NTNU
Innovation and Creativity

Development of the Control Interface for the Fast Magnet Current Change Monitor (FMCM)

Part of the LHC Machine Protection System at CERN

Tonje Vik Jevard

Master of Science in Communication Technology
Submission date: March 2007
Supervisor: Nils Holte, IET
Co-supervisor: Professor Kjetil Svarstad, IET
Dr. Markus Zerlauth, CERN
Dr. Rüdiger Schmidt, CERN

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Problem Description

The LHC, the world's largest particle accelerator currently under construction at CERN, has large number of magnets, both superconducting and normal conducting installed to guide the two proton beams around the accelerator. The Fast Magnet Current Change Monitor (FMCM) has been designed to provide a fast and reliable trigger signal, for dumping the beams in the case of powering failures of magnets which have fast effects on the beam trajectories.

For remote monitoring and Post Mortem analysis every FMCM is connected to the CERN control system by means of an RS-422 interface. This master's thesis is focusing on the software development of the control interface for the described FMCM units. The FMCM control interface will be responsible for the communication between the different FMCMs and CERN's control system, to collect data acquired by each FMCM and to provide a tool for the machine operators to supervise the device. The project also includes the analysis of the currently implemented RS-422 interface in terms of Electromagnetic Compatibility (EMC), Signal Integrity (SI) and reliability. Together with integration and configuration of the RS-422 interface into the VME framework (using CERN standards for hardware and configuration tools) and validation of the communication in terms of the initial requirements for speed based on a representative test-setup.

Assignment given: 04. September 2006
Supervisor: Nils Holte, IET

Abstract

A large number of magnets, both superconducting and normal conducting, are installed for the guidance of the two proton beams around the Large Hadron Collider (LHC), the world's largest particle accelerator currently under construction at the European Organization for Nuclear Research (CERN). Due to the unprecedented energies stored in the beams and the magnets, sophisticated systems are under development to protect the equipment in case of failure. However, scenarios have been identified where failures in the magnet powering will lead to very fast beam losses in less than 100 μs, due to the low time constants of the electrical circuits and the consequent fast current decay. For these circuits, systems that are currently deployed will not be fast enough to generate and transmit a beam dump request before the magnetic field change affects the beam trajectory. A dedicated system for the detection of such fast failures is already operational at the Hadron-Electron Ring Accelerator (HERA) in Hamburg. This system, the Fast Magnet Current Change Monitor (FMCM), has been adapted to meet CERN requirements and needs to be integrated into the CERN accelerator environment.

For remote monitoring and Post Mortem analysis every FMCM is connected to the CERN control system by means of an RS-422 interface. This master's thesis is focused on the software development and analysis of the control interface for the described FMCM units. The communication between the FMCMs and the CERN control system has been designed and implemented in C++, following the guidelines given by the Front End Software Architecture (FESA) framework. An analysis of the RS-422 interface with respect to Signal Integrity and Electromagnetic Compatibility verified the current setup of the RS-422 serial interface for the given transmission parameters. Transient bursts are considered to be the most common type of disturbance in the LHC and the related surface buildings. Hence, error detection has been implemented to ensure reliable communication by causing retransmissions of the data until it has been correctly received.

Preface

This master's thesis has been carried out as part of the Technical Student Program at the European Organization for Nuclear Research, CERN, Geneva, Switzerland.

By participating in the development, installation and commissioning of one of the largest and most complicated machines envisaged to date, I have been given a great opportunity to apply my academic knowledge on a real system and have gained important practical experience in the process.

The Technical Student Program will continue until the end of May 2007, and I will continue to work on this project to prepare the FMCM control interface for the installation of the FMCM units in the LHC and its transfer lines, starting mid 2007.

Acknowledgments

Many people have contributed to the final result presented in this thesis. First and foremost I would like to thank Markus Zerlauth for being a dedicated supervisor, for his ideas and guidance. Secondly, I would like to thank Benjamin Todd for his help to provide a test facility for the system. I would also like to thank Matthias Werner at DESY, for being available and answering questions concerning the hardware and software development of the FMCM. Bruno Puccio and Rudiger Schmidt, thank you for your advice and for introducing me to the LHC and the Machine Protection System. Also, I want to thank the rest of the Machine Interlock Section for creating such a great working environment.

I would like to thank Leandro Fernandez and Alain Gagnaire from the Front End Section at CERN for answering all my questions concerning the FESA framework and assistance with the front end equipment.

Finally I would like to thank my supervisor at NTNU, Nils Holte, for taking on the responsibility for this master thesis and making it possible for me to accept this opportunity at CERN.

Contents

List of Figures

List of Tables

List of Abbreviations

LHC	Large Hadron Collider
CERN	Conseil Européen pour la Recherche Nucléaire
HERA	Hadron-Electron Ring Accelerator
FMCM	Fast Magnet Current Change Monitor
FESA	Front End Software Architecture
EMC	Electro Magnetic Compatibility
SI	Signal Integrity
PS	Proton Synchrotron
SPS	Super Proton Synchrotron
LEP	Large Electron-Positron (collider)
WWW	World Wide Web
CNGS	CERN Neutrinos to Gran Sasso
ALICE	A Large Ion Collider Experiment
ATLAS	A Toroidal LHC Apparatus
CMS	The Compact Muon Solenoid experiment
LHC-b	The Large Hadron Collider beauty experiment
RF	Radio Frequency
PSB	Proton Synchrotron Booster
LEIR	Low Energy Ion Ring
MPS	Machine Protection System
LBDS	LHC Beam Dumping System
BLM	Beam Loss Monitor
BIS	Beam Interlock System
BLMS	Beam Loss Monitor System
SMP	Safe Machine Parameters
GMT	General Machine Timing
WIC	Warm magnet Interlock Controller
QPS	Quench Protection System
PIC	Powering Interlock Controller
PM	Post Mortem
VME	Versa Module Eurocard
SCADA	Supervisory Control And Data Acquisition
CCC	CERN Control Center
DCCT	Direct Current Current Transformer
ADC	Analog to Digital Converter
UTC	Universal Coordinated Time
CTRP	Control Timing Receiver PCI
pps	pulse per second

OSI	Open Systems Interconnection
UART	Universal Asynchronous Receiver and Transmitter
FIFO	First In First Out
XML	eXtensible Markup Language
FPGA	Field Programmable Gate Array
FEC	Forward Error Correction
IEC	International Electrotechnical Commission
HF	High Frequency
BER	Bit Error Rate
RS-422	Recommended Standard 422
PCB	Printed Circuit Board
TTL	Transistor-Transistor Logic
IP	Industrypack
BIC	Beam Interlock Controller
ppm	parts per million
PLL	Phase Locked Loop

Chapter 1

Introduction

Once in operation, the Large Hadron Collider (LHC) currently under construction at the European Organization for Nuclear Research (CERN) will be the largest and one of the most complex scientific instruments every built. Two counter rotating proton beams will be accelerated to 7 TeV and collided in 4 large underground experiments to find further answers to fundamental questions in particle physics. Particle accelerators such as the LHC require a large number of magnets, both superconducting and normal conducting for the guidance of the two proton beams around the 27 km long circular tunnel. Due to the unprecedented stored energies, 360 MJ in each of the two proton beams and 10 GJ in the magnet system, sophisticated systems have to be developed to protect the equipment from any damage in the case of equipment failures. Especially for the normal conducting magnets installed in high radiation areas, the decay time of the magnetic field in case of powering failures is very small resulting in a large deviation of the proton beams, in some cases within less than 100 μs. For these circuits, the protection systems currently deployed will not be fast enough to request the removal of the proton beams from the machine before the magnetic field change affects the beam trajectories. A dedicated system for the detection of such fast failures has been developed and put into operation in the Hadron-Electron Ring Accelerator (HERA) in Hamburg. This system, the Fast Magnet Current Change Monitor (FMCM), has recently been adapted to meet CERN requirements and needs to be integrated into the CERN accelerator environment.

Due to the geographical distribution of installations in CERN's accelerator complex and for personnel safety the equipment will not be accessible during machine operation. To allow for remote monitoring and read out of Post Mortem data after an event, every FMCM is connected to the CERN control system by means of an RS-422 interface. This master's thesis is focusing on the software development of the control interface for the described FMCM units. The project also includes the analysis of the currently implemented RS-422 protocol in terms of Electromagnetic Compatibility (EMC), Signal Integrity (SI) and reliability. Together with integration and configuration of the RS-422 interface into the VME framework (using CERN standards for hardware and configuration tools) and validation of the communication in terms of the initial requirements for speed based on a representative test-setup.

Part 1 of this report describes the background and context for this thesis and explains the main motivations for the development of the FMCM. Chapter 1 is an

introduction to CERN and the LHC in general, with facts and figures of the challenges involved in such a large project. Chapter 2 is an introduction to the FMCM and its role within the Machine Protection System.

Part 2 contains the preliminary study of the FMCM control interface, design and implementation. Chapter 3 includes the system specifications and functional requirements, and a description of the resulting design and the implementation of the front end computer software. Chapter 4 includes a general discussion of error handling in serial communication as well as a description of the error handling methods applied to the FMCM control interface.

Part 3 discusses the different tests and analysis of the control interface in general and the RS-422 interface in particular. Chapter 5 describes the background and purpose of the different tests as well as how they were carried out. Chapter 6 presents the obtained results with discussions comparing the results to the initial requirements. Finally, Chapter 7 is a summary of the conclusions drawn from the different results.

Part I

Background & Context

Chapter 2

CERN and the LHC

CERN officially came into being in 1954 as one of the first joint ventures in Europe with the following mandate:

> "*The Organization shall provide for collaboration among European States in nuclear research of a pure scientific and fundamental character, and in research essentially related thereto. The Organization shall have no concern with work for military requirements and the results of its experimental and theoretical work shall be published or otherwise made generally available.*" [1]

The name CERN is derived from the French term Conseil Européen pour la Recherche Nucléaire, or European Council for Nuclear Research which was a temporary name and later changed to European Organization for Nuclear Research. However, today 50 years later, CERN is usually referred to as European laboratory for particle physics as the work of the physicians long time ago went beyond the study of the atomic nucleus. Today one talks about high energy physics. The main historical events at CERN are listed in Table 2.1. CERN is run by the 20 European member states, but also non-European countries make significant contributions. The main function of CERN is to provide particle accelerators and other infrastructure which

1949	Louis de Broglie proposes the creation of a European science laboratory
1954	The foundation of CERN
1957	600 MeV Synchrocyclotron commissioned
1959	28 GeV Proton Synchrotron (PS) commissioned
1963	Bubble Chambers give first evidence of neutrino interaction
1973	Neutral Currents discovery ratified
1976	Super Proton Synchrotron (SPS) commissioned
1984	Noble Prize awarded to C. Rubia ans S van der Meer for W and Z
1989	Large Electron Positron (LEP) commissioned
1990	T. Berners-Lee propose the World Wide Web (www)
1992	Nobel Prize awarded to G. Charpak for Multi-Wire Chambers
2006	CERN Neutrinos to Gran Sasso Project (CNGS) begins
2007	First commissioning with beam scheduled for the Large Hadron Collider (LHC)

Table 2.1: A brief history of CERN [2]

1982	First studies for the LHC project
1994	Approval of the LHC by the CERN Council
1996	Final decision to start the LHC construction
1996	LEP operation at 100 GeV (W-factory)
2000	End of LEP operation
2002	LEP equipment removed
2003	Start of the LHC installation
2006	Start of hardware commissioning
2007	Commissioning with beam

Table 2.2: History of CERN related to the LHC Project [4]

international collaborations use for research in high energy physics. Several important achievements in particle physics are derived from experiments at CERN, and not all of them are mentioned in Table 2.1. These days most of the activities at CERN are directed toward building a new accelerator, the LHC.

2.1 The LHC

Once put into operation the LHC (see Figure 2.1) will be the world's largest scientific instrument and the most powerful particle accelerator ever build. The motivation behind the LHC is to find answers to some fundamental questions in particle physics. The current understanding of particles and mass relies on a model which implies the existence of a Higgs boson. However, to detect the Higgs boson energy in the TeV range is required, which is what the LHC will provide. The LHC is being build in the

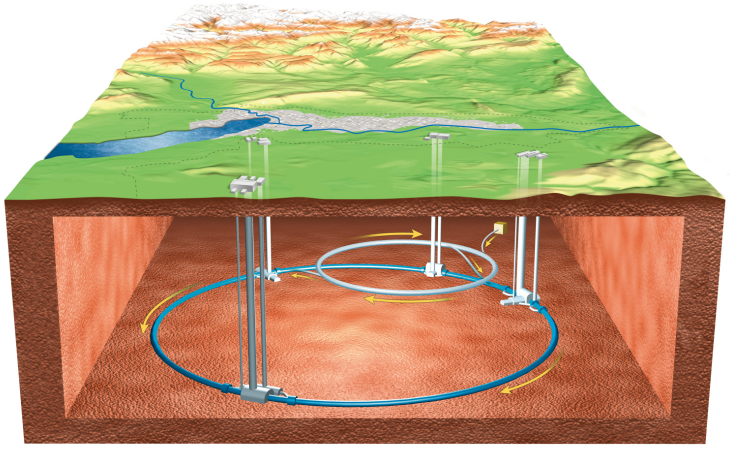

Figure 2.1: The Large Hadron Collider

already existing infrastructure of the Large Electron-Positron storage ring (LEP)[3], a 27 km long circular tunnel underground. The LEP experiments were finished in 2000 and the first LHC installations were done in 2003. The first commissioning with beam in the LHC is intended for 2007, 25 years after the first initial studies of the LHC, see Table 2.2.

In the LEP experiments electrons and positrons were accelerated and made to col-

lide. Before the LEP was dismantled in 2000 it was able to accelerate the particles to 104 GeV/c. In the LHC experiments protons or heavy ions will be injected into the LHC machine at 450 GeV/c and accelerated to 7 TeV/c in approximately 30 minutes. There will be two beams, one in each direction and the two beams will collide at four intersections where large detectors are located. The LHC has an eight-fold symmetry with eight arc sections and eight straight sections. The detectors and the machine operation systems are placed in the straight sections, see Figure 2.2. The

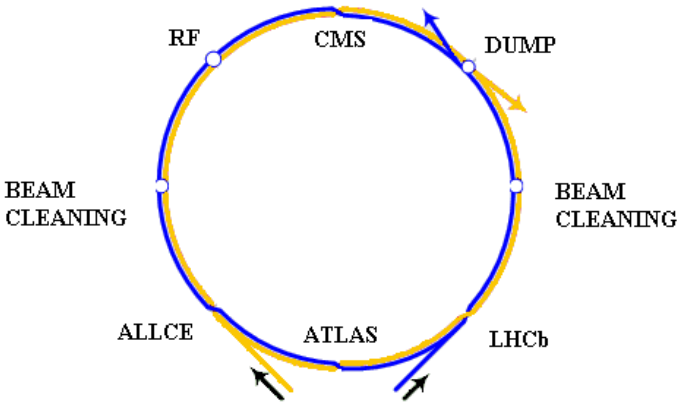

Figure 2.2: Schematic layout of the LHC

four detectors belong to four different experiments: ALICE (A Large Ion Collider Experiment), ATLAS (A Toroidal LHC Apparatus), CMS (The Compact Muon Solenoid experiment) and LHC-b (The Large Hadron Collider beauty experiment). The researchers at ALICE will study quark-gluon plasma, a phase of matter that existed for just a fraction of seconds after the Big Bang. At LHC-b they will study particles containing beauty and anti-beauty quarks to understand the assumed unbalance between matter and anti-matter in the universe. The CMS is a general purpose detector for experimental physics dedicated to proton collisions. ATLAS is the main detector for head-on collisions of protons, and where one would expect to find proof of the existence of the Higgs boson. The four remaining straight sections are dedicated to systems for machine operation: Beam Cleaning, Beam Dump and Radio Frequency (RF) acceleration.

To prepare the particle beams that are injected into the LHC a series of accelerators are used: The Super Proton Synchrotron (SPS), the Proton Synchrotron (PS) and the Proton Synchrotron Booster (PSB). The Low Energy Ion Ring (LEIR) is not an accelerator but an accumulator and is only used for experiments with ion collisions. The other accelerators however, always take part in the pre-acceleration of the particles before they are injected into the LHC, see Figure 2.3. The SPS injects the particle beams into the LHC through the transfer lines connecting the two accelerators.

The fundamentals of particle accelerator theory is based on the Lorentz equation, which states that the force on a particle in an electromagnetic field is given by the

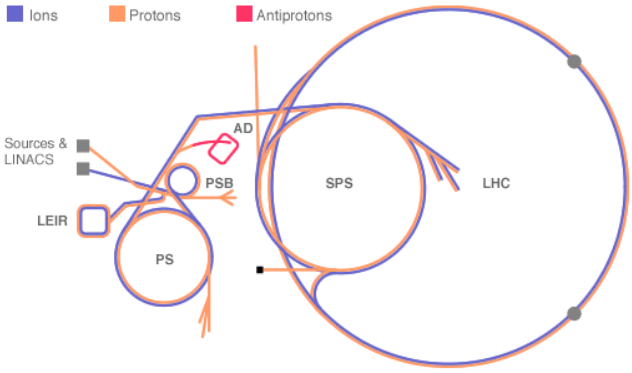

Figure 2.3: The CERN Accelerator complex

cross-product of the particle velocity $\vec{v}$ and the magnetic field $\vec{B}$ plus the electrical field $\vec{E}$, multiplied with the particle charge q:

$$\vec{F} = q\left(\vec{E} + \vec{v} \times \vec{B}\right) \tag{2.1}$$

Thus a positively charged particle will be accelerated in the same linear orientation as the electrical field, but will be deflected perpendicularly to the magnetic field according to the right-hand rule. In the LHC two types of magnets are needed for basic particle steering. That is dipole magnets for deflection and quadrupole magnets for focalization of the beam. The quadrupole magnets focus the beam similar to lenses used in light optics in the sense that it focalizes the beam in one plane but defocalizes it in the other plane. Thus, to focus the beam in both planes one needs a pair of quadrupoles with alternating optical functions. The particle acceleration, in sense of increase of energy, takes place in the RF cavity. A time-varying electrical field accelerates charged particles in bunches. A bunch consists of up to 1.15×10^{11} particles which enter the cavity just at the right time, or equivalently with the right phase. A particle entering out of phase will be decelerated and lost.

The magnets to guide the particle beams in the LHC will be both superconducting and normal conducting. In circular accelerators the particles pass through the RF cavity once during every turn, and are accelerated at every passage. The LHC RF cavity operates at 400 MHZ and as already mentioned after 30 minutes of acceleration the particles reach 7 TeV/c. The angle of particle deflection is proportional to the magnetic field divided by the momentum, and as the momentum increases one must increase the magnetic field to maintain the same angle of deflection throughout acceleration. This is why superconducting magnets are needed as they can reach the target magnetic field of 8.33 Tesla and keep the particle in their orbit. Normal conducting magnets are limited to 1-2 Tesla. However, the normal conducting magnets are more resistant to radiation and are therefore used close to the detectors and in other parts of the machine where the radiation is too high for the use of superconducting magnets.

2.2 LHC Parameters and Challenges

The design and realization of the LHC has required engineers from several fields to deliver solutions that represent the cutting edge of accelerator technology. The main LHC parameters are summarized in Table 2.3 and Table 2.4.

Momentum at collision	7	TeV/c
Injection energy	0.45	TeV/c
Number of dipole magnets	1232	
Number of quadrupole magnets	430	
Number of corrector magnets	about 8000	
Particles per bunch	1.15×10^{11}	
Number of bunches per beam	2808	
Bunch spacing	25	ns
Beam pipe diameter	56	mm
Typical rms beam size in arcs at 7 TeV	200-300	μm

Table 2.3: Some machine parameters [4]

Energy stored in magnet system	10	GJ
Energy stored in one dipole circuit	1.1	GJ
Energy stored in one beam	362	MJ
Average power over a fill of 10 hours, both beams	20	kW
Beam power averaged over one turn, one beam	3.9	TW
World Net Electricity Generation (2002)	1.7	TW
Energy needed to heat and melt 1 kg copper	700	kJ

Table 2.4: Energy stored in magnets and beams [5]

The LHC has 1232 superconducting dipole magnets with niobium-titanium coils and the superconducting magnets are maintained at only 1.9 K (about -271°C). The magnets are cooled by super-fluid helium which is provided by a large and complex cryogenic system. Each sector of the LHC is served by one of the four cryogenic plants through a 3.3 km cryogenic distribution line. The sectors are further divided into cells, where cryogenic loops extends a long a lattice cell of 107 m, supplying the magnets in one cell. Contrary to electron-positron or proton antiproton colliders, the two counter-rotating proton beams in the LHC need opposite deflecting magnetic fields in the arcs. A major feature of the LHC dipoles is their two-in-one design, providing opposite magnetic fields for the two counter-rotating beams within a single structure. Four vacuum systems are needed, one for each beam, one insulation vacuum system for the magnets and one insulation vacuum for the cryogenic distribution line. Along the arc, there will be several thousand crates with radiation tolerant electronics for quench protection, orbit corrector power converters, instrumentation for the beams and the vacuum and cryogenic system. A quench is what happens when a superconducting magnet looses it superconductivity and becomes resistive. There are several mechanisms that can lead to a quench like beam loss, movement of the superconductor by several μm (friction and heat dissipation) or a failure in the cryogenic system. In each case, the temperature of the magnet is rising. Feeding current into magnets that operate at 1.9 K is an other issue, and has led to new technology like industrial use of High Temperature Superconducting

material [4].

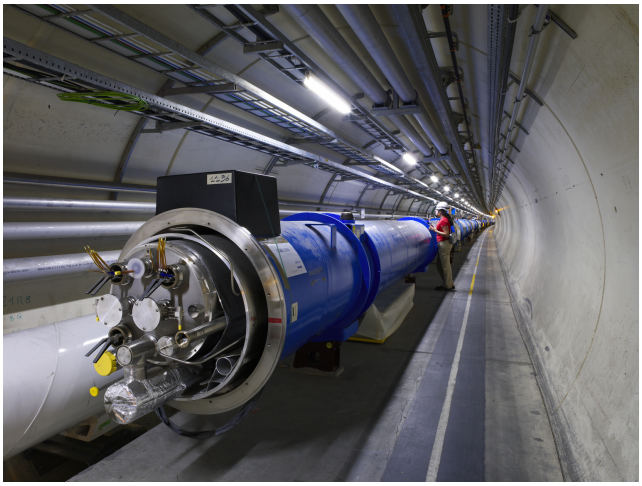

Figure 2.4: A dipole magnet about to be installed in the LHC

The energy stored in each dipole magnet is 7.6 MJ, and the energy stored in the entire magnet system is 10 GJ. For comparison this is equivalent to 230 kg of gasoline and with such an amount of energy one could melt nearly 15 tons of copper [5]. In order to handle the energy stored in the magnet system, the LHC magnets are powered in several independent powering sub-sectors. This reduces the energy stored in the individual circuits, and reduces the complexity of the protection systems of the superconducting magnets which are then similar to equivalent systems in other accelerators (HERA, TEVATRON, RHIC), see Figure 2.5. The proton momentum before collison is 7 TeV/c which is a factor of seven to sixteen above accelerators such as SPS, TEVATRON and HERA. However, the energy stored in one beam is 362 MJ and is larger by a factor of 200 due to the high beam intensity. The maximum energy density, which is important when it comes to equipment damage, is a factor of 1000 higher than in other accelerators. This is due to the small beam dimensions. An uncontrolled loss of even a very small fraction of the 7 TeV proton beam could cause significant equipment damage. Even the beam injected from the SPS into the LHC at 450 GeV can damage the equipment and it is therefore necessary that the protection is efficient through the whole cycle of machine operation [5]. The risk related to storing such a large amount of energy has been minimized by advanced protection systems such as the Fast Magnet Current Change Monitor. These systems will be discussed in further detail in Chapter 3.

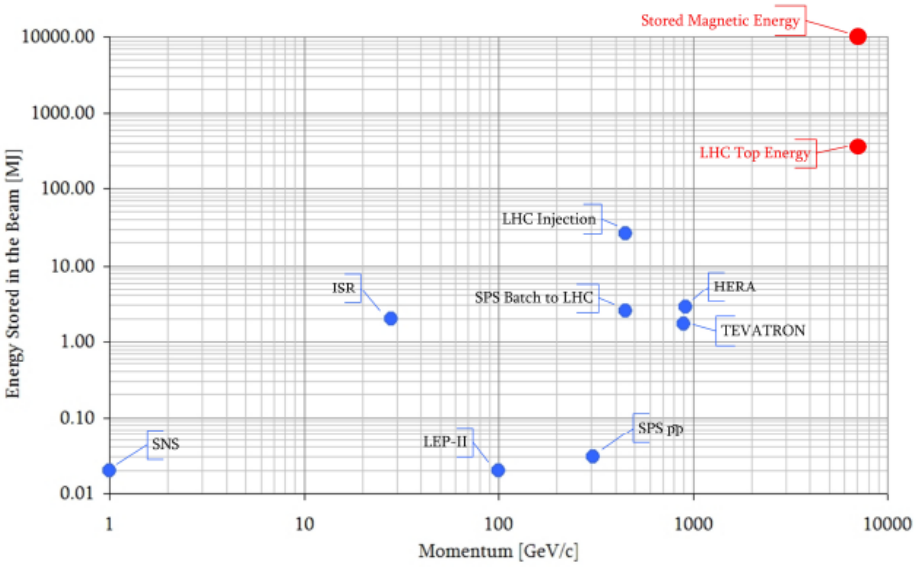

Figure 2.5: Stored Energy in the LHC and other accelerators [6, 7]

Chapter 3

The LHC Machine Protection System

The LHC will collide two counter-rotating proton beams, each with a momentum of 7 TeV/c. Uncontrolled beam loss could cause significant equipment damage and accelerator downtime and safe operation of the LHC will rely on the Machine Protection System (MPS).

