan2001.PDF .

[30] A. Walsch, "A Hardware/Software Triggered DMA Engine," LHCb 2001-125.

[31] A. Guirao et al., "A networked mezzanine card Linux processor," in Proc. 12th Real Time Congress on Nucl. and Plasma Sciences, Valencia June 2001, p 81 ff.

[32] W. Bux et al., "Technologies and Building Blocks for Fast Packet Forwarding," IEEE Commun. Mag., Jan. 2001.

[33] Network Processor Central, [online] http://www.linleygroup.com/npu/ .

[34] B. Jost and N. Neufeld, "A versatile Network Processor based electronics module for the LHCb Data Acquisition System," LHCb 2001-132.

[35] C. Bizeau et al., "RD31 Status Report '97," CERN/LHCC/97-05.

[36] IBM PowerNP documentation [online] http://www-3.ibm.com/chips/techlib/techlib.nsf/products/IBM_PowerNP_NP4GS3 .

[37] IBM PowerNP Reference Platform [online] http://www-3-.ibm.com/chips/techlib/techlib.nsf/products/IBM_PowerNP_NP4GS3_Reference_Platform .

[38] C. Gaspar, M. Dönszelmann, Ph. Charpentier, "DIM, a portable, light-weight package for information publishing, data transfer and inter-process communication," in Computer Physics Communications, October 2001. Vol 140, Num 1+2, p. 102-109.

[39] B. Franek and C. Gaspar, "SMI++ - An Object Oriented Framework for Designing Distributed Control Systems," in IEEE Trans. Nucl. Sci., Vol 45, Num 4, p.1946-1950), August 1998.

[40] OPC foundation, [online] http://www.opcfoundation.org/ .

[41] LHC Data Interchange WG, "User Requirements," Report_0600, June 2000, http://wwwlhc.cern.ch/Controls/WG/LDIWG/Report0600.PDF .

[42] D. Breton and Ph. Cros, "Serial Protocol for the Experiment ControlSystem of LHCb", LHCb Technical Note 2001-144.

[43] See e.g. http://atlasinfo.cern.ch/ATLAS/GROUPS/DAQTRIG/DCS/ELMB/elmb.html and links therein.

[44] C. Gaspar et al., "The Use of Credit Card sized PCs for interfacing electronics boards to the LHCb ECS", LHCb Technical Note, 2001-147.

[45] Rack Control see e.g. http://ess.web.cern.ch/ESS/RackProject/LHCRackMain.htm .

[46] JCOV http://proj-jcov.web.cern.ch/proj-JCOV/ .

[47] See e.g. http://lhcb-comp.web.cern.ch/lhcb-comp/ECS/pdf/mccontrol_report.pdf .

[48] See SPEC homepage [online] http://www.spec.org/ .

[49] Moore's Law at Intel, http://www.intel.com/research/silicon/mooreslaw.htm .

[50] LHCb Collaboration, "Muon System TDR", CERN/LHCC 2001-010.

[51] S. Bethke et al., "Report of the Steering Group of the LHC Computing Review," CERN/LHCC/2001-004 CERN/RRB-D-2001-3, Feb. 2001.

[52] Network Technology Tracking Team (NT3) Report 1999 [online] http://it-div-cs.web.cern.ch/it-div-cs/public/studies/nt3/nt3-1999.html .

[53] GAUDI : The software architecture and framework for building LHCb data processing applications, Pres. at: International Conference on Computing in High Energy and Nuclear Physics, Padua, Italy, 7 - 11 Feb 2000 / Ed. by Mazzucato, Mirco - INFN, Padua, 2000. - Comput. Phys. Commun. : 140 (2001) no.1-2 - pp.92 – 95.

[54] Myrinet [online] http://www.myri.com/ .

[55] J.-P. Dufey et al., "The LHCb trigger and data acquisition system," IEEE Trans. Nucl. Sci. : 47 (2000) no.2, pp.86-90.

[56] Ch. E. Spurgeon, Ethernet – The Definitive Guide, O'Reilly & Associates, Sebastopol, 2000.

[57] IEEE 802.3 Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications, 2000 ed.