3.1 General Introduction to the MPS

The general requirements for the LHC protection systems are the following:“

- Protect the accelerator equipment: The first priority is to protect equipment from damage, in the LHC ring and during the transfer from the pre-accelerator SPS to the LHC. The second priority is to protect superconducting magnets from quenching.
- Protect the beam: Protection systems should only dump the particle beams when necessary. “False” beam dumps should be avoided in order not to compromise availability.
- Provide the evidence: In case of failure, complete and coherent diagnostics data should be provided to accurately understand what caused failure and if the protection systems functioned correctly. [5]”

The Machine Protection System is a collection of systems that interact to ensure safe operation of the LHC. The different elements are shown in Figure 3.1. The strategy behind the design of the MPS is to detect failures in accelerator equipment early and dump the beam before it is affected. Also, active beam monitors will detect if there are abnormalities in the beam parameters and if so request a beam dump within a very short time. Reliable and fast transmission of the beam dump requests from the various systems are ensured by the Beam Interlock System. The MPS is mostly redundant as in most cases a failure may be captured by more than a single system though with different time-scales and different effects [5]. The most relevant parts of the MPS are briefly described in the following sections.

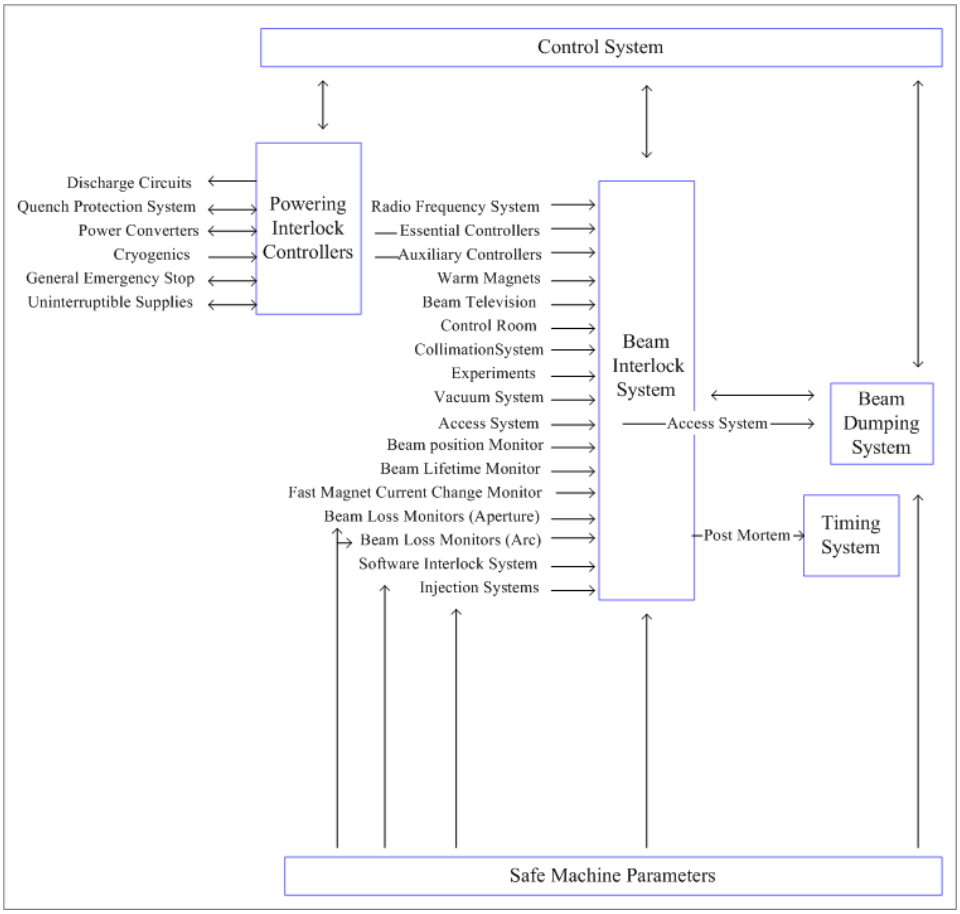

Figure 3.1: Overview of the Machine Protection System [6]

The LHC Beam Dumping System (LBDS)

The Beam Dumping System dumps the beam into a 10 meter long graphite block that absorbs the beam energy without any significant damage. This is the only part of the LHC machine that can withstand the impact of a full LHC beam. The LBDS will perform a beam dump when receiving a signal either from the Beam Interlock controller, the access system (if personnel are entering potentially dangerous parts of the accelerator complex) or a dedicated Beam Loss Monitor (BLM) close to the extraction line. The LBDS can also trigger a beam dump by it self if there is an internal failure. [6, 4, 5]

The Beam Interlock System (BIS)

The Beam Interlock System collects beam dump requests from the various user systems distributed around the LHC and transmits such requests to the Beam Dumping System. Due to the possibility of fast beam losses, the system must transmit dump requests from connected systems around the LHC to the Beam Dumping System within less than approximately 100 μs.[6, 5, 8]

The Beam Loss Monitor System (BLMS)

"The Beam Loss Monitor System consists of several thousand ionisation chambers located along-side the beam pipe in the SPS, LHC and transfer lines. When particles escape from the accelerator they are detected by these chambers and if the number of lost particles is above a certain threshold then a beam dump is requested."[6]

Safe Machine Parameters (SMP)

The Safe Machine Parameters (SMP) System generates and distributes machine parameters that are critical for LHC beam operation, such as beam energy and intensity, to equipment using this information for protection. Safe Machine Parameters are sent through the CERN General Machine Timing (GMT) system. [6, 5]

Beam Absorbers and Collimators

Beam absorbers and collimators are large movable carbon or metal jaws which can be positioned very close to the beam orbit, capturing any particles with large amplitudes which would otherwise be lost in the accelerator. [6, 5]

Normal and Superconductive Magnet Powering Interlocks

The normal conducting magnets in the LHC are protected by Warm magnet Interlock Controllers (WIC). The WIC is based on temperature sensors which detect temperatures above a certain threshold in the magnets. These sensors are connected to industrial controllers that will switch off the power converters in case of failure. A beam dump request is then transmitted via the Beam Interlock System.

The superconducting magnets are protected by two systems, the Quench Protection System (QPS) and Powering Interlock Controllers (PIC). In the case of a quench the QPS will initialize the discharge of energy stored in the quenched magnet and send a request to the PIC to switch off the corresponding power converters. In case of a quench or another powering failure, the PIC sends signals to the Beam Interlock System which will request a beam dump before the decaying magnetic field can affect the beam orbit. [6]

3.2 The Fast Magnet Current Change Monitor (FMCM)

The FMCM is an important protection device in the Machine Protection System. A beam incident in the TT40 extraction line of the SPS (destroying several vacuum chambers and a quadrupole magnet) showed the necessity of an additional protective device. The main task of the FMCM is to monitor fast current changes in normal conducting magnets in critical parts of the LHC and its transfer lines. A fast current change can be caused by sudden powering failures and implies a change in the magnetic field which changes the particle trajectory leading to fast beam losses.

When the FMCM detects such a fast current change it will send a beam dump request to the Beam Interlock System.

The FMCM was initially developed at DESY for the HERA storage ring to protect the machine from beam induced damage in case of sudden power supply failures. It has been adapted to meet CERN requirements and several units have already been successfully operated during the commissioning and initial operation of CERN Neutrinos to Grand Sasso (CNGS).

The difference between the FMCM and other systems protecting the normal conducting magnets is the FMCMs capability of detecting current changes in the range of 10E-4 within less than 1 ms. Detection of slow changes in the absolute value of the current is assured by other systems [9].

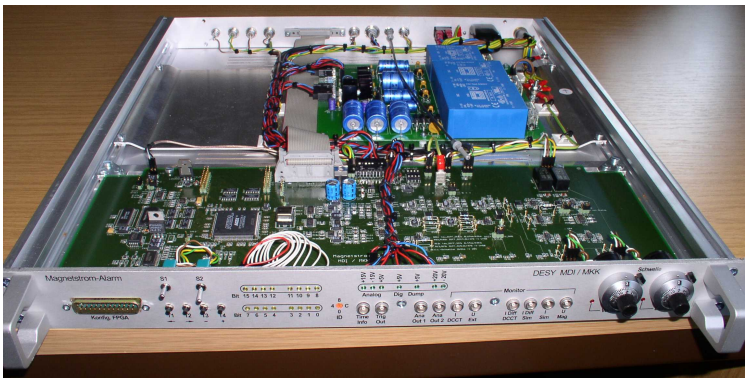

Figure 3.2: The Fast Magnet Current Change Monitor

3.2.1 Introduction to the Studies Leading to the Development of the FMCM

Failures in the powering system of the LHC magnets are generally relatively slow events due to the large inductance of superconducting magnets. The natural time constant of the electrical circuits are in the order of a few seconds for the normal conducting magnets and up to some minutes for the super conducting magnets. Beam loss monitors will normally detect the corresponding beam losses and trigger a beam dump. However, power failure in combination with normal conducting magnets can lead to very fast decaying magnetic fields, for which the response time of the Beam Loss Monitor is too slow. The number of particles lost in the collimators and the accelerator equipment will reach damage level before the beam has been safely extracted from the accelerator. In case of power converter failure the error magnet field behaves according to:

$$\Delta B_{error}(t) = B_0(1 - \exp^{-\frac{t}{\tau}}) \tag{3.1}$$

where $\tau = L/R$ is the decay time constant determined by the circuits inductance and resistance. By modeling the accelerator as a cascade of lenses, and applying laws from optical physics one can simulate the trajectory of one particle as it turns in the accelerator. Combining this with Equation 3.1 and knowing that the particles in one bunch are generally gaussian distributed, one can find the number of particles which

are extracted by the collimators after powering failure of the corresponding magnet. Figure 3.3, shows the result of a simulation following this model [10]. These studies

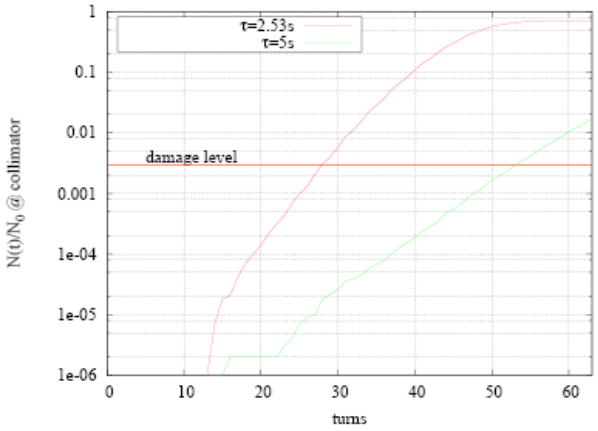

Figure 3.3: Total number of particles lost at collimator as a function of the number of turns the beam have made after the occurrence of a sudden power failure. The horizontal red line represents the damage level of the collimator. In the green plot to the right, the natural time constant of the circuit has been doubled compared to the red plot to the left. This illustrates how sensitive the collimators are for fast powering failures of normal conducting magnets. [10]

made clear that the Beam Loss Monitor was not suitable to act on such fast powering failures, and other concepts were studied as well [11] before the final decision was to further develop the FMCM designed at DESY.

3.2.2 Short Functional Description

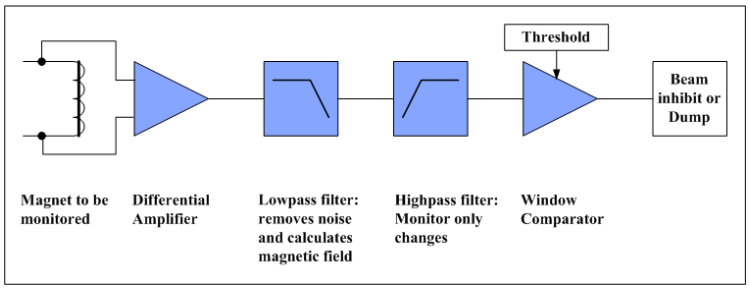

Figure 3.4: The working principle of the FMCM [9]

The FMCM measures the voltage across the magnet or circuit with an external voltage divider and an isolated amplifier. It then performs a real-time calculation of the magnet current using a low pass filter. Next, the signal passes a high pass filter to amplify the fast current changes which typically happen during a power converter failure. The resulting signal is fed to a comparator which decides whether or not the fast current changes are inside a tolerance window. The tolerance window is set by a potentiometer on the front panel of the FMCM. If the signal exceeds the tolerance window, a signal is sent to the Beam Interlock System and Post Mortem (PM) data

is stored for later analysis, see Figure 3.4. The acquisition of PM data is described in Section 4.1.1.[12]

The LHC is a slow cycling machine with a long filling process and operational cycle, see Figure 3.5. The time interval between two injections is approximately 20 seconds, depending on the cycle-time of the pre-injections, see Figure 3.6. Thus, there are slow ramping magnets in the LHC and fast ramping magnets in the SPS and transfer lines. This leads to two different configurations for the FMCM, especially for the comparator and read out of PM data. The signal to the Beam Interlock System is different in the transfer lines and the LHC. In the transfer lines the FMCM gives the BIC an extraction permit when the magnets are ready. Thus the signal is "low" most of the time and then going "high", when the permit is given (maximum every 20 seconds). In the LHC the FMCM gives the BIC a beam permit, which is "high" in normal operation an only changes to "low" when there is an event and the beams need to be dumped.

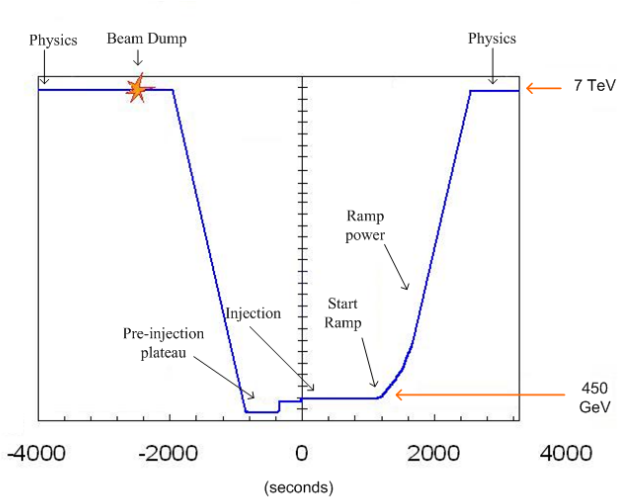

Figure 3.5: The LHC cycle

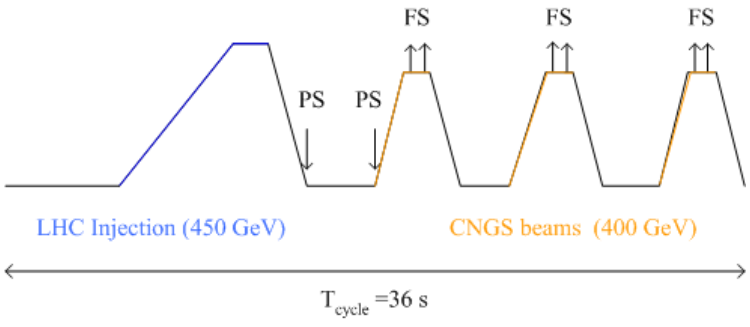

Figure 3.6: A typical SPS cycle with a LHC injection followed by CNGS injections.

Part II

Study & Implementation

Chapter 4

The FMCM Control Interface

This chapter describes how the third requirement of the LHC Machine Protection System, see Section 3.1, will be achieved for the FMCM. That is, how to *provide the evidence* after a failure. The FMCM control interface is responsible for the communication between the different FMCMs and the CERN control system, to collect data acquired by the FMCM and to provide a tool for the machine operators to supervise the device. The control interface of the FMCM can also be said to contribute to the protection of accelerator equipment as the parameters retrieved from the FMCM can be used by diagnostic tools for surveillance and operational control of the LHC. Remote surveillance of the different protection systems makes it possible to analyze events and understand less critical failures without having to stop the accelerator.

4.1 System Specification and Requirements

This section is based on the functional specification of the FMCM [12].

This master thesis is focused on the development of the software for the control interface as the hardware has already been defined. However, tests will be done to verify the installations of the hardware in sense of electromagnetic compatibility and signal integrity, see Chapter 6. The choice of hardware has influence on the software design, and there are certain guidelines in the LHC project for integration of hardware that must be followed.

The FMCM is connected to a front end computer through a serial RS-422 interface. The front end computer is the crate manager in a Versa Module Eurocard (VME) crate, where several cards for different systems are installed. The front end computer is the master and communicates with the other cards, called slaves, via the VME bus. One of these slaves is the FMCM, although the FMCM is situated outside the crate itself. As the FMCM uses the RS-422 protocol and the VME bus has a RS-232 interface, two intermediate modules are used to make a RS-422 interface on the VME side of the connection. These two modules, the VME bus industrypack carrier VIPC626-ET and the industrypack IP-OCTALPLUS422, are installed in the VME crate and connected to the FMCM through a twisted pair cable. A short introduction to the RS-422 standard, and a more detailed description of the installation of the

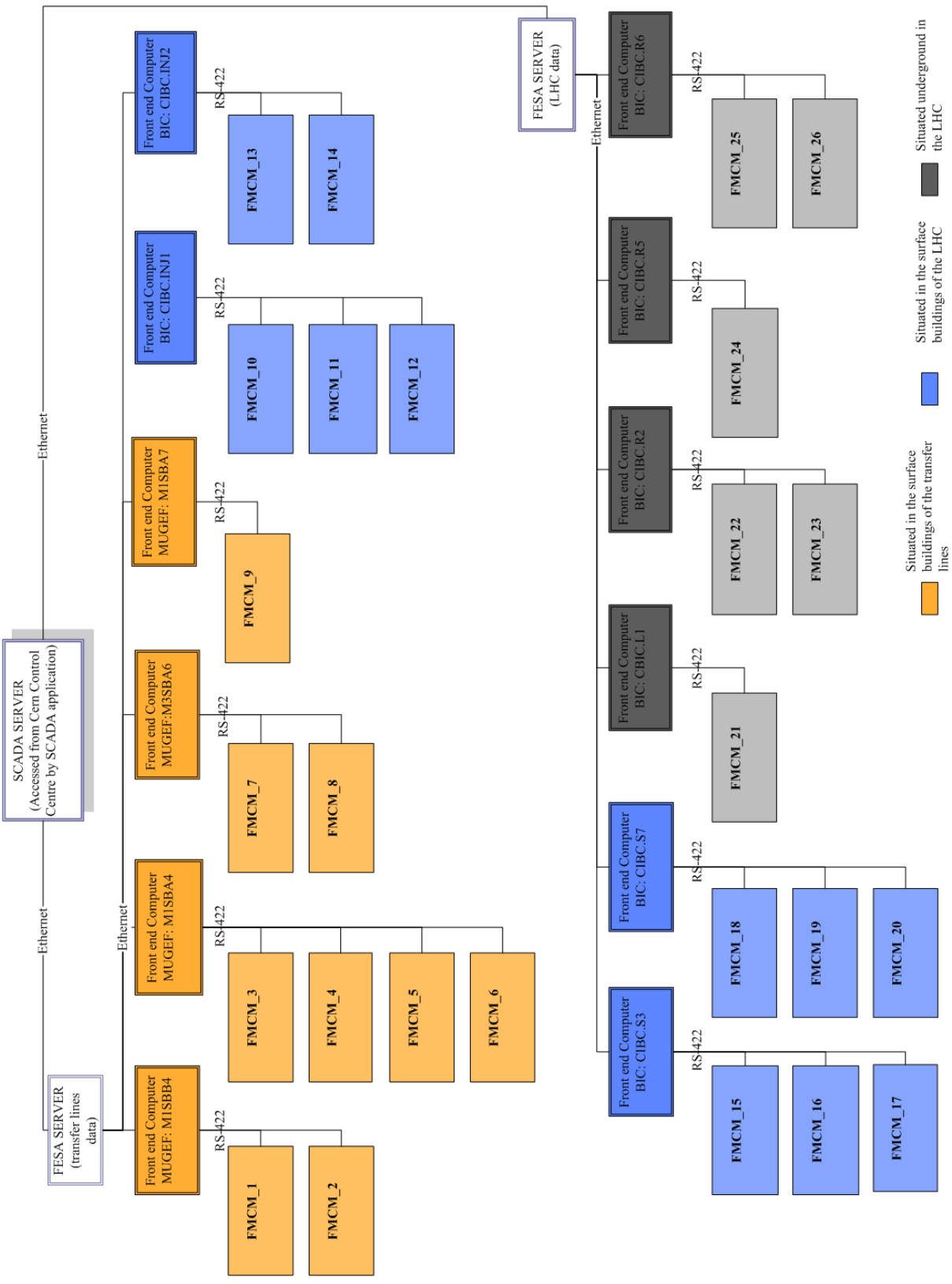

Figure 4.1: The FMCM network

serial interface on the VME side is given in Appendix A.

Figure 4.1 is an overview of the future installation of FMCMs in the LHC and the transfer lines, and shows how the different units are connected to the higher level modules of the control interface. There will be 26 FMCMs installed in the LHC and the SPS-LHC transfer lines, and a maximum of four FMCMs will be connected to the same front end computer. A further description of the installations concerning hardware, cabling and connections to other systems is given in Appendix B.

The communication between the different FMCMs and the front end computers will be realized by software implemented in the front end computer. The set of commands and messages going over the RS-422 interface, and the rules for when and how they will be transmitted is defined in the system specification, and most of them are already implemented in the current prototype of the FMCM. One of the restrictions given by the LHC project is that all front end computer software shall be developed in the Front End Software Architecture (FESA) framework. The purpose of the framework is to make consistent rules for how front end software is implemented for equipment used by the Accelerator and Beams department at CERN. An introduction to the FESA framework is given in Appendix C.

The front end computer will store the information retrieved from the different FMCMs in a FESA server. This information can then be accessed by a Supervisory Control And Data Acquisition (SCADA) system. The higher level development of the control interface is outside the scope of this thesis, however the chosen implementation of the front end computer software will define the requirements and the limitations for the SCADA application. This application will give the operators situated in the CERN Control Center (CCC) access to the acquired data and diagnostic tools for analysis.

4.1.1 Data Acquisition and Signal Processing in the FMCM

The most important data acquired by the FMCM is the Post Mortem (PM) data, which contains the voltage and current recordings of the protected magnet around the time of failure. In case of an event (either self-triggered or triggered by the global PM trigger distributed via the LHC timing system, see Section 4.1.2), the FMCM will freeze the internal buffers with the PM data and register the PM acquisition with a time-stamp. There are four different versions of the PM data available in the FMCM:

- The measured magnet voltage: U_Mag
- Simulated magnetic field changes: I_Diff_Sim
- Direct Current Current Transformer (DCCT) current changes: I_Diff_DCCT
- External Voltage: U_Ext

The resolution of the analog to digital converter (ADC) is 12 bits per sample for the mentioned signals, which requires 2 bytes of memory per sample. A data set is decided to be 2000 samples per signal.

12 bits	2 bits	1 bit	1 bit
Data	Zeros	Optical isolated trigger	FMCM alarm output

Table 4.1: 1 sample of PM data

Two versions of the PM buffers will be implemented due to the different characteristics of the magnets in the LHC and the transfer lines. The fast ramping magnets in the transfer lines will be measured with the highest resolution (one sample every 21.33 μs), during a time window of 40 ms (30 ms before the event and 10 ms after the event). For the slow ramping magnets in the LHC the recordings will be done with a lower resolution (one sample every 42.66 μs) and a consequently longer time window of 80 ms (60 ms before the event and 20 ms after the event).

To reach the desired number of particles in the LHC, the proton beams are injected through the transfer lines in several steps (injection in so-called batches). The time interval between two injections is approximately 20 seconds, depending on the cycle-time of the pre-injections. In the worst case, this means that one may encounter new events in the FMCM along the transfer lines every 20 second. In the LHC however, being a slow cycling machine with a long filling process and operational cycle, things are more stable and the interval between possible events is longer, see Figure 3.6 and Figure 3.5 for comparison. Every time a new event is registered, old PM data will be lost. Thus, readout of PM data must be ensured within the interval between consecutive events. To summarize, it is more time-critical to read out the PM data from the FMCMs in the transfer lines than in the LHC.

The transmission time of a PM data set is expected to be about 1.2 seconds at a transmission rate of 38.4 kbps (available in the current prototype of the FMCM). The FMCM is decided to operate at 115.2 kbps, which allows for retransmission of data in the event of transmission errors between the FMCM and the front end computers. At the same time this transmission rate is expected to be low enough to ensure reliable communication while transmitting over long cables.

The FMCM also keep records of important information about the current state of the device. This information can be used for both diagnostic purposes and also to detect a failure before it actually happens.

4.1.2 The FMCM Timing Interface

The purpose of the FMCM timing interface is to make the internal clock in the FMCM synchronous with the LHC machine and the Universal Coordinated Time (UTC). The FMCM will use the internal clock to time stamp any occurring events and the PM data. For external events, that is events happening somewhere else in the LHC, the FMCM will be notified by a timing signal (called the PM trigger) coming from a CTRP (Control Timing Receiver PCI) card. Figure 4.2 shows the three signals involved in the FMCM timing process.

Signals from the CTRP card:

- *PM trigger:* 1 negative pulse of 2 μs. The PM trigger signal indicates an external event in the LHC machine. Such an event will require PM data to

be sent from the FMCM to the VME front end computer, which again will forward this data to the CERN control system.

- *Pulse per second (pps):* 1 Hz signal with accuracy in the range of some nanoseconds, which is used to synchronize the internal clock in the FMCM with the LHC machine timing.

Signal from the front end computer:

- *UTC time:* A timing message of 32 bits, which indicates the absolute UTC time with 1 second accuracy. This message is used to synchronize the internal clock in the FMCM upon arrival of the next timing pulse (pps) to the specific date and time (UTC).

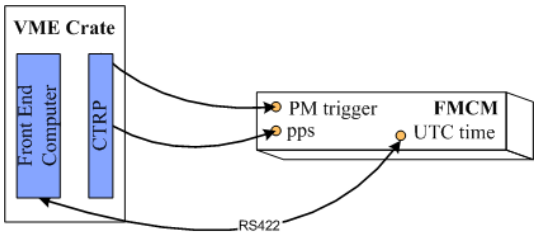

Figure 4.2: The FMCM timing interface

A new prototype of the FMCM arrived at CERN in end of January 2007, and tests were done to verify the behavior of the timing interface and to make an analysis of the accuracy in reference to the PM time-stamp. A presentation of these tests and the results are given in Appendix D.

4.1.3 The FMCM Control Interface and Network Architecture

Serial data communication is not strictly a network communication protocol, but the design of the control interface can be compared to the Open Systems Interconnection Basic Reference Model (OSI model) to show how the communication is implemented in different layers [13].

Layer	Function	Data Unit
Application Layer	SCADA Application	Data
Transport Layer	Front End Software	Packets
Data Link Layer	UART and driver	Frames
Physical Layer	RS-422	Bits

Table 4.2: The Network Architecture

As one can see in Figure 4.1, the distribution of FMCMs is more a communication tree then a network. The different FMCMs can not communicate with each other and in fact they are under the strict rule of their proper front end computer. A FMCM can only send a message to its front end after the front end computer has specifically ordered it to do so. The front end computer retrieves information from

the FMCMs connected to it, stores the data in the FESA server, which again the SCADA application can use to monitor the device. Thus, as there is no need to route the messages, the network layer is extracted from the model. Also for simplicity, the transport, the session layer and the presentation layer has been combined to one single layer and called the transport layer.

The Physical Layer

The physical layer is concerned with transmitting raw bits over a communication channel. It defines the mechanical, electrical and timing interface to the network. The physical link chosen for the FMCM control interface is a shielded twisted pair cable with a differential serial interface, namely RS-422. The RS-422 standard together with a description of the hardware chosen for this interface is described in Appendix A. For the final version of the FMCM the transmission rate will be set to 115.2 kbps, whereas for the current prototype the available speed is 38.4 kbps.

The Data Link Layer

The main task of the data link layer is to transform the raw bits into a stream of data that appears free of errors to the upper layer. It operates with data in the form of frames. The data link layer defines algorithms for achieving reliable, efficient communication between to network nodes. The defined transmission protocol over the serial link sends one byte at a time encapsulated by a start bit, a parity bit and a stop bit, see Table 4.3.

1 bit	8 bits	1 bit	1 bit
Start bit	Data	Odd parity bit	Stop bit

Table 4.3: The transmission protocol

The Universal Asynchronous Receiver and Transmitter (UART) translates data between parallel and serial interfaces, see Appendix A.2, and converts bytes of data to and from asynchronous bit streams. In addition, each UART channel has a 64 bytes FIFO buffer for both transmission and receive, flow control, handling of special characters and a programmable bit rate up to 1.5 Mbps. It also performs error detection by computing the parity bit and comparing it with the expected value.

A driver is needed to interface the UART with the front end computer. It is required to use a general driver developed by the front end section at CERN, as similar hardware for serial communication is used by several systems.

The Transport Layer

The transport layer provides transparent transfer of data between end users and control the reliability of a given link through flow control, segmentation/desegmentation, and error control. The transport layer can keep track of the packets and retransmit those that fail.

The transport layer of the control interface is defined as the set of commands and messages going over the RS-422 interface, and the rules for when and how they are transmitted. The front end computer will manage the communication with the FMCMS according to the system specification [12]. The rest of this section describes the different commands and messages and the rules of transmission which all together will be the requirements for the front end software. The design and implementation of the software is presented in Section 4.2.

Six different commands will be sent from the front end computer to the FMCM. The structure of the general command message is as shown in Table 4.4.

Byte	Description	
0:9	Synchronization	10 times '0d'(hex)=cr(ASCII))
10	Start of command	'2a'=*
11	The command code	See Table 4.5
12:17	The command arguments	See Table 4.5
18:19	The command checksum	Sum of command code and command arguments as unsigned chars +55aa, truncated to 16 bits

Table 4.4: The command (20 bytes)

The different commands are defined by the command code (1 byte) and the corresponding command arguments (6 bytes), see Table 4.5.

Command code	Description	Command arguments	
p	Post Mortem	1 byte	'30'=0= U_Mag '31'=1= U_Ext '32'=2= I_Diff_Sim '33'=3= I_Diff_DCCT
		5 bytes	'30 30 30 30 30'=00000
t	Update UTC	4 bytes	UTC as UNIX time stamp
		2 bytes	Empty
d	Simulate Dump	1 byte	'31'=1 (ch1 only) '32'=2 (ch2 only) '33'=3 (ch1 and ch2)
		5 bytes	'44 55 4d 50 21'=DUMP!
r	Reset Alarm Counter	1 byte	'31'=1= Reset pre-alarm counter only '32'=2= Reset alarm counter only '33'=3= Reset both counters
		5 bytes	'30 30 30 30 30'=00000
i	Idle	6 bytes	By choice
s	Status	6 bytes	'30 30 30 30 30'=00000

Table 4.5: Command code and command arguments

When the FMCM receives a command it replies with a message, and the front end computer always waits for the message before sending a new command. The *status* command is a request to the FMCM to send a *status* message, containing information regarding the current state of the FMCM, configuration settings and so on. The *update UTC* command includes the timing message of 32 bits described in Section 4.1.2. When the FMCM receives this command it will synchronize the internal clock to the pps and the UTC, and send a message back to the front end. The *simulate dump*, *reset alarm counter* and *idle* command are mainly intended for test purposes. When receiving a *simulate dump* command, the FMCM will send a beam permit signal to the CIBU. The *reset alarm counter* command will tell the FMCM to reset the alarm or pre-alarm counter which stop counting when they reach their maximum value of 65535. The *idle* command could serve for several purposes and its use in the communication process has not been exactly defined yet. Upon arrival of a *PM* command the FMCM will read the PM data in its internal buffer and include the data in a *PM* message to the front end computer.

Byte	Description	
0	Synchronization	1 times '0d'(hex)=cr(ASCII))
1	Start of command	'2a'=*
2	The command code	See Table 4.5
3:8	The command arguments	See Table 4.5
9:10	The command checksum	Sum of command code and command arguments as unsigned chars +55aa, truncated to 16 bits
11	Error bits	Bit 0= parity error Bit 1= Framing error Bit 2= Unknown command Bit 3= Unexpected command argument Bit 4= Checksum error Bit 5:7= 0 (spare)
12:18	Current time stamp	4 bytes for integer seconds 3 bytes for fractional seconds
19	Information bits	Bit 0= LHC mode. Last PM acquisition was triggered by external trigger Bit 1= Time stamp message received, and next pulse on UTC input will update time. Bit 2= Time stamp initialized Bit 3= Presently no reliable time stamp Bit 4= PM flag. Inverted for each new PM data set. Bit 5= Logic state of time stamp synchronization input (Input high = 1) Bit 6:7 = 0 (spare)
20:26	Time stamp of last PM acquisition	4 bytes for integer seconds 3 bytes for fractional seconds
27	Spare	0

Table 4.6: The response header (28 bytes)

The *update UTC* and *status* commands will be sent periodically, whereas the *simulate dump*, *reset alarm counter* and *idle* command will be transmitted on request by an operator. The *PM* command will be transmitted only when the time-stamp of the last PM acquisition in the *response header* of one of the other messages changes, see Table 4.6.

Byte	Description	
0	FPGA configuration.	Version 1:255, 0=unknown
1:3	Up time in minutes	24 bits
4:5	Pre-alarm threshold set by potentiometer	
6:7	Alarm threshold set by potentiometer	
8:9	Alarm counter	0:65535, stopped at max value
10:11	Pre-Alarm counter	0:65535, stopped at max value
12:13	Actual Magnet Voltage	1 sample
14:15	Actual Voltage at connector U_Ext	1 sample
16:17	Actual calculated field change I_Diff_Sim	1 sample
18:19	Actual calculated DCCT current change I_Diff_DCCT	1 sample
20:21	Field deviation	Minimum of last minute
22:23	Field deviation	Maximum of last minute
24:27	Difference between old and new time stamp	
28	ID and configuration status	Bit 0:5= Device ID Bit 6= Magnet type (LHC=1,TL=0) Bit 7= 1 if alarm activation below 5% is demanded
29	Device status	Bit 0= Current alarm status channel A(0=user permit for CIBU) Bit 1= Current alarm status channel B(0=user permit for CIBU) Bit 2= Opto-coupled input(1=high) Bit 3= TL mode: Alarm was active during last extraction trigger. If YES: PM data should be read If NO: data can be read optionally to compare with data of faulty magnets.
30:31	Spare	

Table 4.7: The status data (32 bytes)

The message from the FMCM is built up by the following elements: The *response header* with 28 bytes, the data which can have various sizes, the *response checksum* and the *trailer*. The *response checksum* is defined as the sum of the header bytes and all data bytes referenced as unsigned chars, plus 55aa(hex), truncated to 16 bits. The *trailer* marks the end of the packet using 2 bytes: '3c 3e'(hex)= <>(ASCII).

Only two of the six commands generate a response containing additional information apart from the header. Those two are the *PM* message and the *status* message. The *status* message consists of 32 bytes and the structure of the data is as shown in Table 4.7. This information includes basic status data and configuration settings that can be used for diagnostic purposes. In the *PM* message, the 2000 samples of PM data are transmitted in consecutive order with one sample being two bytes as indicated in Table 4.1.

Figure 4.3 summarizes the different types of commands and messages that are sent over the RS422 interface.

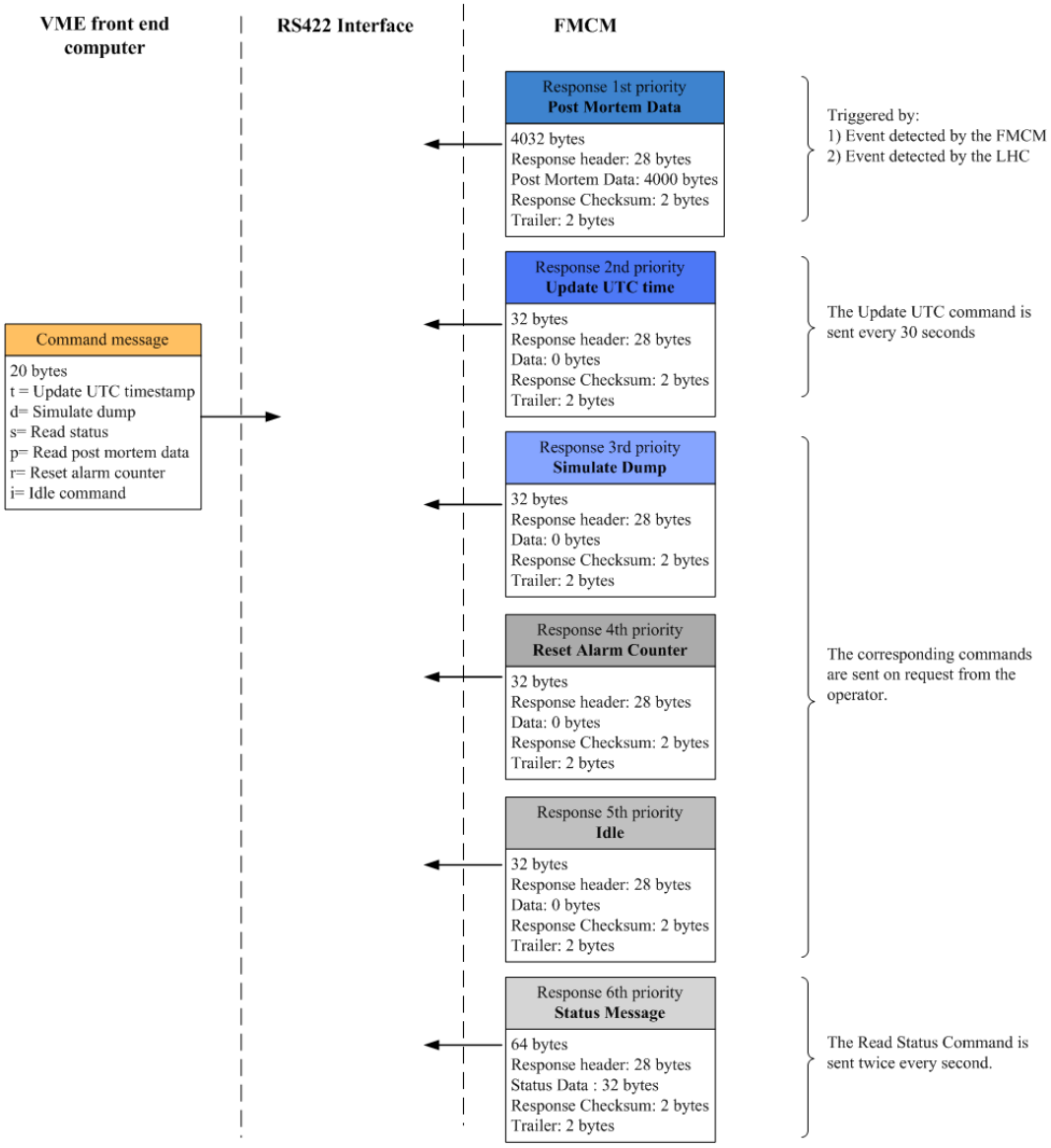

Figure 4.3: Messages transmitted to and from one FMCM

As the structures of the different commands and messages have been defined it is time to establish the rules for when and how they are transmitted:

- The front end computer is always the master of communication. No data will be transmitted by the FMCM without the front end computer having requested it to do so. A command from the front end is always followed by a response message from the FMCM.
- The FMCM will be requested by the front end computer to send a *status* message twice every second.
- The *update UTC* command will be sent to the FMCM every 30 second.
- If the FMCM expects the start of a command and receives undefined data, it answers with '?'.
- There will be a timeout if a command to the FMCM is not completed after 0.2 seconds.
- If the time stamp of the last PM acquisition in the response header changes, all other data transmission will be temporarily suspended and readout of the PM data will be requested by the front end computer.
- The front end computer can only request PM data from one type of signal at the time (U_Mag, I_Sim_Diff, I_Diff_DCCT, U_Ext).
- The RS-422 interface on the front end side must be able to receive a block of at least 4096 bytes at 115.2 kpbs without buffer overflow.
- The system has to be able to manage the case when 26 FMCMs transmit PM data at the same time, and up to four FMCMs for a single front end.
- Error handling procedures have to be part of the communication process in order to ensure the readout of data from the various FMCMs, see Section 5. If data is lost, alarms should be sent to the operators and the SCADA system.
- Recommended priority of transmission is:
  1. *PM* message
  2. *update UTC* message
  3. *simulate dump*, *reset alarm counter* and *Idle* message
  4. *status* message

The Application Layer

The Application layer provides the user with an application to access information on the network. This layer is the main interface for the user to interact with the application and the network.

The SCADA application for the FMCM control interface will be implemented by the Application Section at CERN. As mentioned in the introduction, the objective is to deliver an overall supervision application from which one can gain access to the acquired data in the FMCM, present the data on a console in the CCC and remotely supervise the device.

4.2 Implementation of the Front End Computer Software

4.2.1 Software Design

The front end software must fulfill all the requirements mentioned in Section 4.1.3. A first attempt at understanding how this could be achieved resulted in the flowchart in Figure 4.4. The flowchart sums up the most important tasks of the program, their dependencies and priorities. The diamond shaped boxes are decisions regarding the tasks represented in the square boxes. The arrows show the program flow. The task with the lowest priority is the request and transmission of status data. At the same time, this is the most frequent one. In normal operation, the *status* message is likely to be the message that contributes with most information about the current state of the FMCM. The next frequent task is the request and transmission of the *update UTC* message which is transmitted every 30 seconds and has the second highest priority. These two tasks will run continuously during normal operation and most likely be the ones to trigger the *PM* command. The three remaining commands *simulate dump*, *reset alarm counter* and *idle* message have an intermediate level of priority and will be mostly used for test purposes.

The following occurs during the execution of a programme: A *status* command will first be sent to the FMCM, the program waits for the response and checks if it is valid. If the transmission takes too long the procedure will be repeated. The *response header* of the incoming message is checked to see whether or not the PM time-stamp has changed, if so, then the sending and receiving of PM data begins. Depending on the parameters set for the *PM* command, the front end will either request PM data in form of the measured magnet voltage (U_Mag), simulated magnetic field changes (I_Diff_Sim), DCCT current changes (I_Diff_DCCT) or the external voltage (U_Ext). Again the program will wait for the response, check if it is valid and retransmit if the procedure takes too long. In normal circumstances, the program will then continue to ask for status data, until the first *update UTC* command is triggered after 30 seconds.

The *update UTC* command will have to wait if the *PM* command is triggered at the same time, and the *status* command will have to wait if any of the other commands are triggered at the same time. Only one message can be processed at once, and commands that are triggered during the processing of another message must wait until the proceeding message has been successfully received or aborted. In the meantime the pending tasks are stored in a First In First Out (FIFO) queue.

4.2.2 Software Design in the FESA Framework

To transfer these ideas into the FESA framework, the starting point was the FESA Design tool, see Appendix C. The FESA Design tool is a wizard for modeling equipment-software. The resulting equipment model contains Properties, Server Actions and Real-time Actions, User Events and Timer Events, which together reflect the system specifications and define the software structure. This model is in fact an XML (eXtensible Markup Language) file which is finally translated into a FESA C++ project. The equipment model is shown in Appendix C.1. Figure 4.5

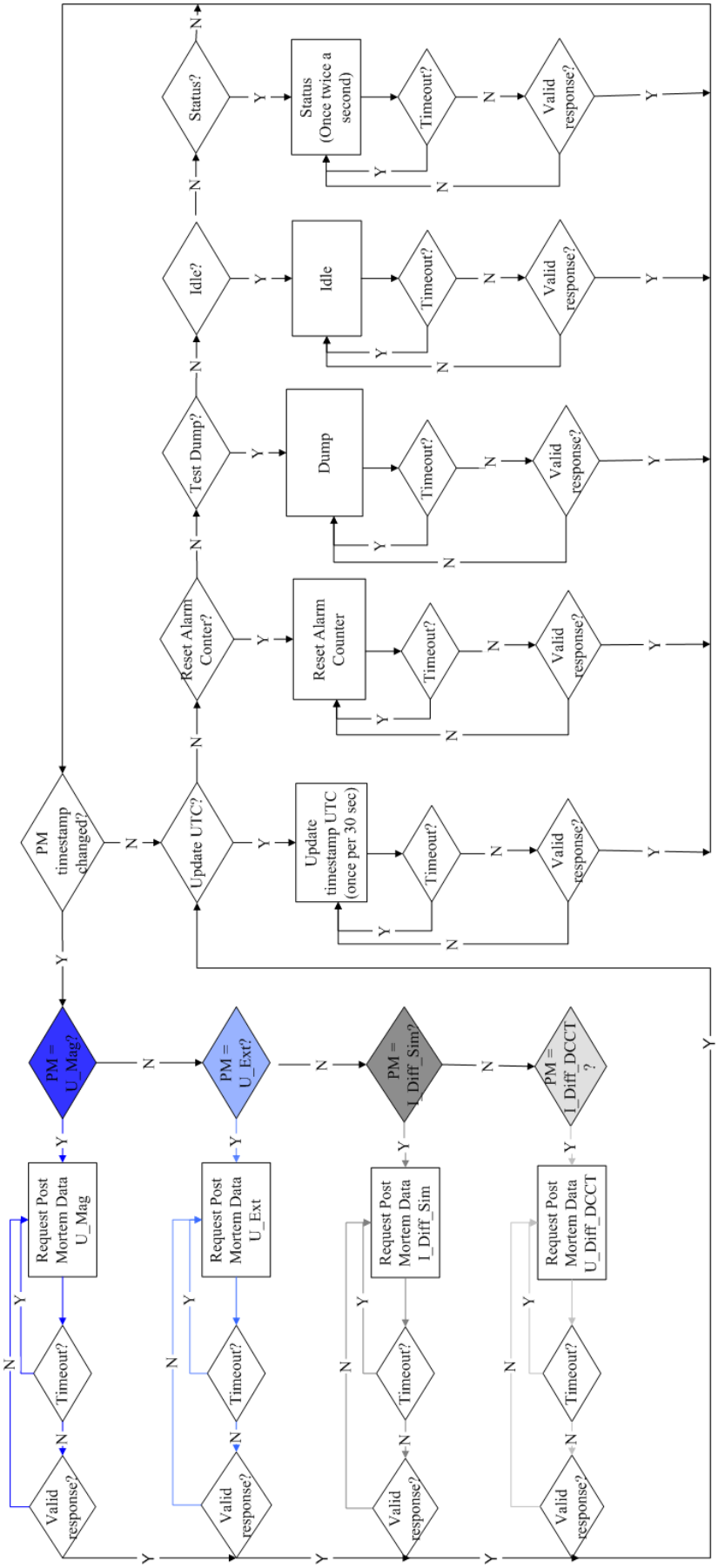

Figure 4.4: Flow Chart of front end computer software

shows the resulting software design and the relationships between the different elements in the equipment model and how they interact. The different pairs of commands and messages which are being sent over the RS422 interface are defined as actions. The request and receiving of *UTC* and *status* data are real time actions, whereas the transmission of the *idle* command, the *reset alarm counter* command and the *simulate dump* command with the corresponding responses are merged into one server action. The request and receiving of *PM* data is implemented as a C++ class, independently of the equipment model, but has the same functionality as the other real time actions. Each action is related to a property which is responsible of keeping the information of the data fields that are changed due to that action. While the program is running, the properties are displayed in the FESA Test tool and can be accessed by an operator.

In Figure 4.5 the property SetCommand is set by the operator, and this triggers the corresponding server action. By entering different values in the belonging datafields before setting the property, the operator can choose to send an *idle* command, a *simulate dump* command or a *reset alarm counter* command. In the server action the function call fireUserEvent() fires the user event which triggers the RT action SetCommandRT. Depending on the command chosen by the operator, SetCommandRT will write the corresponding command to the RS422 link, wait for the response and read the *response header*, the *response checksum* and the *response trailer*. SetCommandRT is linked to four properties (Command, ResponseHeader, ResponseChecksum and ResponseTrailer) whose data fields will be updated when the real time action is finished. The inclusion of the Command property is not really necessary to fulfill the requirements, but is a nice feature for test purposes.

The RT action UTCCommand is triggered every 30 seconds by the timer event TickUTC. It writes the *update* UTC command to the RS422 interface, waits for the response and reads the *response header*, the *response checksum* and the *response trailer*. UTCCommand is linked to the same properties as SetCommandRT, except for the Command property.

The RT action StatusCommand is triggered twice every second by the timer event TickStatus. It writes the *status* command to the RS422 interface, waits for the response and reads the *response header*, the *status* data, the *response checksum* and the *response trailer*. StatusCommand is linked to the same properties as UTCCommand, but has in addition the property StatusData.

The PMcommand can be triggered by all the other real time actions if the PM timestamp in the *response header* changes. It is also linked to the same properties as UTCCommand, but has in addition the property PMData.