[58] See http://www.oti.net/Systems/Foundry/PDF/big_iron.pdf for more details.

[59] J-P. Dufey, B. Jost, N. Neufeld and K. Paszkiewicz, "Results of Benchmark Tests upon a Gigabit Ethernet Switch," LHCb 2001-129.

[60] B. Jost, J-P. Dufey, N. Neufeld and M. Zuin, "Event building in an Intelligent Network Interface Card for the LHCb Readout Network," IEEE Trans. Nucl. Sci. vol. 48, 4.

[61] M. Zuin, "Embedded Event Building on a Gigabit Ethernet Network for the LHCb Experiment", Tesi di Laurea, Università Ca' Foscari - Venezia, 2000.

[62] N. Neufeld, "Implementation of the Event Filter Farm CPU node", LHCb 2001-143.

[63] RLX System 324, [online] http://www.rlxtechnologies.com/home.html .

[64] D. Clark et al., "An Analysis of TCP processing overhead," IEEE Commun. Mag., vol 27, no 6, pp. 23-29, June 1989.

[65] I. Pratt and K. Fraser, "Arsenic: a User-accessible Gigabit Ethernet Interface," Presented at IEEE INFOCOM 2001. [online] Available: http://infocom.ucsd.edu/papers/394-3981268191.pdf .

[66] J.-P. Dufey et al., "The LHCb Trigger and Data Acquisition System," IEEE Trans. Nucl. Sci., vol. 47, No 2, April 2000.

[67] V.E. Benes, "Optimal rearrangeable multistage connecting networks", Bell Syst. Tech. J., vol. 43, pp. 1641-1656, July 1964.

[68] C. Clos, "A study of non-blocking switching network", Bell Syst. Tech. J., vol. 32, pp. 406-424, March 1953.

[69] M.J. Karol et al., "Input versus Output Queuing on a Space-Division Packet Switch," IEEE Trans. Commun., vol. Com-35, No 12, Dec. 1987.

[70] G. Antchev et al., "The CMS Event-Builder Demonstrator based on Myrinet," IEEE Trans. Nucl. Sci., vol. 47, No 2, April 2000.

[71] Prizma, [online] http://www.zurich.ibm.com/cs/powerprs.html and references therein.

[72] CERN ECP/FEX: CASCADE User's Guide, CERN, Geneva, 1997.

[73] S. Schmeling, "LHCb Testbeam DAQ and Controls Support," [online]. Available: http://cern.ch/LHCb-Testbeam .

[74] S. Schmeling and R. Beneyton, "LHCb Testbeam 101," LHCb-2001-111.

Glossary of Terms

ADC	Analogue to Digital Converter
BCR	Bunch Counter Reset
CAN	Serial field bus, originally conceived for use in the automotive industry. Speed ranging between 2 Mb/s down-to 30 kb/s depending on length. Daisy-chained.
CC-PC	Credit-Card PC. Credit-Card sized electronics board containing the full functionality of a PC. To be embedded on a carrier board for full functionality.
DCS	Detector Control System. Used to be called 'slow control'. Hardware and software suite allowing control and monitoring of the operational state of the detector hardware, such as high and low voltage, gas flow, temperatures, etc.
DIM	Distributed Information Management.
DSS	Detector Safety System. A failsafe small system that ensures a safe state of the detector even if the ECS is not operational.
ECAL	The LHCb Electromagnetic Calorimeter
ECR	Event Counter Reset, equivalent to the L0 Event ID counter reset in LHCb
ECS	Experiment Control System. Hardware and software suite allowing control and monitoring of the entire experiment (detector and Data Acquisition) in a coherent and integrated fashion.
EFF	Event Filter Farm
ELMB	Embedded Local Monitoring Box. A CANbus node performing monitoring of up-to 64 analogue inputs. Also features some digital I/O capabilities. Developed in the framework of the Atlas experiment.
ESD	Experiment Summary Data. The output of the reconstruction of the raw data.
Event Building	Assembly of several fragments of data from different sources, belonging together through some criteria, to form one larger event(fragment).
FE chip	Front-end electronics chip
FE electronics	Front-end electronics
FEM	Front-End Multiplexer Component in the dataflow system to merge event fragments from several input links to form one output fragment. Interfaces to the Readout Units.
Finite State Machine (FSM)	Commonly used formalism allowing the modelling of a given 'process' in terms of states and transitions. Transitions from one state to the next are triggered by input 'events' and may cause the generation of output 'events' or actions. Finite state machines can be implemented in hardware or in software. They are used in hardware design, protocol modelling, compilers, control automation and even natural language.
FPGA	Field Programmable Gate Arrays
GPS	Global Positioning System
HCAL	The LHCb Hadron Calorimeter
HLT	High-Level Trigger. Software algorithms that perform the final selection of the events. Last stage in the dataflow that selects events for further processing. Events accepted by the HLTs will be reconstructed and written to permanent storage.
I2C	A communication protocol invented by Philips to control integrated circuits
IT	The LHCb Inner Tracking Detector