The communication over the RS422 interface is supported by the implementation of another class called TestFMCMRT. It establishes the connection with the FMCM, and has write and read functions which are used by the real time actions to access the RS422 interface. A collaboration diagram of the different C++ classes is shown in Figure 4.6. It is important to note that these classes constitute a small part of the FESA project generated from the equipment model. However, they are the only ones that are implemented by the developer and they contain the most important functionality. For example, the priorities of the different actions are handled by a scheduler that is automatically generated. The developer can define these priorities in the Equipment model, but is not allowed to change the code later on. The

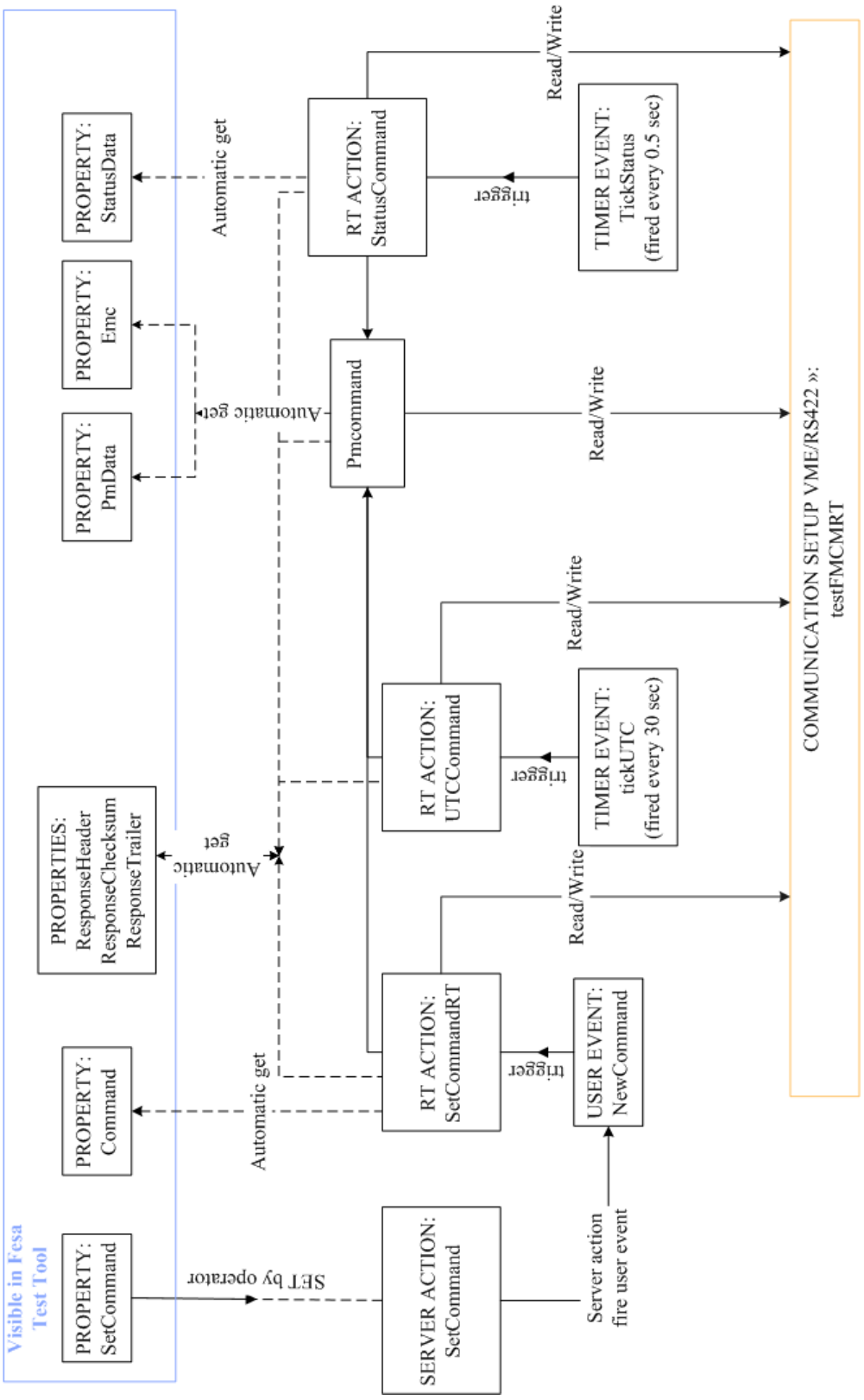

Figure 4.5: FESA Front end computer software design

connection between a RT action and a Timer Event is also automatically generated.

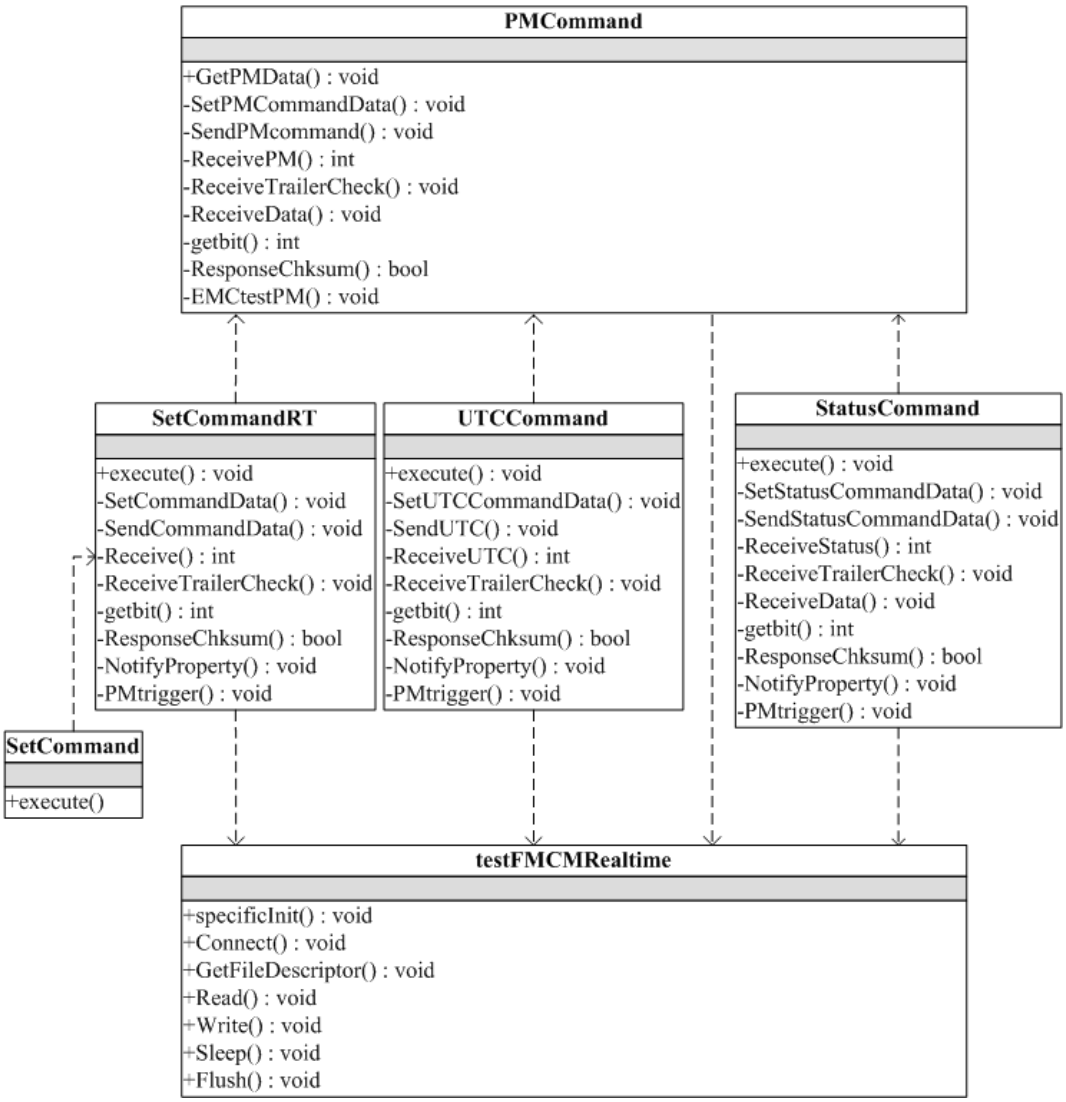

Figure 4.6: Collaboration diagram of the implemented C++ classes

Chapter 5

Error Handling

In a communication system there is noise present in all practical situations. Noise is defined as any signal interfering with the message being sent. It can be another unwanted message or a random fluctuation in the signal level. As a result of the physical processes generating the noise, errors tend to come in bursts. Having the errors arriving in bursts has both advantages and disadvantages over single-bit errors. In data transmission, data are always sent in blocks of bits. If errors were independent most blocks would contain an error, but if the errors came in bursts only a few blocks would be affected in average. The disadvantage of burst errors is that they are much harder to correct than isolated bit-errors. A further discussion of the noise sources in the LHC is given in Chapter 6.

The physical layer and the data link layer were specified from the beginning of this project. In fact the decisions regarding the choice of technologies were a direct consequence of the design of the FMCM and politics at CERN. Due to this fact, there are certain limitations as to what can be done to improve the error handling. To be able to correct bit errors it would be necessary to add some functionality in the UART or in the driver. The UART is an of-the-shelf merchandise together with the other RS-422 related hardware and is therefore difficult to change. The library and driver for the RS-422 communication is developed and maintained by the Front End Section at CERN. As many applications rely on this driver it is difficult to introduce new functionality. This leaves the front end software, which only concerns itself with packets of bytes. The only error detection on bit level is the odd parity check in the UART.

5.1 Error Detection and Correction in Serial Communication

There are two basic strategies for dealing with errors. One way is to include enough redundant information along with each block of data, to make the receiver able to deduce what the transmitted data must have been. The other way is to include just enough information to make the receiver able to detect that an error has occurred, but not which error or how many. The receiver will then request a retransmission. The two strategies are discussed in this section together with a description of the

chosen implementation concerning the front end computer software.

5.1.1 Error Detection

Error detection can ensure reliability by causing retransmission of blocks of data until they are correctly received. It is not even necessary to detect all the errors as one single error will require retransmission of the entire block. Error detection is quite efficient when the errors come in bursts.

A well known error detection method is a retransmission scheme based on a checksum calculation. The sender calculates a checksum for the block of data and appends it at the end of the block. When the receiver receives the block it calculates exactly the same checksum and compares it to the checksum at the end of the block. If the two checksums match, the entire block is considered to be correct. However, if the checksums do not match the receiver discards the block of data and it is retransmitted by the sender.

Odd Parity Check

Odd parity check is a simple error-detecting method, where a parity bit is added to the data. The parity bit is chosen to make the number of 1 bits odd for a given codeword (sequence of bits). For example, when 1011010 is sent in odd parity, a bit is added to the end to make it 10110101. With even parity 1011010 becomes 10110100. The receiver will perform the same calculation and check if the number of 1 bits is in fact odd. If so, the last digit is removed which leaves the original codeword. If the number is even, an error has occurred and the receiver will ask the sender to retransmit the data.

Odd parity check is implemented in the UART and in the field programmable gate array (FPGA) of the FMCM.

Error Handling in the Front End Computer

The FMCM sends messages to the front end computer as responses to commands. The FMCM calculates a checksum for each message and sends it together with the trailer at the end of the message. The front end calculates the same checksum upon arrival of a new message and compares it with the checksum calculated by the FMCM. The *response checksum* is defined as the sum of header bytes and all data bytes referenced as unsigned chars, plus 55aa(hex), truncated to 16 bits. A bit error during transmission will make the two checksums different, and the front end computer will send the command once more to make the FMCM retransmit the same message.

A counter is implemented to prevent an infinite loop of retransmissions, see Figure reffig: sekvensdia. The total number of times a message can be retransmitted depends on how often the *update UTC* command is sent and the time interval between events. If a new event occurs before the old PM data has been successfully received, one would no longer want to keep retransmitting the old PM data. Further, it is not desired to delay the *update UTC* command as this would leave the

time-stamp unsynchronized. Thus, the sequence of retransmissions for any message should take less time than the interval between the *update UTC* commands or the *PM* commands, whichever has the shortest interval. Also, the *status* message, which is sent most frequently, should be successfully received before the next *status* message is scheduled. If the counter expires the front end computer will try to store the message anyway and send an alarm to the operator stating that the message is not valid.

The checksum does not reveal whether only one or several bytes of the received data are corrupted. To discover wrong conditions for transmission, such as buffer overflow or a disconnected wire, the front end also checks the number of received bytes to make sure this number matches the type of message it is supposed to have received. It also checks if the command code is in the right place in the message, to make sure the transmission was synchronized.

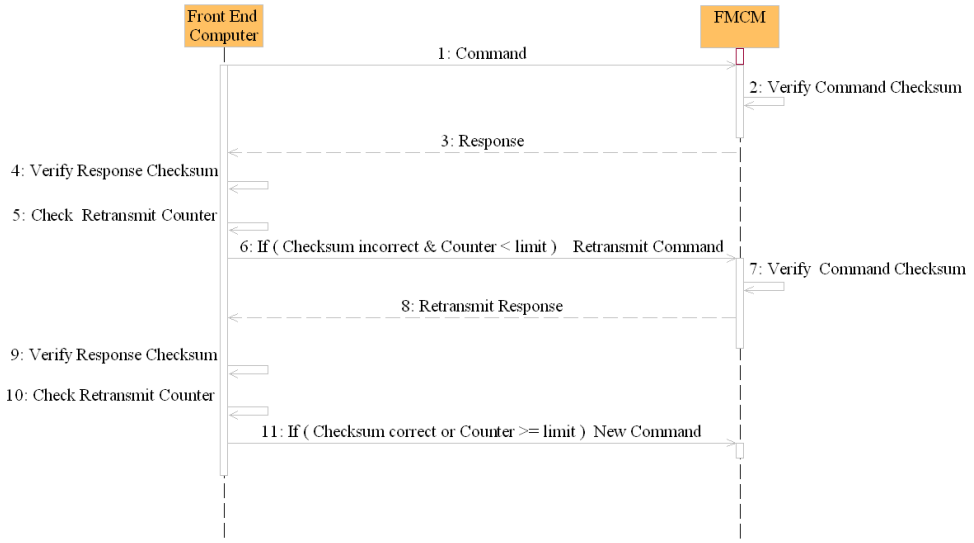

Figure 5.1: Sequence diagram of implemented error handling

Error Handling in the FMCM

A checksum is also included in the command sent to the FMCM. The purpose of the checksum is to validate the incoming command especially to prevent the case where an error in the bit stream would produce a dump of the LHC beam by activating the simulate-dump feature inside the FMCM.

5.1.2 Forward Error Correction

The deliberate addition of redundant bits in order to correct possible errors is called Forward Error Correction (FEC). A simple example of a FEC code would be an analog to digital converter that samples three bits for every transmitted bit. If two or more samples are zero, the transmitted bit is assumed to be zero, and if three samples are all one, the transmitted bit was probably a one. FEC codes can be

divided into two groups, block codes and convolutional codes. A block code maps a block of k information bits into a block of n coded bits, where $n>k$. The n coded bits only depend on the k source bits and is therefore said to be a memoryless coding scheme. Such a block code has code rate k/n and is referred to as a *(n,k)* code. A convolutional coder also produces coded bits at a higher rate than the source bits, but it does it without dividing the source bits into blocks. The name refers to the fact that the added redundant bits are generated by modulo-two convolutions. A convolutional coder is a finite memory system and implemented by use of m delay elements and modulo-two adders. The transformation from information bits to coded bits is a function of the last m bits in the stream. Some examples of block codes are Hamming codes, Reed-Solomon codes and LDPC codes. The Turbo algorithm uses two simple convolutional codes with interleaving. [14, 15]

The advantage of FEC is that retransmission of data often can be avoided, at the cost of higher bandwidth, and is therefore applied in situations where retransmission is relatively costly or impossible. FEC devices are often located close to the receiver of an analog signal, in the first stage of digital processing after a signal has been received. That is, forward error correction is often an integral part of the analog-to-digital conversion process. It would be feasible to implement error correction in the FPGA of the FMCM. The problem is on the other side of the link as it would be preferable to have this functionality in the UART, or at least in the software controlling the UART (driver). However, as this is not possible it would have to be implemented in the front end software, which would be a poor solution. The front end software receives bytes from the RS-422 interface and treats them as characters. The characters would then need to be split into bits and reassembled again after the error correction. If the results of the tests in Chapter 6 shows that retransmission is too costly, one would rather choose to change the hardware than to implement error correction in the front end computer. [14, 13, 16]

Part III

Testing & Analysis

Chapter 6

Tests

Primarily three tests will be done on the FMCM control interface: Electromagnetic Compatibility (EMC) tests, Signal Integrity (SI) analysis and an endurance run with error handling tests. The EMC tests and signal integrity analysis will show whether or not the RS-422 serial interface can handle the noisy environment it will operate in. The EMC test will be done twice, first without any error handling and later after implementation of the error detection it will be repeated to verify an increase in the performance regarding reliability. The final test will be an endurance run where several events will provoke errors. These errors should then be corrected by the implemented error handling scheme. The endurance run will also be an overall test to see if both hardware and software of the FMCM control interface is working correctly.

This chapter starts with a discussion over the most probable noise sources in the LHC and transfer lines that can affect the RS-422 link. This is mostly related to the EMC test. Background theory on the EMC test and SI analysis is given together with a description of the corresponding test plan, test setup and test requirements. The results of the tests will be presented in Chapter 7.

6.1 Discussion of the Environmental Conditions for the RS-422 link

The different types of disturbance in an environment like the LHC can be divided into four groups. *Mains failure* can be caused by thunder storms and short-circuits inside the electrical network at CERN. Such failures will definitely result in accelerator downtime. *Voltage dips and voltage swells* are mostly caused by sudden change of load, short-circuits inside or outside CERN and thunder storms. This might result in accelerator downtime. *Transients* are often caused by switching mode equipment, power converters (thyristors) and thunder storms and leads to failure in the electronics. *Harmonics* can be caused by non-linear loads and results in malfunctioning electronics.

Noise sources are either wide-band or narrow-band. Wide-band noise is experienced for thermal noise, noise from fluorescent lamps, brush motors and data links. Narrow-band noise can originate from a transient or an harmonic due to switching of

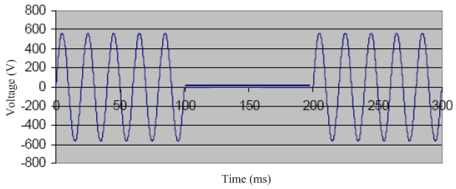

Figure 6.1: Mains failure [17]

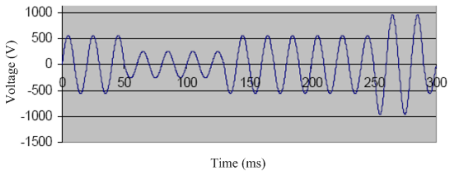

Figure 6.2: Voltage dip/swell [17]

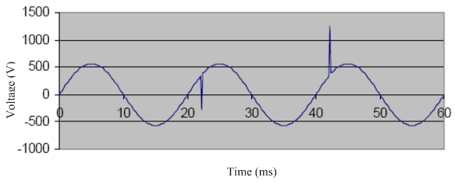

Figure 6.3: Transients [17]

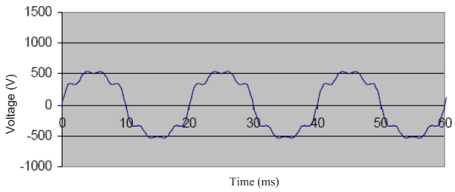

Figure 6.4: Harmonics [17]

power supplies, power lines and discharges. The equipment for the control interface of the FMCM is either installed in the LHC tunnel, or in the LHC surface buildings. The equipment installed at the different locations contribute to the electromagnetic environment. The front end analogue electronics have a bandwidth that is usually lower than 20 MHz and is sensitive to interference in this range, mostly coupled to cables in form of common mode voltage.

Impulsive Noise

The equipment most likely to cause interference in the LHC tunnel are stepping motors and power converters. A stepping motor is an electromagnetic device that converts digital pulses into mechanical shaft rotation. In the LHC stepping motors are used to position the collimator jaws and there will be about 250 collimators along the beam pipe. Each collimator is connected to four stepping motors. The stepping motors work with pulses with amplitudes of 2 to 5 Amperes and exhibit noise in the frequency band up to 20 MHz.

The noise due to power converters originates from fast switching of components such as thyristors and control electronics. The noise emission of the power converter current have peaks at 25 kHz and 75 kHz in the frequency range from 0 to 150 kHz. The power cables radiate rather strong magnetic fields in addition to radiation caused by ripple currents riding on top of the fundamental frequency. [18, 19, 20, 21, 22]

Crosstalk

If two lines carrying signals are run near to one another then by inductive and capacitive coupling the signal of one appears one the other and vice versa. This is called crosstalk. The numerous installations in the LHC tunnel requires a high density of cables. Due to the dominant noise emission from the power converters and the stepping motors, their cables are in separate cable trays and as far away as possible from other equipment and cables used for equipment control. Crosstalk

varies with frequency and proper shielding. Cable termination and earthing is the best way to limit crosstalk and cable emissions in general. However it can never be entirely eliminated.

White Noise

In electronic circuits, white noise originates from many sources, one being the slightly random motion of electrons through a conductor sometimes called thermal noise. In addition, the other equipment installed in the LHC will also emit noise that has not yet been specified nor measured. This noise is assumed to be negligible.

To meet the general requirements for electromagnetic compatibility, proper design of the equipment and proper planning of the installations are crucial. Equipment is tested before installation to make sure it fulfills the general requirements and is immune to the common noise sources in the tunnel and related surface buildings. However, it is during the design process of the equipment one can truly affect its electromagnetic qualities.

6.2 Electromagnetic Compatibility Test

> *Electromagnetic compatibility is the ability of an equipment or system to function satisfactorily in its electromagnetic environment without introducing intolerable electromagnetic disturbance to anything in that environment. [23]*

There are different levels of electromagnetic compatibility, with level 1 being suitable for equipment installed in a well-protected environment (e.g. computer room). Level 2 applies to equipment installed in a normally protected environment like computer and control rooms of industrial or electrical plants. Level 3 applies to equipment installed in an unprotected environment like public distribution networks and industrial process areas. Level 4 is the highest EMC level, where the equipment must function satisfactorily in a heavily disturbed environment. The RS-422 interface of the FMCM should be electromagnetic compatible with a level 4 environment.

The International Electrotechnical Commission (IEC) has standardized several immunity tests which are used to document the EMC level of a system [24]. These tests are divided into 6 categories by the type of disturbance experienced by the system. The first category is low-frequency disturbance which is typical for low-voltage power supplies. The second category is conducted transients and high-frequency (HF) disturbances. The four remaining categories are electrostatic charges, magnetic disturbances, electromagnetic disturbances and other immunity tests. The test chosen for the control interface of the FMCM is a fast transient burst test in the second test category. The purpose of this test is to verify the immunity of the system against bursts of very short transients. These transients are typically due to switching of small inductive loads, conducted interference and radiated interference. Significant for this test is the fast rise time, short duration, low energy but high repetition rate of the transients. The transient burst test is chosen by practical reasons, as it covers the most likely cases of disturbance and is easy to carry out. It does

not require lots of equipment or any special facilities, and has been shown to be a representative indicator for the susceptibility of equipment to any electromagnetic noise in the LHC.

The EMC test is primarily a post layout test for the RS-422 interface where a burst generator applies a test voltage on the RS-422 link. There will be no changes in the hardware design of the RS-422 interface due to the results of these tests, but encountered errors must be handled by the front end software. Errors that occur due to the transients will propagate through the system and be detected by the software. This test is therefore helpful to investigate the quality of the software and to reveal the necessity of better error handling. The free parameters of the test is the shielding technique of the cable and the amplitude of the transients. Other interfaces of the FMCM like the CIBU and timing interface will also be tested simultaneously. The test results are classified into four categories, see Table 6.1

Test Result	Description	Example
A	No Noticeable Fault	No signals are seen to be perturbed
B	Corrected Fault	Single byte errors
C	Fault	Complete messages are lost
D	Complete Failure	No communication

Table 6.1: EMC Test Result Classification

6.2.1 Characteristics of the Fast Transient Bursts

The fast transient burst is specified by the IEC standard 61000-4-4. Typical characteristics of bursts applied during EMC tests are listed in Table 6.2. The open circuit voltage of the generator corresponding to the different EMC levels are given in Table 6.3. The specifications in the standard are meant as guidelines, but can be slightly changed according to specific needs. However, any changes must under no circumstances lower the requirements on the different EMC levels. Figure 6.5 and Figure 6.6 show a typical transient pulse and a transient burst.

Rise time of a pulse (10 % - 90 %)	5 ns ± 30 %
Pulse duration	50 ns ± 30 %
Repetition frequency	5 kHz
Duration of a burst	15 ms
Burst period	300 ms

Table 6.2: Test wave characteristics [23]

EMC Level	U_{ps} (kV) (IEC standard)	U_p (kV)
1	0.25	0.5
2	0.5	1
3	1	1.5
4	2	2

Table 6.3: Recommended amplitudes for the different severity levels. U_{ps} (kV) is the voltage level recommended by the IEC standard. U_p is the open circuit voltage of the burst generator chosen for the current tests. [23]

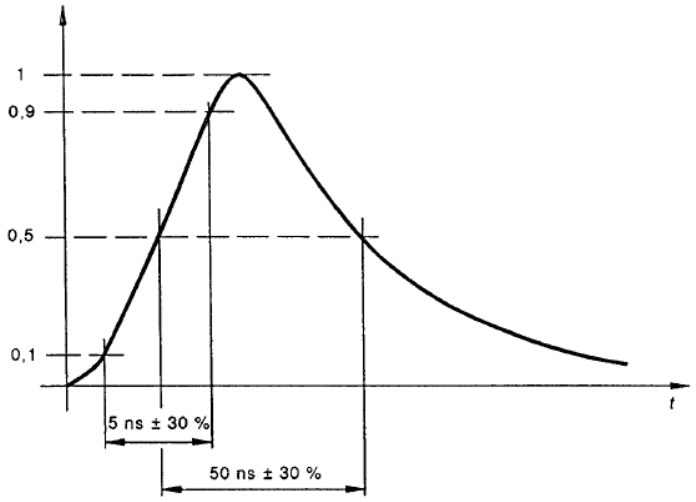

Figure 6.5: Waveshape of a single pulse into a 50 Ω load (normalized double exponential pulse) [24]

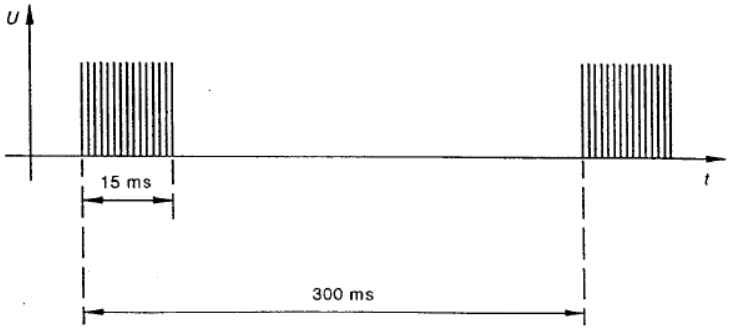

Figure 6.6: Fast transient burst with burst duration 15 ms and burst period 300 ms [24]

6.2.2 Test Setup

Test Equipment

- Fast transient burst generator (Hafely)
- Coupling device, 30 cm of conductive foil enveloping the lines.
- Textronix oscilloscope with 250 MHz bandwidth.
- Metal plate to produce common ground and reference potential.
- Shielded twisted pair cable for the RS-422 link (104 m), as will be used in the installations later on, with CANON D-SUB 9 pin connectors.
- Timing cable, CAD 50 with LEMO connectors.
- CIBU cable, NE8 with metal Burndy connectors

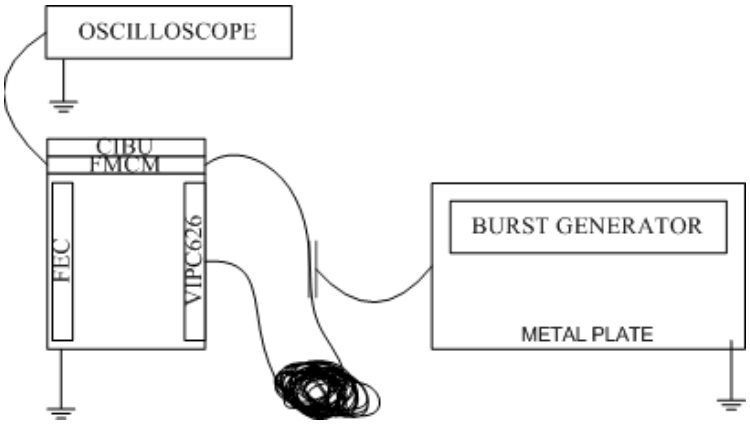

Figure 6.7: Installation of equipment during the EMC test

Figure 6.7 shows the installation of the equipment for the EMC test. A metal plate is used to provide a common ground reference. The burst generator is a device that produces voltage transients in bursts and can be programmed for the different EMC levels. A conductive metal foil is wrapped around 30 cm of the cable which is to be tested, and the voltage from the burst generator is applied to this foil. The RS-422 cable is connected between the FMCM and the VIPC626 card, the timing cable is connected between the FMCM and the timing card connected to the front end computer, and finally the CIBU cable is connected between the FMCM and the CIBU. The different signals are taken from the corresponding receiver circuits in the FMCM and analyzed on the oscilloscope. For the tests of the RS-422 interface, the transmission errors are detected by the front end software.

The Front End Software

A simplified version of the front end software will be used for the first EMC tests. The FMCM will be requested to continuously transmit PM data. One message will consist of 512 bytes, containing the integer numbers from 0 to 256. The FMCM

transmits one data sample as two bytes, and this is why the temporary test message counts to 256 and not 512. The communication is not continuous but the gap between consecutive messages will be minimized. The idea is that since the implementation of the different messages in the final software is very similar, any problems occurring with the PM message will occur with the other messages. The need for error handling will be the same as well. The software can detect any occurring byte errors and counts the number of bytes successfully received as well as the number of erroneous bytes.

For the second EMC test, an appropriate error handling procedure will be implemented, see Chapter 5. The performance of the driver will be increased so it can receive and send packets of 4096 bytes. The FMCM will be requested to continuously transmit PM data of 4000 bytes containing the numbers from 0 to 1999. As before, the communication is not continuous but the gap between consecutive messages will be minimized. In addition to the counting of erroneous bytes, the front end software will count the number of retransmission, the number of failed transmissions and further distinguish between two types of errors. The first type of error is corrupted bytes where the entire message has been received but one or several bytes have the wrong value. The other type of error is corrupted bytes that are not understood by the receiver and therefore not received at all.

The transmission rate is set to 38.4 kbps as this is the only speed available with the current prototype of the FMCM.

6.2.3 Test Plan

For the RS-422 interface the tests will be done with the following combinations of shielding techniques and transient amplitudes, see Table 6.4. The different shielding techniques are shown in Figure 6.8, Figure 6.9 and Figure 6.10. The CIBU interface and the timing interface will also be tested for the different EMC levels, with the shield of the cables grounded on both sides.

Shielding of cable	Severity level			
	0.5 kV	1 kV	1.5 kV	2 kV
Grounded on both sides Grounded on one side Not grounded				

Table 6.4: Matrix of combinations for the EMC test of the RS-422 interface

A solid shield that completely surrounds a cable can be at any potential and still provide effective shielding, as long as the cable has no connection to the outside world. Thus the shield does not need to be grounded nor to have its potential defined in any way.

In most practical cases however, the shield is not a complete enclosure, and the cable inside does have connections to the outside world. In such cases the shield must be grounded, to avoid the potential of the shield from coupling with the signal going through the cable.

Grounding has a number of additional benefits. It prevents the build-up of AC

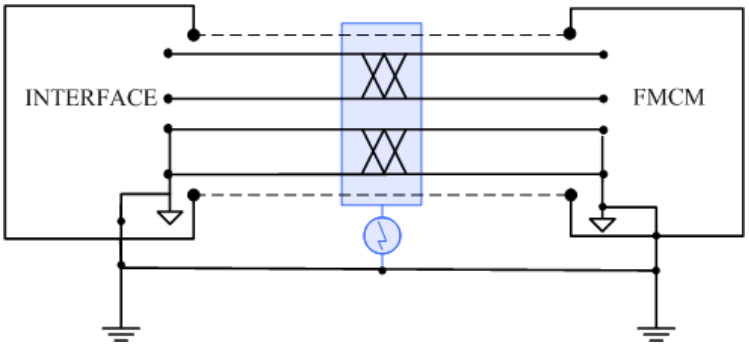

Figure 6.8: The first test uses a 360-degree shield at both ends of the cable, with two ground wires fixed at the FMCM side and the interface side.

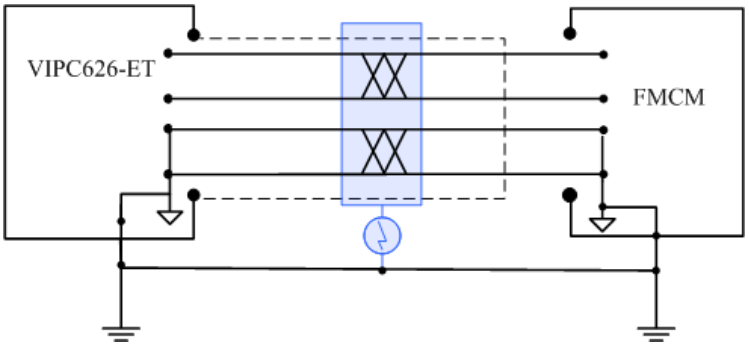

Figure 6.9: The second test uses a 360-degree shield at both ends of the cable, with the ground wire connected on the VME side.

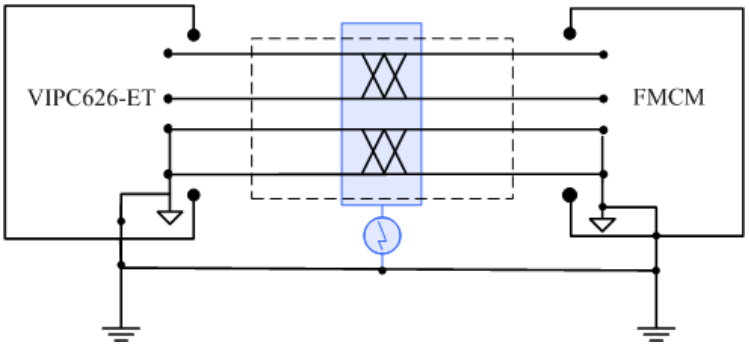

Figure 6.10: The third test uses a 360-degree shield at both ends of the cable, with the ground wires connected on neither side.

potential on the equipment enclosure, it provides a current-return path to protect personnel from shock hazards, and it prevents the build-up of static charge. For these reasons, all metallic parts of a system should be grounded, and each part must have a low-impedance contact in at least two places. [25]

6.2.4 Test Requirements

For the testing of the CIBU interface and the timing interface the requirements are quite simple. A disturbance should not change the transmitted value of the USER_PERMIT signal from the FMCM to the CIBU. The received data can be monitored on the CIBU by two LEDs blinking green if the signal is true, and red if it is false. The signal on the timing cable is a 1Hz clock signal. This signal can be monitored on the front panel of the FMCM by one green LED. In normal operation it blinks once per second. If this changes in any way it means the signal is disturbed.

For the RS-422 interface, it is expected to obtain the best results with the shielded cable grounded on both sides. The purpose of the EMC test for the RS-422 interface is to establish this fact and to analyze the severity of the errors that occur for the best choice of cable shielding.

For the second EMC test, it will be important to confirm the results from the first test, but also to verify the softwares capability to handle errors when they occur.

6.3 Signal Integrity Analysis

Signal Integrity (SI) or self-compatibility is an other aspect of EMC. In contrast to regular EMC tests which check the systems capability to handle disturbance coming from outside the system, SI analysis reveals interference within the system itself. Digital systems become sensitive to the analog nature of the signal waveform at high transmission speeds. Ideally digital signals are square waves, but in reality they are limited by finite rise times. The logic which is used to determine whether the state of a digital signal is high or low is based on thresholds. Analog effects like ringing, overshoot, undershoot and non-monotonic behavior in the transition region can change the state of the signal and create transmission errors. A way to analyze the signal integrity is to measure the opening of the eye-diagram. The eye diagram of a signal is an infinite persistent trace where the waveform is traced over the previous trace. The signal integrity of digital signals can be affected by poor layout of the printed circuit board, cross-coupling from other signals, poor shielding of the cables and by overloading a driving circuit. Unmatched transmission lines can cause unwanted reflections which interfere with the signal. All of these factors combine to shrink the opening of the eye-diagram [26].

The jitter of an eye diagram is directly linked to the signal quality from which one can predict the reliability of the data link. The eye diagram is used because it requires simple equipment and is easy to conduct compared to a Bit Error Rate (BER) test, which requires a very long test time. The total jitter t_{TJ} is measured at the line of zero differential voltage and the quantity t_S is the duration of a bit symbol as shown in Figure 6.11. Since the measurements are made with a differential

probe, which subtracts V_{OUT+} from V_{OUT-}, the eye diagram shows the differential peak-to-peak voltage defined as V_{p-p} [27].

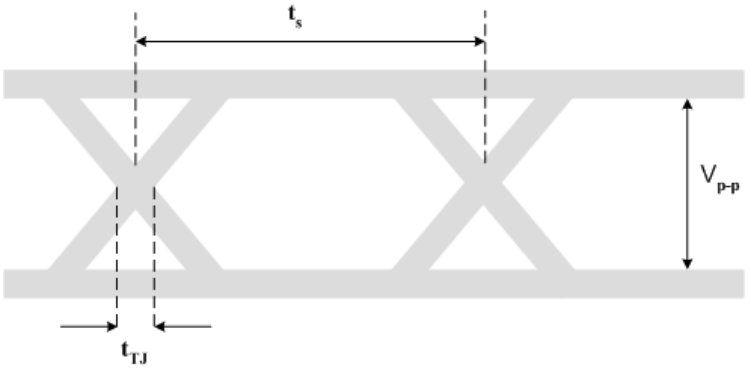

Figure 6.11: Jitter measurement at zero differential voltage line

To decide if it is necessary to perform a SI analysis on a connection one should first calculate its electrical length. When the wavelength becomes comparable to the cable dimensions, the interconnection must be treated as a high frequency design. The electrical length is defined as the ratio of the physical length l of the device to the signal wavelength λ [28]:

$$\text{Electrical length} = \frac{l}{\lambda} \tag{6.1}$$

In general, any device whose electrical length is less then about 1/20 can be considered electrically short [28]. Circuits that are electrically short can be described by basic circuit theory. On the other hand, electrically long circuits require RF techniques and knowledge of electromagnetism. Signal Integrity analysis is important for electrically long circuits. The signal wavelength is calculated from the signal rise time. In digital systems the rise time is significantly smaller than the symbols "high" or "low"period, hence the equivalent wavelength is smaller than the natural signal wavelength. The wavelength based on the rise time is therefore best suited for the electrical length calculation as it is the smallest measurable wavelength in the signal and more critical with respect to signal integrity. The rise time of the RS-422 signal with the envisaged transmission rate of 115.2 kbps is approximately 70 ns. The corresponding frequency can be found from

$$f_{\mathrm{r}} = \frac{1}{\pi \times t_{\mathrm{r}}} \tag{6.2}$$

where f_r is the frequency in Hertz and t_r is the rise time in seconds [28]. This means that a 70 ns rise time has an equivalent frequency of 4.5 MHz, in a cable this can be approximated by the following [6, 28]:

$$\lambda_r = \frac{0.5 \times c}{f_r} \tag{6.3}$$

Hence,

$$\text{Electrical length} = \frac{l}{\lambda} = \frac{l}{0.5 \times c \times \pi \times t_{\mathrm{r}}} \tag{6.4}$$

Table 6.5 shows the electrical length of some typical cables that are going to be used in the LHC and the transfer lines for the RS-422 serial interface of the FMCM.

Cable length	Electrical length
54 m	2.1
100 m	3.5
141 m	5.0

Table 6.5: Calculated electrical length as a function of cable length, and transmission rate equal to 115.2 kbps.

6.3.1 Test Setup

Test equipment

- Oscilloscope (Textronik)
- Differential probe (Textronik)
- Shielded twisted pair cable for the RS-422 link.
- Test signal generated by the FMCM control interface.

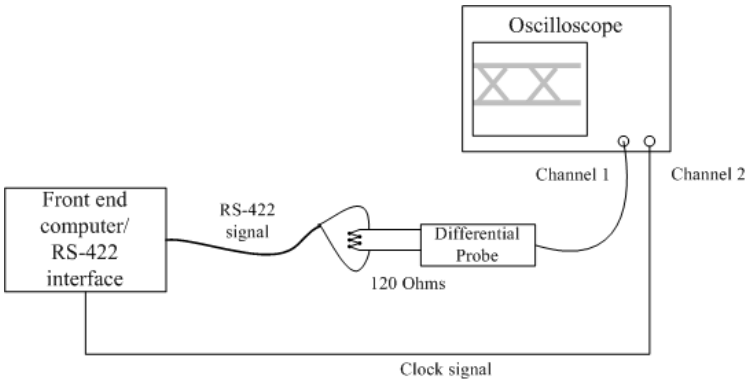

Figure 6.12: Equipment setup for the SI analysis

6.3.2 Test Plan

The SI analysis is a test for the RS-422 interface. The free parameters of the test are the length of the RS-422 cable and the transmission speed. It is assumed that the EMC test in Section 6.2 has been conducted on this stage, and that the shielding of the cable with the best EMC qualities has been chosen.

Transmission rate	Cable length		
	54 m	100 m	141 m
19.2 kbps			
38.4 kbps			
57.6 kbps			
115.2 kbps			

Table 6.6: Matrix of combinations for the SI analysis

6.3.3 Test Requirements

The purpose of the SI analysis is to confirm the current setup of the RS-422 serial interface. That is, to verify the configuration of the cables and the choice of hardware. It is important that the length of the cables does not degrade the signal integrity. If the opening of the eye diagram is wide, the jitter in the eye diagram is negligible compared to the duration of one bit and the signal integrity is said to be good.

6.4 Endurance Run and Reliability Test

The endurance run is the final test to validate the general functionality of the FMCM control interface. The purpose of this test is to verify that the requirements for the FMCM control interface are fulfilled, and that the front end software can deliver its services to the interface in an efficient and reliable way.

The different functionalities of the front end software will be tested according to the requirements in Section 4.1.3. To be validated, a requirement must be implemented and the functionality must be according to the specification. The software will be run on a front end computer in a test laboratory for minimum 48 hours. During the last hour one will try to provoke errors by deliberately disturbing the signal. These errors should then be corrected by the implemented error handling scheme. It is also important to check whether the communication between the FMCM and the front end computer is following the rules set in the requirements.

6.4.1 Status of Required Implementations

As one is currently working on a prototype of the FMCM, not all the functionality is available. Hence, some of the requirements could not be validated with this test. The status of the required implementations common to the FMCM and the front end software are shown in Table 6.7.

All the commands and messages are implemented on both sides, however the *status* message and the *PM* message only contain test data. In the *response header* some bytes are filled with dummy-data, that is for the error bits, the information bits and the time-stamp of the last PM acquisition.

Certain requirements like the transmission rate and the two checksums were validated prior to the endurance run, by connecting the front end computer to a PC. The front end computer communicated with a visual basic program that had the most basic functionalities of the FMCM implemented.

In addition there are certain requirements that only concern the front end software, see Table 6.8.

Required Functionality	FMCM (FPGA_version_7)	Front end software
The *status* command/message	×	×
The *update UTC* command/message	×	×
The *PM* command/message	×	×
The *idle* command/message	×	×
The *reset alarm counter* command/message	×	×
The *simulate dump* command/message	×	×
If the FMCM receives undefined data, it answers with '?'		
Timeout if command is not finished after 0.2 seconds		
Four types of PM data: U_Mag, I_Sim_Diff, I_Diff_DCCT and U_Ext	×	×
Transmission speed 115.2 kbps		×
Command checksum	×	×
Response checksum		×
Time-stamp of last PM acquisition		×
Current time-stamp	×	×

Table 6.7: Functionalities implemented in the prototype of the FMCM and the front end software

Required Functionality	Implemented
The *status* command is sent twice every second	×
The *update UTC* command is sent once every 30 second	×
If the PM time-stamp changes, readout of the PM data will be requested by the front end computer	×
Transmission errors must be handled by the front end software	×
The *PM* command has 1st priority	×
The *update UTC* command has 2nd priority	×
The *simulate dump*, *reset alarm counter* and *idle* message have 3rd priority	×
The*status* message has last priority	×

Table 6.8: Specific requirements for the front end software

Finally, the general requirements for the control interface should be validated:

1. FMCM is the master of communication
2. The RS-422 interface must be able to receive a block of at least 4096 bytes at 115.2 kbps
3. It must be possible for 26 FMCMs to transmit PM data at the same time, and up to four FMCMs for a single front end.

The first general requirement is implemented in both the FMCM and the front end computer. To meet the second requirement it was necessary to add some functionality in the driver controlling the UART. This was taken care of by the Front End Section at CERN. The third requirement has been partly implemented, meaning this functionality is included in the FESA framework. However, as there is only one prototype of the FMCM available, this requirement will not be validated during the endurance run.

6.4.2 Test Setup

The test setup will be as for the EMC test, see Section 6.2.2, and the cable shield of the twisted pair cable will be grounded on both sides during the entire test.

Front End Software

Even though some functionality is missing in the FMCM prototype, most of the requirements will be tested. For example to test the triggering of a *PM* command due to a change in the PM time-stamp, a change in the fourth byte of the current time-stamp found in the *response header* will be used. This byte changes approximately once every fourth minute.

Due to the processing time in the FMCM, the front end computer and the RS-422 interface, the transfer of commands and messages take more time then what the transmission rate implies. By doing simple tests the minimum required time for transmissions of the different messages were measured, and the maximum number of retransmissions were calculated. With a transmission rate of 38.4 kbps the sum of the processing and transmission time of the *PM* message was measured to about 2 seconds, the *status* message 120 ms, the *update UTC*, *simulate dump*, *reset alarm counter* and *idle* 10 ms. It was decided that all the messages should be transfered within 500 ms, the interval between consecutive *status* commands, except for the *PM* message which should be transfered within 20 seconds (the minimum time between events in the transfer lines). By limiting the number of retransmissions to five with a margin of one retransmission, the different messages got the following constraints:

- *PM* message : 5 retransmissions
- *Status* message : 3 retransmissions
- *Update UTC*, *simulate dump*, *reset alarm counter* and *idle* message : 5 retransmissions

To specifically test the error handling, counters are implemented to keep a record of the number of messages sent, the number of messages received, the number of retransmissions and the number of failed transmissions for each type of message. The number of sent messages is the total number of messages sent over the RS-422 interface, including the retransmissions. It is a requirement that every message is received before it reaches the maximum number of retransmission, hence no failed transmissions. It is desired that the total number of retransmissions is kept on a reasonably low level.

6.4.3 Test Plan

The endurance run will last for 48 hours, divided into three test periods. During the first 46 hours and 50 minutes the front end software will run as in normal operation. In the second period, lasting 10 minutes, several sever actions (*idle* commands) will be triggered from the FESA Test Tool, see Appendix C.1. In the last hour the same burst generator as used in the EMC test, see Section 6.2.2, will be used to apply noise onto the shielded twisted pair cable. The repetition frequency will be set to 5 kHz and the voltage to 2 kV.

Chapter 7

Results

This chapter sums up the different results obtained during the project period. The test procedures are described in Chapter 6 and the corresponding results are presented in the same order. Each presentation is followed by a discussion comparing the results with the test requirements.

7.1 Results of EMC Tests

For the EMC tests, the results are classified as described in Table 6.1.

7.1.1 The FMCM Timing Interface

It was important to verify that the timing interface of the FMCM could operate in a level 4 EMC environment. Reliable delivery of PM data depends on the accuracy of the time stamp and the PM trigger received from the timing interface. During

Signal from timing card	Severity level			
	0.5 kV	1 kV	1.5 kV	2 kV
pps	A	A	A	A

Table 7.1: EMC test results of the signal from the timing card to the FMCM

the test, the pps signal made the LED on the front panel of the FMCM blink once per second (1 kHz). Disturbance caused by the burst generator would have made the LED blink more or less frequently if the timing interface was susceptible to this noise. Hence, the timing interface of the FMCM passed the EMC requirements.

7.1.2 The FMCM CIBU Interface

The CIBU interface is important as it is responsible for sending USER_PERMIT signals from the FMCM to the Beam Interlock System. When the FMCM detects an error in the powering of the protected magnet it will send a signal to the CIBU

Signal from FMCM	Severity level			
	0.5 kV	1 kV	1.5 kV	2 kV
USER_PERMIT TRUE	A	A	A	A
USER_PERMIT FALSE	A	A	A	A

Table 7.2: EMC test results of the signal from FMCM to the CIBU

requesting a beam dump. The interface therefore has high requirements for electromagnetic compatibility. Disturbance caused by the burst generator did not change the transmitted value of the USER_PERMIT signal from the FMCM to the CIBU. The received data was monitored by looking at two LEDs on the CIBU blinking green if the signal was true, and red if it was false. This was consistent throughout the EMC test.

7.1.3 The RS-422 Interface

Three groups of tests were conducted for the RS-422 interface. In each group, a different shielding technique for the cable was selected and the EMC level varied from 1 to 4. The main results are shown in Table 7.3.

Shielding of cable	Severity level			
	0.5 kV	1 kV	1.5 kV	2 kV
Grounded on both sides	A	A	A	A
Grounded on one side	B	B	B	B
Not grounded	A	B	B	B

Table 7.3: Results EMC test of RS-422 interface

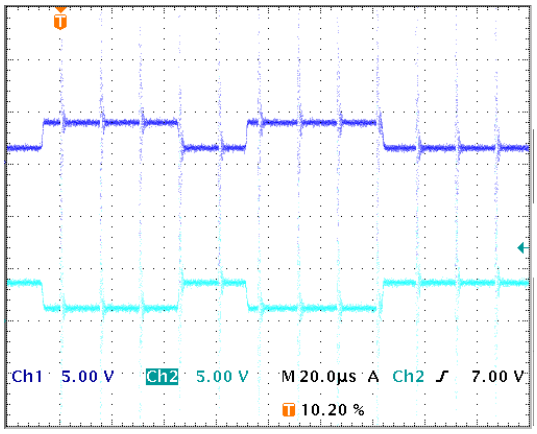

Figure 7.1: Screen shot of the two signals taken from the input of the differential receiver of the FMCM. One can clearly see the transients disturbing the signal.

None of the tests generated more than single byte errors. The communication between the FMCM and the front end computer was stable during the entire test period meaning that all commands were received by the FMCM and it answered with the correct response. Under no circumstances did the burst test cause entire frames to be lost. In fact, it was necessary to increase the repetition frequency of

the transients to detect any errors at all. The burst generator was programmed with a repetition rate of 66 kHz in stead of 5 kHz, see Table 6.2.

The actual disturbance made by the transients could be clearly seen on the oscilloscope. However, as the transients are short of duration compared to the signal pulse, the analog to digital converter managed in most cases to sample the signal in between the transients, see Figure 7.1. With the transmission speed set to 38.4 kbps and a repetition rate of 66 kHz, the number of transients affecting one single data bit is 1.7 in average.

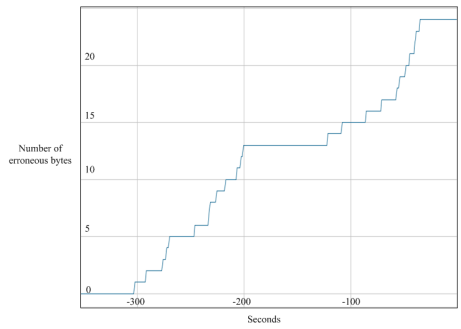

Figure 7.2: Graph showing the number of erroneous bytes during two EMC test periods with transient burst of 0.5 kV and repetition frequency 66 kHz. The shielding of the cable was grounded on the VME side. The first test run provoked 13 byte errors and the second run 11 byte errors.

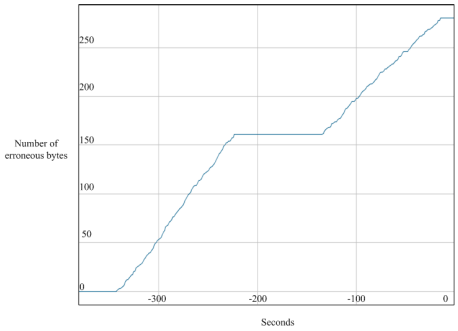

Figure 7.3: Graph showing the number of erroneous bytes during two EMC test periods with transient burst of 1 kV and repetition frequency 66 kHz. The shielding of the cable was grounded on the VME side. The first test run provoked 161 byte errors and the second run 119 byte errors.

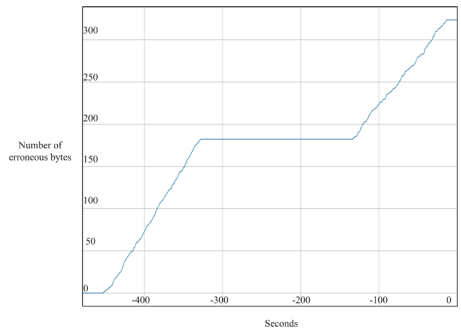

Figure 7.4: Graph showing the number of erroneous bytes during two EMC test periods with transient burst of 1.5 kV and repetition frequency 66 kHz. The shielding of the cable was grounded on the VME side. The first test run provoked 182 byte errors and the second run 141 byte errors.

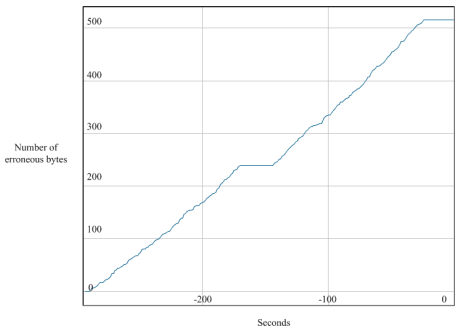

Figure 7.5: Graph showing the number of erroneous bytes during two EMC test periods with transient burst of 2 kV and repetition frequency 66 kHz. The shielding of the cable was grounded on the VME side.The first test run provoked 239 byte errors and the second run 276 byte errors.

The front end software verified the received PM messages and compared each byte with the expected value. 14 bytes of the Response Header are time stamp values that changes continuously and were therefore eliminated from the comparison. This means that during a successful transmission the front end software counted 498 bytes. One test-period lasted 2 minutes, and a PM message was transmitted once every 1.2 seconds. Hence, during one period 51200 bytes were sent from the FMCM to the front end computer, and 49800 bytes were checked. The result of the byte error count was presented in a graph, and Figure 7.2, Figure 7.3, Figure 7.4 and Figure 7.5 show the results for the different EMC levels, with the shielding of the cable being grounded on the VME side.

Table 7.4 gives a resume of the obtained results for all the combinations of shielding. In the worst case the byte error rate was 0.0055 for tests on EMC level 4.

Shielding of cable		Severity level			
		0.5 kV	1 kV	1.5 kV	2 kV
Grounded on both sides	First Run	0	0	0	0
	Second Run	0	0	0	0
Grounded on one side	First Run	13	161	182	239
	Second Run	11	191	141	276
Not grounded	First Run	0	68	159	268
	Second Run	0	65	153	269

Table 7.4: Results of the EMC test for the RS-422 interface. The table indicates the number of erroneous bytes detected by the front end software during one test run where 100 PM messages were transmitted.

7.1.4 Discussion of Results

When the shielding of the cable was grounded on both sides no errors were detected. When the shielding was not grounded at all, the results were better for lower voltages but worse for bursts of 2 kV compared to when the shielding was grounded on one side. A possible explanation for this is that when the shield was grounded on one side, energy might have flown to the ground connected to the receiving circuit on the VME side. This would have caused a bounce in the ground potential and wrong conditions for the receiver. When the voltage increased to 2 kV this effect probably got small compared to the disturbance from the bursts affecting the signal lines themselves.

There are several limitations to this test. To be able to produce a true byte error rate, the tests should have been run continuously for a longer period of time. However, the results give a clear indication to the choice of cable shielding. Also, having bursts in the LHC were the repetition rate of the transient pulses is 66 kHz is very unlikely. This gives an almost continuous train of pulses, and in real life a repetition rate of 5 kHz would cover most worst case scenarios. In addition, a series of burst will most likely only last a couple of seconds. This means in fact that the byte errors will occur less frequently and can therefore be handled by a simple retransmission scheme with the combination of cable shielding grounded on both sides.

7.2 Results of EMC Test with Error Handling

For the second EMC test only the RS-422 interface was considered, and the error handling procedure in Section 5.1.1 was implemented. As before three groups of tests were conducted where in each group a different shielding technique for the cable was selected and the EMC level varied from 1 to 4.

The setup for the two EMC tests were identical except for the driver and the front end software. A new driver gave the possibility to transmit real PM messages with 4032 bytes. It turned out that this change affected the configuration of the EMC test. For the second test it was possible to detect errors when the repetition rate of the transients was 5 kHz, which is according to the standard for these tests. The errors had a tendency of appearing at the end of the message, which is why longer messages gave different results. The main results are listed in Table 7.5.

Shielding of cable	Severity level			
	0.5 kV	1 kV	1.5 kV	2 kV
Grounded on both sides	A	A	A	A
Grounded on one side	A	A	B	C
Not grounded	A	B	B	C

Table 7.5: Results of second EMC test for the RS-422 interface

During a successful transmission the front end software counted 4018 bytes (4032 bytes minus the time-stamps). One test-period lasted 2 minutes, and a PM message was transmitted once every 2.1 seconds. This was about the time it took to process one command, send it, wait for the reply and receive the response. Hence, during one test period 57 PM messages were sent from the FMCM to the front end computer, and 229026 bytes were checked.

The number of byte errors, the number of retransmissions and the number of failed transmissions for each group of tests are given in Table 7.6, Table 7.7 and Table 7.8. The tests were run twice on each EMC level, and each column in the tables correspond to the same run. There are two types of erroneous bytes; corrupted bytes that appear to have been successfully received and corrupted bytes that are not understood by the receiver and therefore not received at all. Erroneous bytes triggered a retransmission of the message and it could be retransmitted 5 times before it was defined as a failed transmission and discarded. No effort was made to try and save parts of the message.

	0.5 kV		1 kV		1.5 kV		2 kV	
Byte Errors in Total	0	0	1	10	35	41	71	59
Bytes Not Received	0	0	1	10	29	35	68	57
Corrupted Bytes	0	0	0	0	6	6	3	2
Retransmissions	0	0	1	10	31	32	50	43
Failed Transmissions	0	0	0	0	0	0	2	3

Table 7.6: Results of the second EMC test. Cable shield not grounded.

	0.5 kV		1 kV		1.5 kV		2 kV	
Byte Errors in Total	0	0	0	0	19	20	82	77
Bytes Not Received	0	0	0	0	19	17	81	77
Corrupted Bytes	0	0	0	0	0	3	1	0
Retransmissions	0	0	0	0	15	20	41	40
Failed Transmissions	0	0	0	0	0	0	5	2

Table 7.7: Results of the second EMC test. Cable shield grounded on one side.

	0.5 kV		1 kV		1.5 kV		2 kV	
Byte Errors in Total	0	0	0	0	0	0	0	0
Bytes Not Received	0	0	0	0	0	0	0	0
Corrupted Bytes	0	0	0	0	0	0	0	0
Retransmissions	0	0	0	0	0	0	0	0
Failed Transmissions	0	0	0	0	0	0	0	0

Table 7.8: Results of the second EMC test. Cable shield grounded on both sides.

7.2.1 Discussion of Results

The second EMC test confirmed the fact that having the shielding of the cable grounded on both sides is very important. No errors were detected during the EMC test with this shielding technique, even with the repetition frequency of the transients being increased to 66 kHz and the voltage raised to 4 kV.

When the shielding was not grounded at all, the results were better for lower voltages but worse for bursts of 2 kV compared to when the shielding was grounded on one side. This was also experienced for the first EMC test and discussed in Section 7.1.4.

The most interesting results concerning the front end software was the number of retransmissions and particularly the number of failed transmissions. Comparing the number of retransmissions with the number of erroneous bytes, one can see that single byte errors in a message was most common, which is why the number of retransmissions was quite high. During the test a new PM message was transmitted every 2.1 seconds which is about the same time it takes to process the message. This means that most of the PM messages were put in a queue while the FMCM was retransmitting the messages containing byte errors. The only case where the transmission failed was when the voltage was raised to 2 kV and the cable shield was not grounded on one side or neither.

During normal operation of the LHC, events that trigger a PM message might occur every 20 seconds in the worst case. This time interval is large enough to ensure reliable transmission of the PM message. The PM message will therefore not be put in a queue due to retransmission of the previous PM data set as seen during the tests. Considering the fact that transmission failures only took place at the highest EMC level with poor shielding of the cable and with a limit of five retransmissions one can conclude that the software will be able to handle the most likely errors. If necessary the number of retransmissions can be further increased.

Erroneous bytes had a tendency of appearing at the end of the messages, indicating flow problems in the buffers of the RS-422 interface. However, the number of errors

increased with higher voltage of the transients. Thus, byte errors can be considered to be a consequence of both.

It is important to note that due to the limitations of the current prototype, the tests were done with a transmission rate of 38.4 kbps instead of 115.2 kbps. With increased speed the pulse width of the signal will be smaller, and the signal will be less immune to the transient bursts. However, as the series of bursts are likely to last only a couple of seconds, transmission errors due to transients will be quite rare and can therefore be corrected with simple retransmissions.

7.3 Results of SI Analysis

Three groups of tests were conducted. In each group, a different cable length was selected and the data link rate varied from 19.2 kbps to 115.2 kbps.

Table 7.9 shows the results of these tests. The pulse duration of the signal is included in the table. If the jitter is very small compared to the duration of one bit, the decision logic can easily decide the correct state of the bit and the signal integrity is intact. In fact, as one can see in Figure 7.6, Figure 7.7, Figure 7.8 and Figure 7.9 the jitter is so small compared to the bit duration that it was impossible to get a screen shot of the entire eye and make an accurate measurement of the jitter at the same time.

Transmission rate	Total jitter vs. cable length			Pulse duration
	54 m	100 m	141 m	
19.2 kbps	16 ns	36 ns	91 ns	52 μs
38.4 kbps	19 ns	39 ns	106 ns	26 μs
57.6 kbps	24 ns	43 ns	124 ns	17.4 μs
115.2 kbps	26 ns	46 ns	155 ns	8.7 μs

Table 7.9: Measurement of Total Jitter t_{TJ} for different cable lengths and transmission speeds

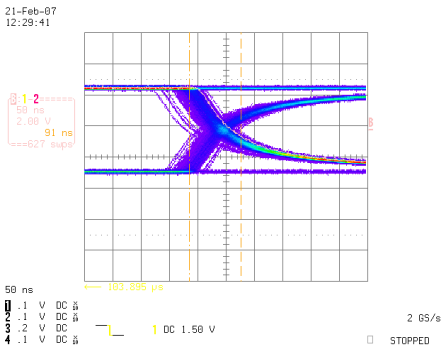

Figure 7.6: Measurement of Total Jitter with eye-diagram, cable length 141 m, transmission rate 19.2 kpbs.

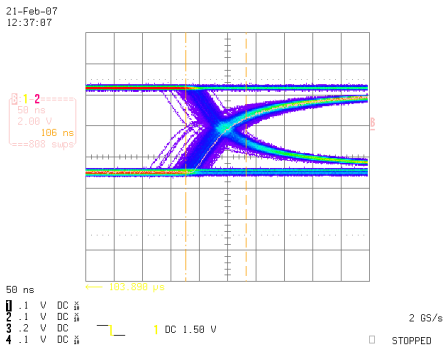

Figure 7.7: Measurement of Total Jitter with eye-diagram, cable length 141 m, transmission rate 38.4 kpbs.

Figure 7.10 shows the eye diagram for the case where the cable length is 141 m and the transmission rate is 115.2.

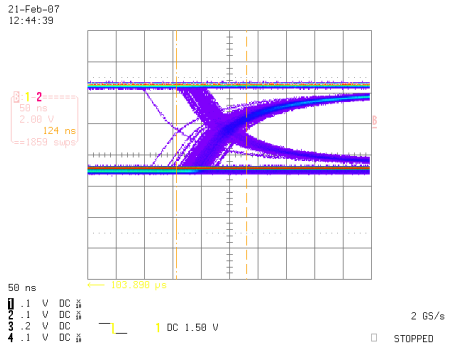

Figure 7.8: Measurement of Total Jitter with eye-diagram, cable length 141 m, transmission rate 57.6 kpbs.

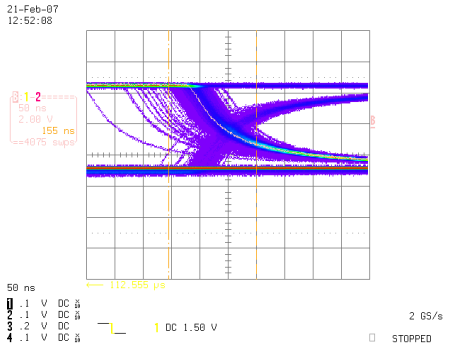

Figure 7.9: Measurement of Total Jitter with eye-diagram, cable length 141 m, transmission rate 115.2 kpbs.

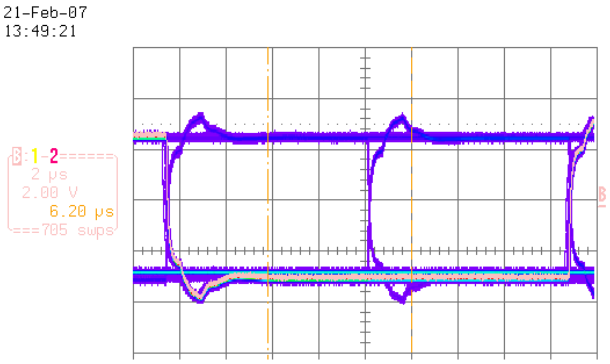

Figure 7.10: Eye-diagram with cable length 141 m and transmission rate 115.2 kbps

7.3.1 Discussion of Results

From the measurements of the jitter and by looking at the eye diagram one can safely conclude that the jitter is negligible. This means that the current setup of the RS-422 serial interface is well chosen for the defined transmission parameters. There is significant overshoot appearing in the eye diagram. However, it does not affect the signal integrity and is most likely coming from the output driver of the RS-422 link. There will be no need to improve the configuration of the cables or to change the hardware as the length of the cables will not significantly degrade the signal integrity. The performed tests confirmed the initial expectations.

7.4 Results of Endurance Run and Reliability Tests

Table 7.10 and Table 7.11 show the validated requirements for the front end software.

Some of the requirements in Table 7.10 could not be validated due to the fact that the functionality was not implemented in the current prototype of the FMCM. The priorities of the different messages were not validated as this would require access to the FIFO queue in the scheduler. The scheduler is automatically generated by

Required Functionality	FMCM (FPGA_version_7)	Front end software	Validated
The *status* command and message	×	×	∨
The *update UTC* command and message	×	×	∨
The *PM* command and message	×	×	∨
The *idle* command and message	×	×	∨
The *reset alarm counter* command and message	×	×	∨
The *simulate dump* command and message	×	×	∨
If the FMCM receives undefined data, it answers with '?'			
Timeout if command is not finished after 0.2 seconds		×	
Four types of PM data: U_Mag, I_Sim_Diff, I_Diff_DCCT and U_Ext	×	×	∨
Transmission speed 115.2 kbps		×	∨
Command checksum	×	×	∨
Response checksum		×	∨
Time-stamp of last PM acquisition		×	∨
Current time-stamp	×	×	∨

Table 7.10: Validated requirements for the front end software

Required Functionality	Implemented	Validated
The *status* command is sent twice every second	×	∨
The *update UTC* command is sent once every 30 second	×	∨
If the PM time-stamp changes, readout of the PM data will be requested by the front end computer	×	∨
Transmission errors must be handled by the front end software	×	∨
The *PM* command has 1st priority	×	
The *update UTC* command has 2nd priority	×	
The *simulate dump*, *reset alarm counter* and *idle* message have 3rd priority	×	
The*status* message has last priority	×	

Table 7.11: Validated requirements for the front end software

the FESA framework and can not be directly influenced by the developer.

7.4.1 Error Handling

During the entire test-period of 48 hours there were no failed transmissions, see Table 7.12. Every message was delivered within the given limit of retransmissions. The retransmission frequency was in general quite high, but less for the messages

Message	PM	Idle	Status	Update UTC
Sent (in total)	721	91	346596	5825
Received	702	68	345494	5757
Retransmitted	19	23	1102	68
% Retransmitted	2.7	33.8	0.32	1.2
Failed	0	0	0	0

Table 7.12: Results of error handling during endurance run

sent most frequently. The highest percentage of retransmissions was encountered with the *idle* message. The *idle* command was triggered through the FESA Test Tool in two series. During the second test-period of 10 minutes, an *idle* message was sent every now and then, and 58 commands were triggered in total. During the EMC test 10 *idle* commands were sent consecutively.

The four graphs in Figure 7.11 shows the retransmissions frequency versus the number of received messages for the *PM*, *idle*, *status* and *update UTC* message. The main characteristic found in this figure is the almost linear relationship between the number of retransmissions and the number of received messages.

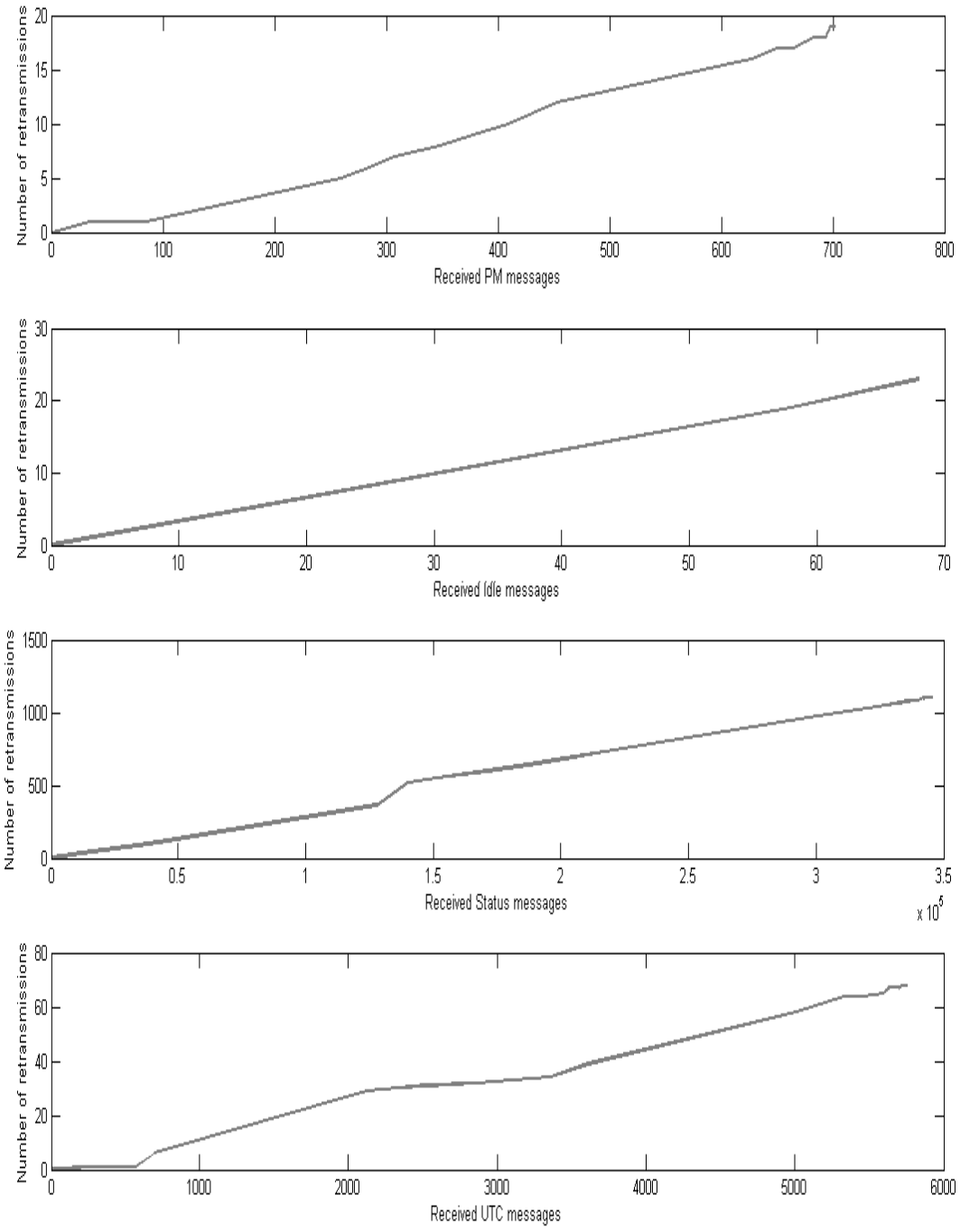

Figure 7.11: Number of retransmissions versus the number of received messages

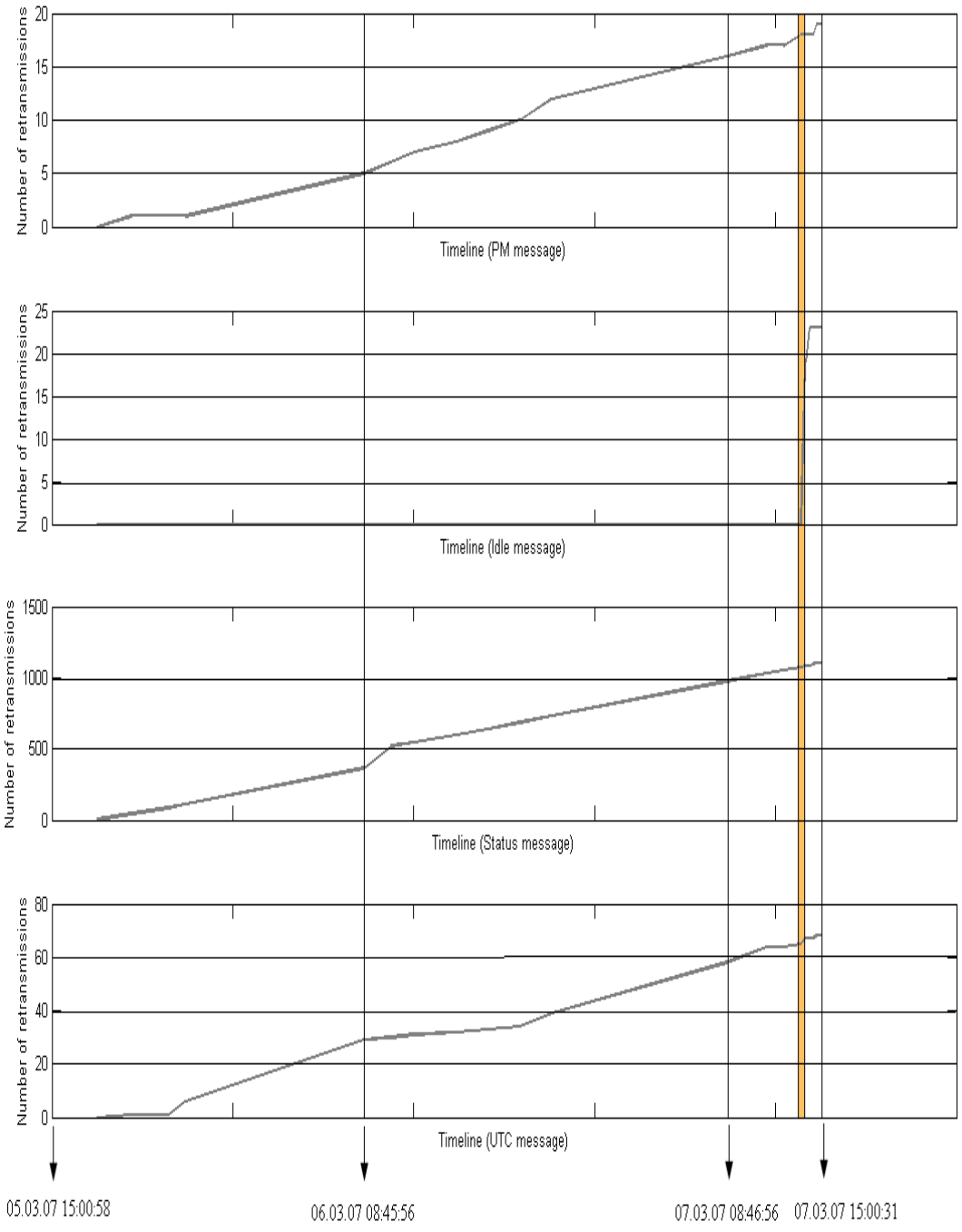

Figure 7.12: Number of retransmissions as a function of time

Figure 7.12 shows the number of retransmissions as a function of time. The shaded area in the figure indicates the second test-period, and the area to the right is the third test-period where a burst generator applied transients to the RS-422 interface.

The generation of retransmissions was monitored during the last two test-periods, and the high retransmission frequency of the *idle* message can be clearly seen in Figure 7.13.

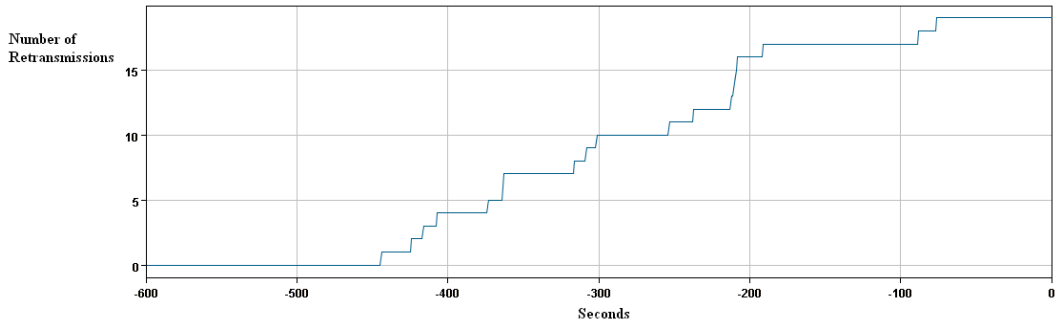

Figure 7.13: Step plot showing the retransmissions of the *idle* messages generated in the second test-period

The number of retransmissions was not affected by the applied burts during the third test-period. However, a certain retransmission pattern was detected while monitoring the *status* message, see Figure 7.14. During a period of 5 minutes the *status* message was retransmitted quite frequently. However, besides these 5 minutes there where hardly any retransmissions at all, indicating that this effect was not due to the transient bursts.

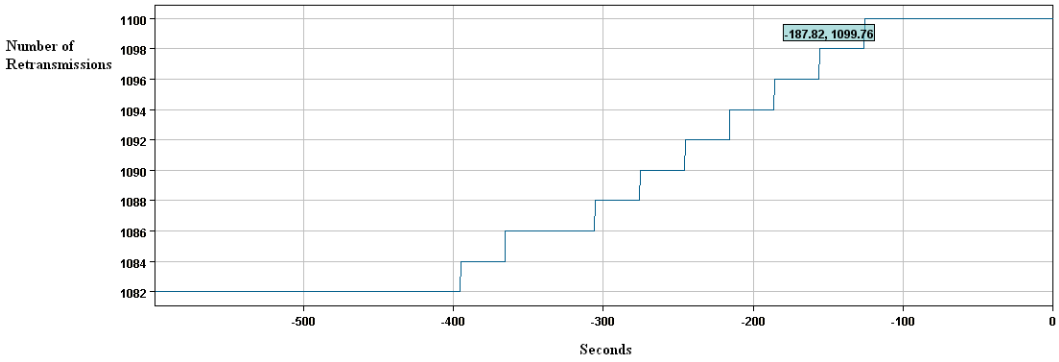

Figure 7.14: Step plot showing the retransmissions of the *status* messages generated during 5 minutes of the third test-period

7.4.2 Discussion of Results

Most of the requirements for the front end software are implemented. The implementations have been validated during the endurance run and other tests outlined in this chapter. Missing functionality in the FPGA of the FMCM prototype and the fact that there is only one prototype available, are the main reasons why some of the initial requirements are not yet fulfilled.

Even though the retransmission frequency was quite high in some cases, the implemented error handling did not fail to deliver a single message. This was the main requirement for the error handling. The high retransmission frequency has two possible explanations. Noise coming from other equipment installed in the same VME crate could have interfered with the signal, or caused conflict in the front end computer. However, a more likely explanation is unsynchronized triggering of commands and conflicts in the scheduler.

The highest number of retransmissions was encountered with the *idle* message, which was triggered via the FESA Test Tool. Delay caused by the server action and thus unsynchronized triggering of the *idle* command with respect to the other commands could have caused conflict. Further, problems with the timing might have led to buffer overflow or underflow in the ring buffer managed by the driver. The front end computer sends a command and waits a certain amount of time before it receives the message returned from the FMCM. If the waiting period is too short, the buffer in the driver does not have enough time to get filled, and the front end computer will only see parts of the message, discard it and ask for a retransmission.

Two results indicate that events occurring regularly might provoke errors. There was an almost linear relationship between the number of retransmissions and the number of messages received. Moreover, during 5 minutes of the third test-period the number of retransmitted *status* messages was very high but not due to the applied noise. By looking at the graphs in Figure 7.11 and Figure 7.12, it can be assumed that short periods with high retransmission frequency occur on a regular basis. These errors might be due to noise from other equipment or more likely, unsynchronized triggering of commands.

Chapter 8

Conclusion

This master's thesis has focused on the software development and analysis of the FMCM control interface. As indicated by extensive particle tracking studies and a beam incident in the beam transfer line TT40 in the fall of 2004, the Fast Magnet current Change Monitors will play an important role within the Machine Protection systems of the LHC and its transfer lines. They will provide protection against fast powering failures in the order of some 100 μs and the hardware design has been carefully chosen to respond to the high demands in terms of reliability and safety of the overall system.

The same stringent requirements apply for the design and development of the related front-end and supervision software in order to guarantee continuous monitoring and control over all devices related to equipment protection. The developments and studies performed within this thesis will allow meeting the initial requirements for the communication between the different FMCMs and the CERN's control system, namely to collect data acquired by each FMCM in case of an event and to provide the required tools to machine operators to remotely supervise and fully exploit the geographically distributed devices.

The communication between the FMCMs and the CERN control system has been implemented in C++, following the guidelines given by the FESA framework. Transient bursts are considered to be the most common type of disturbance in the LHC and the related surface buildings. Hence, error detection was implemented to ensure reliable communication by causing retransmissions until the data has been correctly received. The hardware related to the RS-422 interface has been integrated and configured into the VME framework, using CERN standards for hardware and configuration tools. Analysis were completed on the RS-422 interface with respect to Signal Integrity, Electromagnetic Compatibility and reliability. Through these analysis the communication was validated in terms of the initial requirements based on representative test-setups. Additional tests were done to validate the timing interface of the FMCM prototype.

The EMC tests validated the current implementation of the RS-422 interface. It was demonstrated that most errors are avoided with proper shielding of the cables. The cable shield should be grounded on both sides of the connection. Further analysis of the encountered errors during the EMC test indicated that a simple retransmission scheme would be sufficient.

A second EMC test was done after the error handling procedure was implemented. This test confirmed the results from the first EMC test. With proper shielding of the cables no messages were lost, even with bursts of 4kV and the repetition rate of the transients equal to 66 kHz. The only case where the transmission failed was when the voltage was raised to 2 kV and the cable shield was not grounded on either one side or both. The results from the second EMC test validated the implemented error handling with respect to the main requirements.

A SI analysis was done on the RS-422 interface to reveal interference within the system itself. The jitter measured by means of the signals eye diagram was found to be negligible. This verified the current setup of the RS-422 serial interface for the given transmission parameters. There was no need to improve the configuration of the cables, and the length of the cables will not significantly degrade the signal integrity.

Finally, an endurance run was performed as a final validation of the general requirements of the FMCM control interface. The different functionalities of the front end software were tested and validated according to the initial requirements. The software was running on a front end computer in a test laboratory for approximately 48 hours. Even though the retransmission frequency was quite high in some cases, the implemented error handling did not fail to deliver a single message. This was the main requirement for the error handling. The high retransmission frequency is not yet fully understood, but is assumed to be related to the synchronization of commands rather than disturbance coming from other equipment.

8.1 Future improvements

The high retransmission frequency found during the endurance run, will be studied in further detail. A better understanding of the scheduler embedded in the FESA framework will be necessary to improve the synchronization of commands and to validate the priorities of the different commands.

Tests with four FMCMs connected to one single front end computer, with transmission speed 115.2 kbps should be performed. This will require several FMCMs to be operational. However, the communication from the front end computer to the RS-422 interface can be tested by connecting four industrypack modules to the VIPC626-ET industrypack carrier.

The possibility to retrieve PM data for more than one type of signal will be given. The operator can then select if he or she wants PM data from all the signals or select only one, two or three signals.

Currently, if a message is not received within the given limit of retransmissions, the message will be discarded. A method must be implemented to handle the valid data in the last message for diagnostic purposes.

Appendices

Appendix A

The RS-422 Serial Interface

EIA/TIA-422 is a telecommunications standard for binary serial communications between devices. It is the protocol or specification that must be followed to allow two devices that implement this standard to speak to each other. In this work the standard is referred to as RS-422 (Recommended Standard 422). RS-422 is an updated version of the original serial protocol known as RS-232.

A.1 The RS-422 Standard

With the invention of the microprocessor, logic design has become modular in concept. In most cases the different modules are coupled quite close together on a single printed circuit board, connected to a standard bus. However, peripheral circuits can also be physically separated from the main processor with data communication being handled over cables. When these cables are long, the wavelength of the digital signals might become shorter than the electrical length of the cable and they must be treated as transmission lines, see Section 6.3. Further, these signals are exposed to electromagnetic noise sources which require greater noise immunity then the single board systems. The RS-422 standard is especially developed to meet the requirements of noise immunity when signaling over long cables up to 1.2 km.

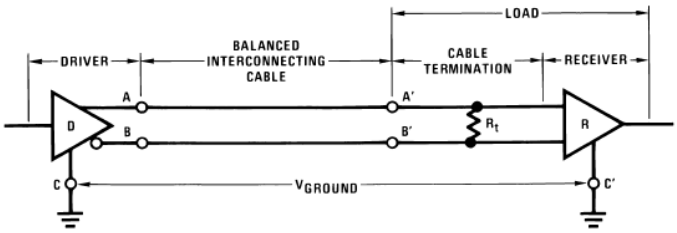

Figure A.1: RS-422 Digital Interface Circuit [29]

Figure A.1 shows the digital interface for the RS-422 circuit. The driver circuit should result in a low impedance balanced voltage source that produces a differential voltage in the range of 2V to 10V. The receiver shall not require a differential input of more than 200 mV to correctly detect the intended binary state. The maximum

differential voltage should be 6V. The reason for using the differential signal is to eliminate the common-mode voltage. The common-mode voltage is the common reference voltage for the two inputs of the driver and receiver interfaces, which consists of the common ground potential and possibly noise. It is defined as the algebraic mean of the two voltages appearing at the receiver input terminals with respect to the receiver circuit ground.

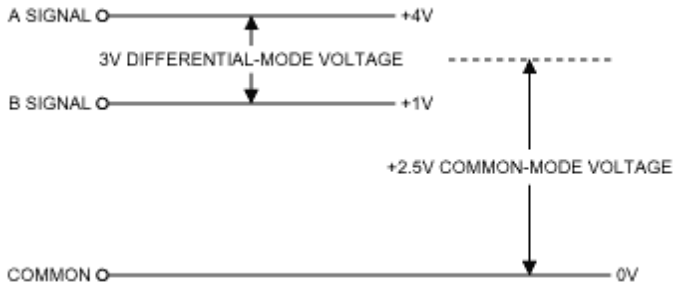

Figure A.2: Common-Mode Voltage [30]

The controlling factors in a voltage digital interface are the cable length, the data signal rate, the characteristic of the interconnection cable and the rise time of the signal. The standard does not mention a maximum cable length, but guidelines are given with respect to distance as a function of data signaling rate. Tolerable signal distortion can be guaranteed for maximum cable length of about 1.2 km with data signaling rate up to 100 kbps. Above 100 kbps the length must be decreased with increased signaling rate to maintain the same signal quality. The maximum cable length is also influenced by the amount of common-mode voltage, ground potential differences between driver and load and cable termination.

The standard recommends that the balanced voltage digital interface will be utilized on data, timing or control circuits where the data signaling rate on these circuits is below 10 Mbps. As long as the interface meets the electrical characteristics of the standard, they can be designed to operate over narrower ranges to satisfy specific applications.

The characteristics of the interconnecting cable should result in a transmission line with a characteristic impedance in the range of 100Ω for frequencies greater than 100 kHz. The cable may be composed of twisted or untwisted pair and is not further specified in the standard.

The signal rise time is a high frequency component which causes interference to be coupled to adjacent channels in the interconnecting cable. A major cause of distortion is the affect the transmission line has on the rise time of the transmitted signal. Attenuation and delay of the pulses increases with the length of the cable. [29, 31, 30]

A.2 Setup of the RS-422 Serial Interface

A schematic overview of the RS-422 interface is shown in Figure A.3.

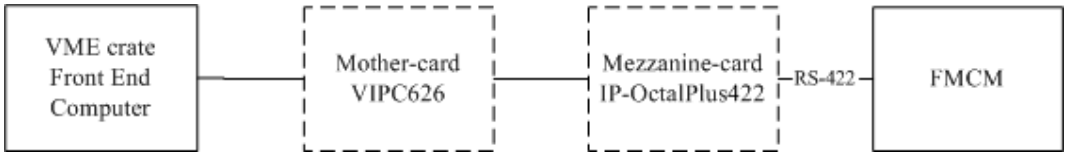

Figure A.3: Communication chain between the FMCM and the front end computer

A.2.1 The VME Crate and the Front End Computer

The Versa Module Eurocard (VME) standard defines an interfacing system for interconnecting microprocessors, data storage and peripheral control devices in a closely coupled hardware configuration. Eurocard is a European standard format for Printed Circuit Boards (PCB). A VME crate has 21 slots for inserting different modules all following the Eurocard standard, and the first slot is dedicated to a crate manager. Every module can be viewed as an address, or block of addresses and is connected through the VME bus which is a Transistor-Transistor Logic (TTL) based backplane. The system allows communication between devices on the VME bus without disturbing the internal activities of other devices interfaced to the bus. The VME bus system consists of 4 sub-buses: the Data Transfer Bus, the Arbitration Bus, the Priority Interrupt Bus and the Utility Bus. The data transfer bus allows masters to direct the transfer of binary data between themselves and the slaves. [32, 33]

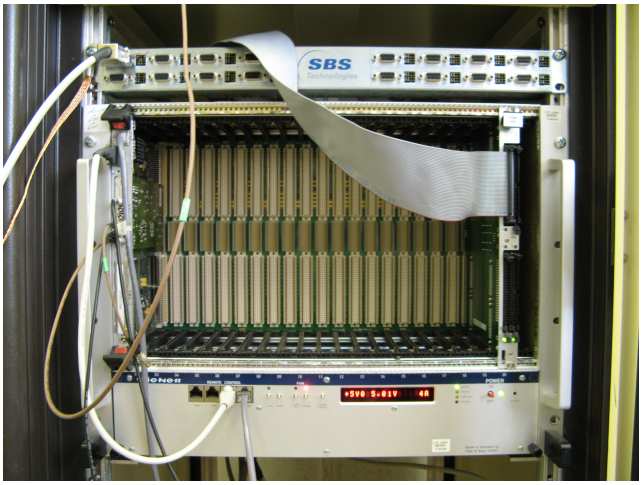

Figure A.4: The VME crate with the front end computer in slot 1 and a VIPC626-ET card with mezzanine IP-IOCTALPLUS422 in slot 20

In the current project a VME64 is used and the module in slot one is the front end computer which also serves as the crate manager. The front end computer is a LynxOs PowerPC 4.0.0, and is the master of communication whereas as the VIPC626-ET card in slot 20 is the slave. The data transfer bus supports 64-bit

data transfers in multiplexed and non multiplexed form. The transfer protocols are asynchronous with varying degrees of handshaking depending on the speed required. The available bandwidth is 80 Mbytes per second.

A.2.2 VMEbus Industrypack Carrier VIPC626-ET

The VIPC626-ET is a VMEbus Industrypack Carrier used to build modular I/O solutions for applications in process control, medical systems, telecommunication and traffic control. Carrier boards have slots for other modules. The VIPC626-ET is a non-intelligent carrier device, or slave, that connects the modules via a bus bridge to the VME bus. It can connect to four single-sized or two doubled size Industrypack (IP) modules [34]. The IP module chosen for the FMCM control interface is the IP-OCTALPLUS422.

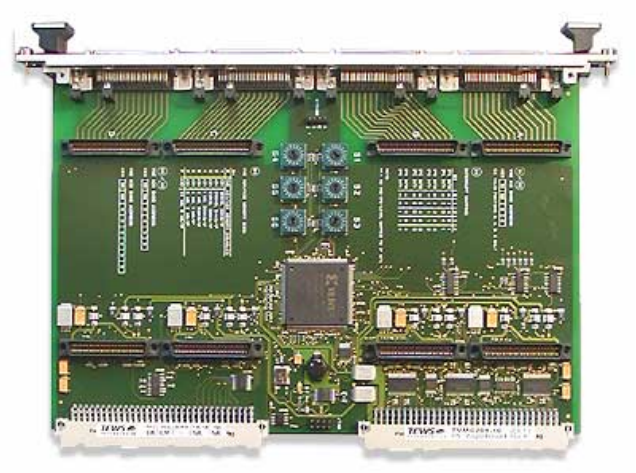

Figure A.5: VMEbus Industrypack Carrier VIPC626-ET

A.2.3 Industrypack Module IP-OCTALPLUS422

Mezzanine modules allow for customization of a VME board to meet specific application requirements. The IP-OCTALPLUS422 interface is an IP compatible module providing eight channels for the serial RS-422 interface. Each channel has a 64 byte FIFO for both transmitting and receiving data. The transmission rate is programmable up to 460.8 kbps. The most important component of the IP module is the Universal Asynchronous Receiver and Transmitter (UART).

The UART translates data between parallel and serial interfaces. It converts bytes of date to and from asynchronous bit streams. Each byte is encapsulated by a start bit and a stop bit which synchronizes the stream with the internal clock of the UART and indicate that the next eight bits must be read and handled as one byte. The bits are binary electrical pulses according to the RS-422 standard. The serial to parallel conversion is handled by shift-registers. The basic components of a UART is the clock-generator, I/O shift registers, transmit/receive control, read/write control logic and a FIFO buffer. The UART can perform parity checking to ensure correct transmission. In this case the transmission of one character requires 11 bits in total.

The UART used by the IP-IOCTALPLUS422 is ST16C654D from EXAR, which actually is a concatenation of four UARTS. Each of them is independently controlled having its own set of device configuration registers. In addition, each UART channel has a 64 bytes FIFO buffer for both transmit and receive, flow control, handling of special characters and a programmable baud rate up to 1.5 Mbit/s [35, 36].

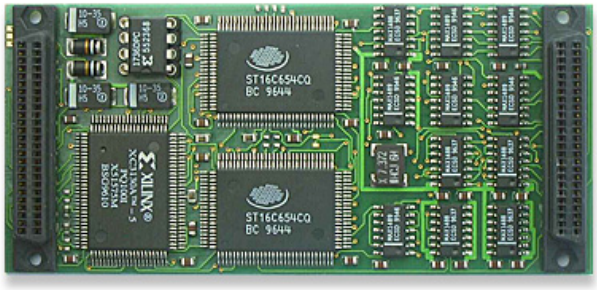

Figure A.6: Industrypack module IP-IOCTALPLUS422

Appendix B

Installations

Due to hardware differences in the LHC accelerator infrastructure, two different installation strategies had to be chosen for the FMCM in the LHC and the transfer lines, see Figure B.1 and Figure B.2. In the LHC, the front end computers are placed in the VME crate situated in the same rack as the corresponding Beam Interlock Controller (BIC) of the Beam Interlock System. In the transfer lines the front end computers are placed in the VME crates of the power converters (MUGEF).

Timing signals required for the synchronization of the internal clock of the FMCM, and the PM trigger are received from a TG8 timing card in the transfer lines and from a CTRP timing card in the LHC. The user interface with the Beam Interlock System, the CIBU is in both cases installed in the same location as the FMCM. The installation of the RS-422 interface is as described in Appendix A.2.

In the case where the BIC is in the same location as the power converter, the voltage across the magnets will be picked up at the power converter terminals. In the other cases, where the where power converters are in the surface buildings and the BIC is in the underground area, the voltage across the magnets will be picked up at the magnet terminals.

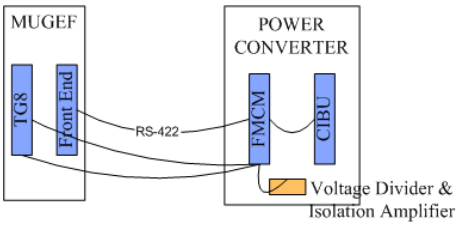

Figure B.1: Installation of the FMCM in the transfer lines

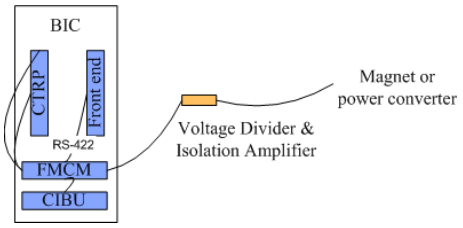

Figure B.2: Installation of the FMCM in the LHC

Figure B.3 and Figure B.4 show the distribution of FMCMs along the LHC and the transfer lines, both underground and in the corresponding surface buildings.

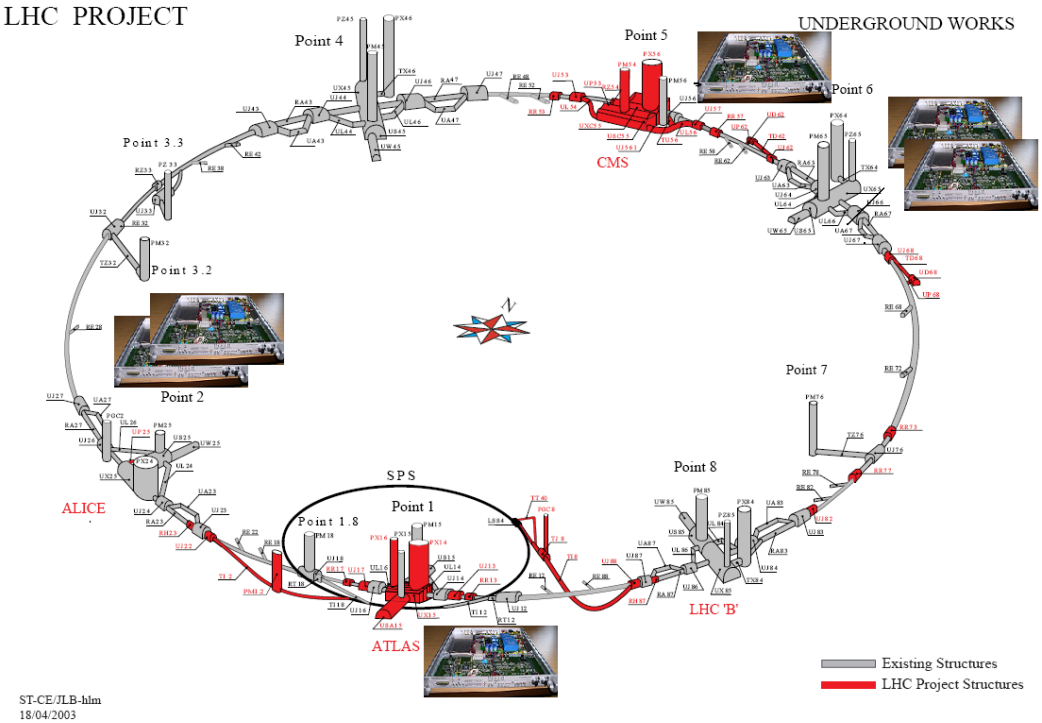

Figure B.3: FMCMs situated underground in the LHC

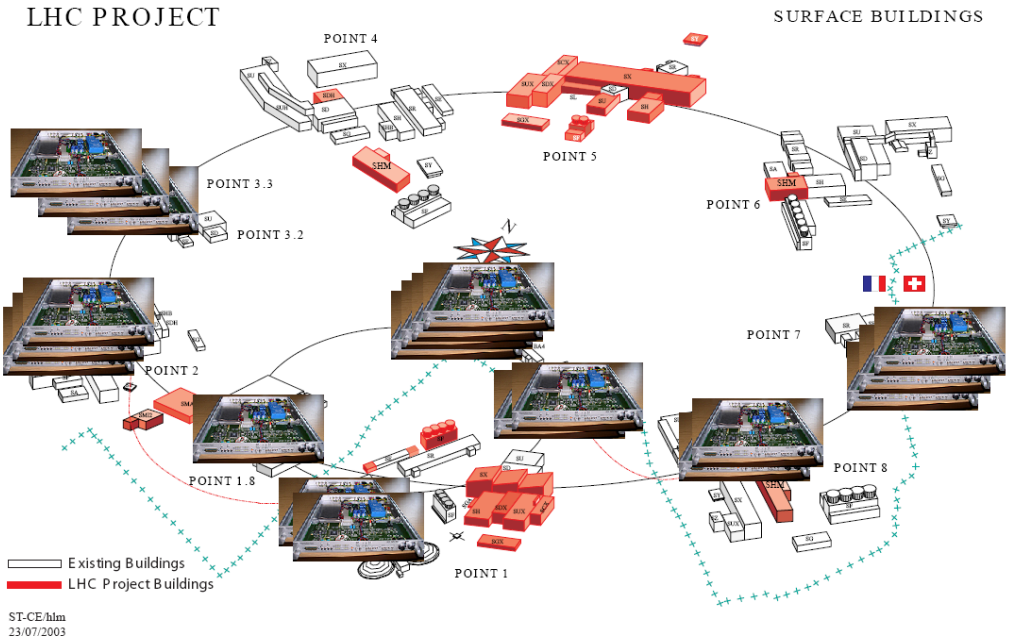

Figure B.4: FMCMc situated in the surface buildings of the LHC and the transfer lines

Appendix C

Front End Software Architecture (FESA)

> *FESA is a comprehensive framework whereby front-end software is to be designed, developed, deployed and maintained according to the standards of the Accelerator and Beams (AB) department at CERN. [37]*

The purpose of the framework is to make consistent rules for how front end software should be implemented for equipment used by the Accelerator and Beams department at CERN. The Front End Section in the AB department is responsible for the development and maintenance of the FESA framework, and they offer different tools which are helpful throughout the development process. These tools are Java applications which are available on the FESA Homepage. The Design Tool is used to prepare a model of the equipment software, where one defines the most important equipment parameters, their dependencies and the program flow. The model is then converted from the original XML format into a C++ project. The developer must then implement the required functionality and establish the connection between the front end computer and the device. With a well defined equipment model the programming should be simplified. The Deployment Tool is used to deploy FESA equipment classes on a front end computer. The Instantiation Tool is used for instantiating devices of a given equipment class on a specific front end computer. The Test Tool is used to access the device while the software is running.

The Equipment model defines some Properties, Server Actions and Real-time Actions, User Events and Timer Events, which together reflect the system specifications and decide the software structure.

A *property* is a collection of *data-fields* that holds information about the device. A data-field can belong to several properties.

A priori, the only part of the software that the developer needs to implement is the executable functions of the different actions. A *real-time action* is an action that is fully managed by the front end computer. A *server action* on the other hand is triggered by an operator request via the FESA server.

A real-time action is always triggered by an event. The most common type of event is the *timer event*, which is nothing else than a timer. One can set the timer event

to trigger a RT action e.g. every 5 ms. The *user event* is defined by software and can be used to trigger a RT action anywhere, anytime with the function call fireUserEvent().

A property is either of type Set or Get and every action is linked to a property. A Get property is usually linked to a RT action and when a RT action is executed it updates the data-fields of this property. In fact a RT action can be linked to several properties. When the software is running the operator can subscribe to the different properties via the FESA Test Tool to see the content of the data-fields. A Set property is usually linked to a server action. When the software is running, the operator can set a property in the Navigator application and this will trigger the server action.

C.1 The FESA Design Process for the FMCM Control Interface

Based on the flowchart in Figure 4.4, an equipment model was designed using the FESA Design Tool, see Figure C.1. The equipment model is shown in Figure C.2, Figure C.3, Figure C.4 and Figure C.5.

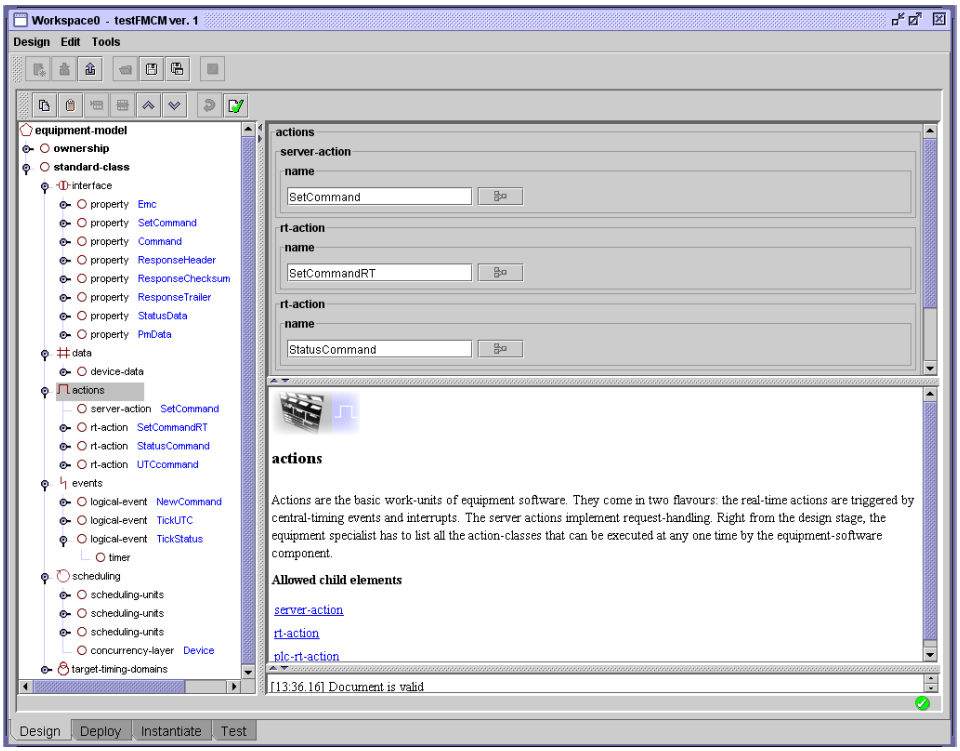

Figure C.1: The Fesa Design Tool

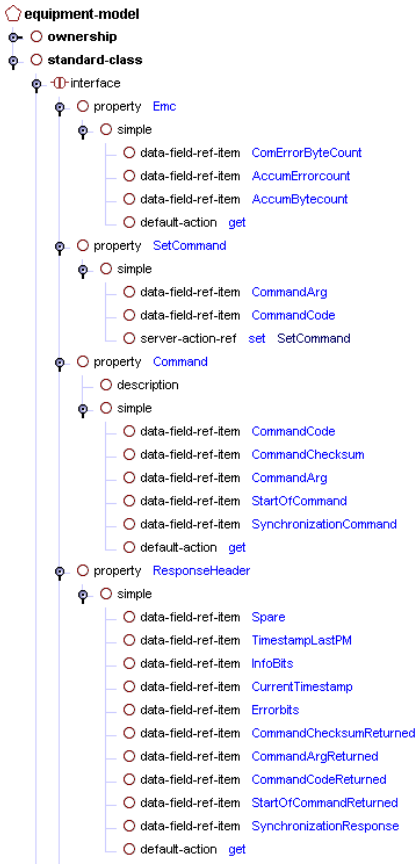

Figure C.2: Equipment model

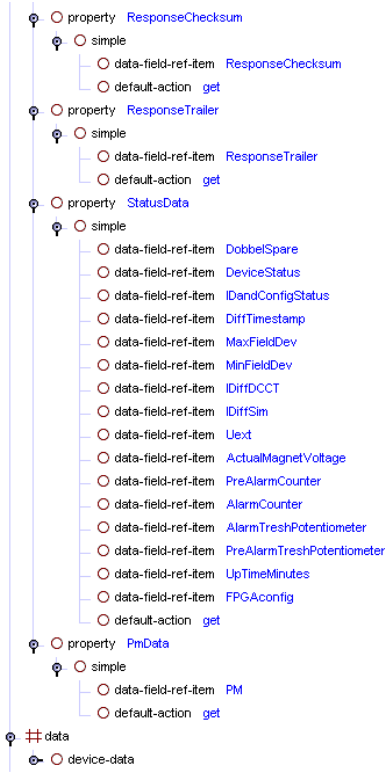

Figure C.3: Equipment model continued (1)

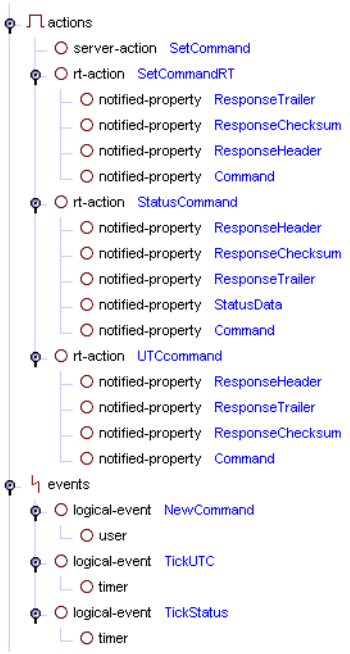

Figure C.4: Equipment model continued (2)

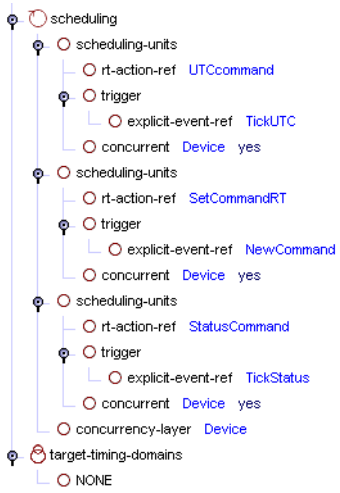

Figure C.5: Equipment model continued (3)

When the equipment model was converted into a C++ project and the implementation was finished, the program was installed in the relevant front end computer with the right configuration. For this the Fesa Deployment Tool was used, see Figure C.6. In this case the software was deployed on a front end computer named ppccodv1 which was used for test-purposes.

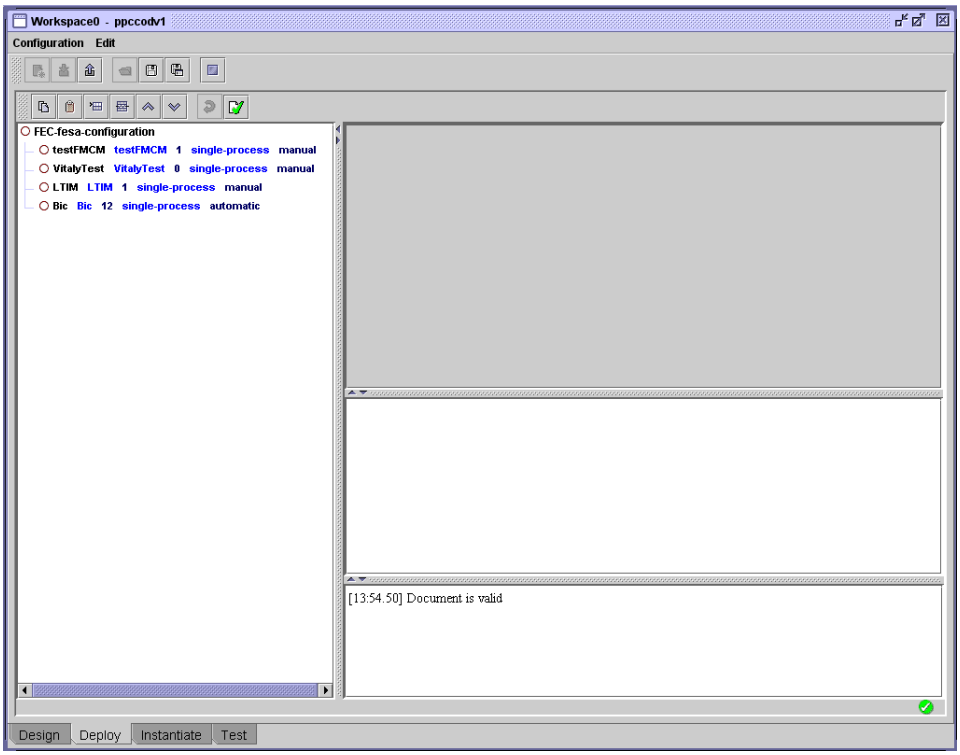

Figure C.6: The Fesa Deployment Tool

To instantiate the different equipment on a single front end the Fesa Instantiation Tool was used, see Figure C.7. Using this tool one can set the timing for the events defined in the equipment model, and define the parameters that are constant during operation of the device like equipment-id, hardware-id and the name of the device. For initial testing only one instance of the FMCM was connected to the front end computer ppccodv1.

When the program is running it is possible to monitor the properties and the belonging data in the Fesa Test Tool. In Figure C.8 one can see the values of the *response header* belonging to a *update UTC* message. The data fields shown at the bottom of the page are hexadecimal numbers, whereas the data fields at the top of the page are decimal numbers and one field corresponds to one byte of information.

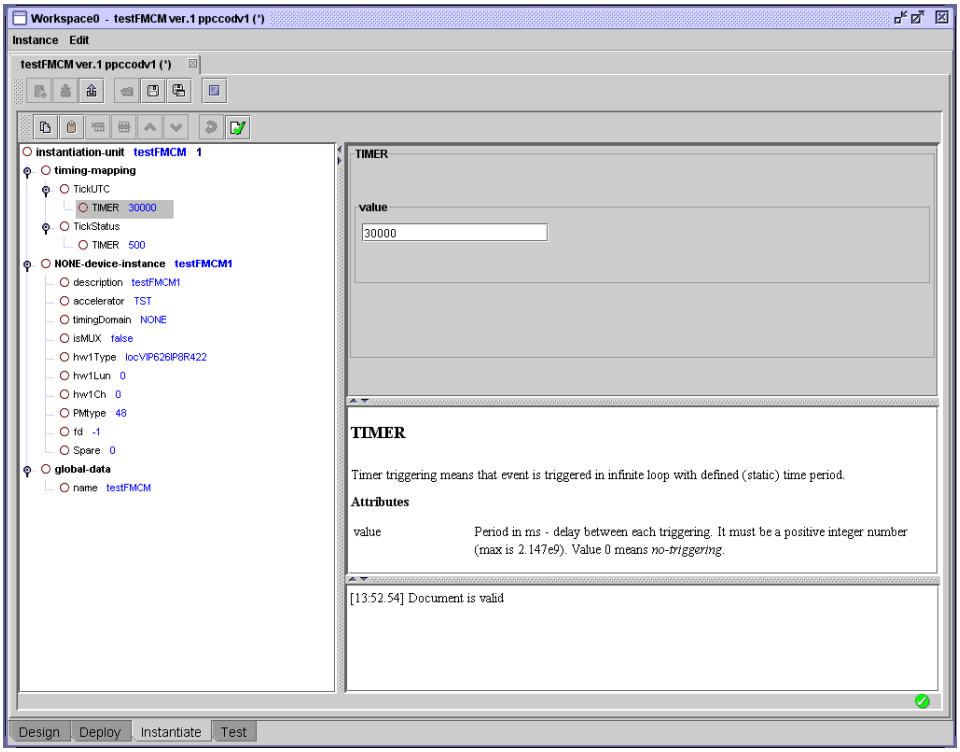

Figure C.7: The Fesa Instantiation Tool

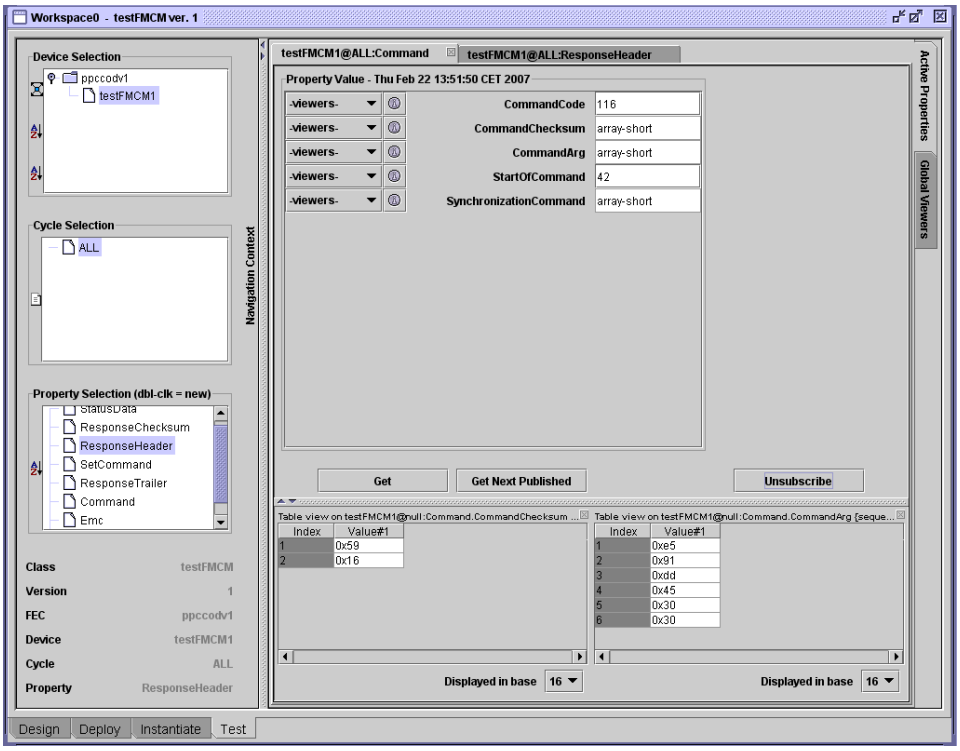

Figure C.8: The Fesa Test Tool

Appendix D

Test of the FMCM Timing Interface

As a new prototype of the FMCM arrived in end of January 2007, it was necessary to perform tests to validate the prototype. One of these tests aimed to verify the behavior of the FMCM timing interface.

The timing interface is shortly described in Section 4.1.2. In Section 4.1.2 a CTRP card is responsible for a PM trigger signal which indicates an external event in the LHC, and the pps signal which synchronizes the internal clock in the FMCM with the timing of the LHC machine. However, in the transfer lines an older version of this card is used, which is called TG8. The timing interface must be tested for both versions. It is also interesting to see the effect of the length of the cable between the TG8/CTRP and the FMCM. In the current version of the FMCM the internal clock is synchronized to the pps only on arrival of the UTC time stamp (every 30 second). This test will verify if this solution is adequate or if the UTC time stamp should be refreshed more often. A solution could also be to increase the rate at which the internal clock is synchronized with the timing of the LHC machine, independently of how often UTC time stamp is updated. In the extreme case one could try to synchronize on every pulse or every other pulse instead of every 30 pulse.

The internal clock in the FMCM oscillates at 12 MHz with an accuracy of 100 parts per million (ppm). The length of one pulse is approximately 83 ns. In the worst case, during one second the oscillator may encounter a drift of 1200 pulses, which is equivalent to 100 μs. After 30 seconds the drift would be about 3 ms.

The specification of the cable, a coaxial cable with 50 Ohm characteristic impedance, states that the rated delay is 5.03 ns per meter. In this test a 103 meter long cable was used, and one could therefore expect a delay of 518 ns in the worst case.

The measurements were done on an oscilloscope with channel one being the pps timing signal taken from the input of the FMCM after the signal had gone through the cable. Channel two was pps signal taken directly from the output of the timing card. Channel four was the same signal taken from within the FMCM after the opto-coupler, which will serve as input for the FPGA, see Figure D.1.

The tests with the CTRP card, see Figure D.2, Figure D.3 and Figure D.4, verified that nor the length of the cable, nor the circuits in the FMCM contribute significantly

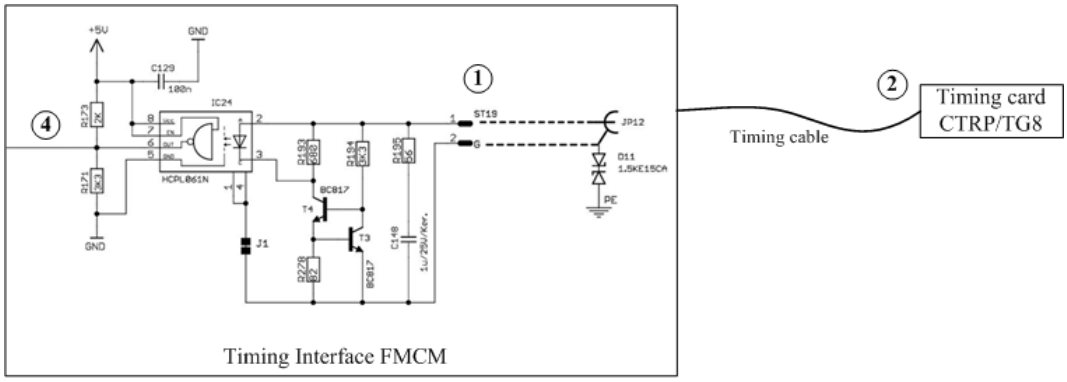

Figure D.1: Schematics of the timing interface of the FMCM. The different signals analyzed on the oscilloscope are shown, with the numbers corresponding to the incoming channels on the oscilloscope.

to the total delay of the timing system. The inaccuracy of the 12 MHz oscillator is still the biggest issue and the total delay is 100.1μs. One could also see that when the cable is not terminated with the 50 Ohm resistor the signals were clearly distorted.

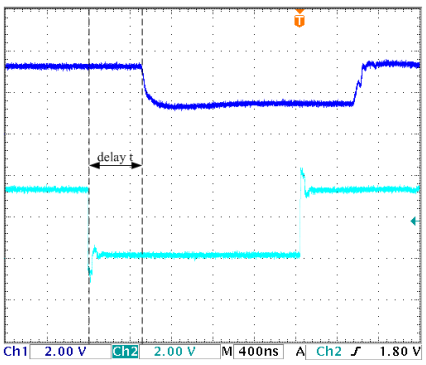

Figure D.2: Test with CTRP card. The cable was terminated with a 50 Ohm resistor. The light blue signal (channel 2) is the pps signal coming directly from the CTRP card. The dark blue signal (channel 1) is the same signal on the other side of the 104 m long cable. The delay due to the cable is approximately 485 ns.

The test with the TG8 card, see Figure D.5, Figure D.6 and Figure D.7, confirms the result of the test with the CTRP card. That was as expected. The delay of the cable and the decision logic in the FMCM was of course the same, but one could clearly see that correct termination of the cable was even more crucial in combination with the TG8 card.

Having a delay of 100 μs of each pulse, means that after 30 seconds or 30 pulses the drift will be about 3 ms. This is too much. As the values of the PM data are recorded during a short period of 40 ms the belonging time stamp will be rather inaccurate. These tests have established the fact that the internal clock of the FMCM causes the most significant delay. A digital Phase Locked Loop (PLL) would improve the

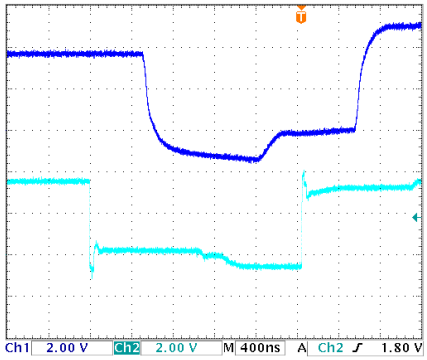

Figure D.3: Test with CTRP card with the cable not being terminated. The light blue signal (channel 2) is the pps signal coming directly from the CTRP card. The dark blue signal (channel 1) is the same signal on the other side of the 104 m long cable. Both signals were distorted.

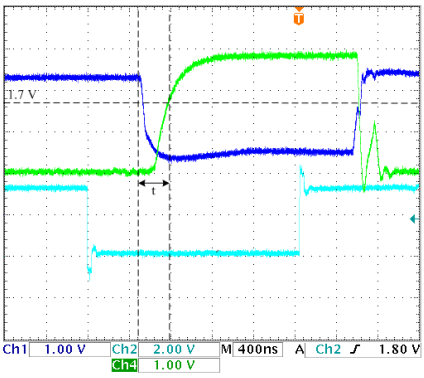

Figure D.4: This is the same image as in Figure D.2 except for the green signal (signal in the middle, channel 4) which is the signal after the opto-coupler. The FMCM triggered the signal on 1.7 V, which gave a delay of approximately 300 ns.

accuracy of the oscillator with a factor of 20. Thus, the maximum drift of the internal clock in the FMCM would be about 5 μs during one second. This means that if a maximum drift of 50 μs was desired, the refresh rate of the UTC time stamp should be 10 seconds. Another possible solution is to synchronize the oscillator with the pps signal from the timing card more often. This is something that will be discussed with the responsible at DESY. It is important to note that this measurements are based on a worst case scenario, and further tests should be done to measure the average drift of the oscillator.

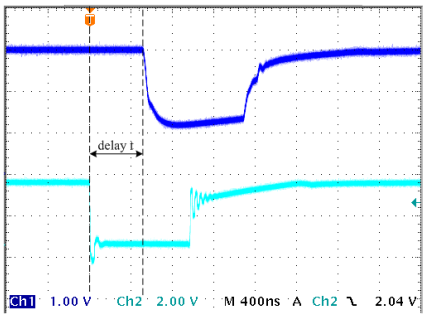

Figure D.5: Test with TG8 card. The cable is terminated with a 50