JCOP	The Joint Controls Project. A collaborative effort between the four LHC experiments and CERN/IT-CO to provide common solutions and support for the controls systems
JTAG	An IEEE standard, who's main purpose was initially to allow testing of the connections on electronics boards. Later also used for configuration of integrated circuits.
L0	Level-0 trigger
L0DU	L0 trigger Decision Unit
L0FE	L0 front-end electronics
L1	Level-1 trigger
L1DU	Level-1 trigger Decision Unit
L1FE	Level-1 front-end electronics. Component in the Data Acquisition chain that holds the data for the latency of the Level-1 trigger and performs the zero-suppression. The output of the Level-1 Front-end electronics is the starting point of the LHCb DAQ system proper.
LAN	Local Area Network
Level-1 front-end electronics	see L1FE
MUON	The LHCb Muon System
NIC	Network Interface Card. An interface in a computer to a network technology.
NP	Network Processor. Specialized integrated circuits for network packet or frame manipulations.
OT	The LHCb Outer Tracking Detector.
PLC	Programmable Logic Controller. Reliable processor based equipment used for (closed loop) process control.
PS	The LHCb Pre-Shower Detector.
RICH	The LHCb Ring Imaging Cerenkov Detector.
RN	Readout Network. Large switching network to support event building.
RS	Readout Supervisor
RU	Readout Unit. Component in the dataflow system to merge event fragments from several input links to form one output fragment. Also interfaces to the Readout Network.
SCADA	Supervisory Controls And Data Acquisition system. An industry term for controls systems.
SEU	Single Event Upset. A flip of a gate or a bit under the influence of highly ionising particle.
SFC	Sub-Farm Controller. Component of the dataflow system that performs the final event-building and distributes the complete events to the CPUs of the sub-farm
SI95	A measure of CPU power for integer operations defined through a suite of programs issued by the SPEC organisation.
SMI	State Manager Interface.
SPD	The LHCb Scintillator Pad Detector
SPECS	Serial Protocol for ECS. A serial bus developped at LAL/Orsay to provide high-speed (10 Mb/s) controls access to front-end electronics.
TFC	Timing and Fast Control
TFC Switch	Programmable patch panel in the TFC system to distribute the timing, trigger and control information
Throttle OR	Module making a logical OR of the throttle signals from several L1 front-end electronics boards

TTC	Timing, Trigger, and Control system developed by RD12
TTCcf	TTC clock fanouts implemented in the TTCmi
TTCex	TTC encoder module
TTCmi	TTC machine interface to receive the LHC timing information
TTCpr	TTC PCI/PMC receiver used to receive the TTC signal and transfer the data to a PC
TTCrx	TTC receiver chip receiving and decoding the TTC signal
TTCtx	TTC laser transmitter converting the TTC signal from electrical to optical
TTCvi	TTC-VME-bus interface developed by the ATLAS experiment
TTCvx	TTC-VME-bus encoder used to encode the TTC signal
UDP	User Datagram Protocol. A connectionless, IP-based and user-defined protocol. It does not guarantee the correct delivery of the data.
VELO	The LHCb Vertex Locator
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit
VELO	The LHCb Vertex Locator
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit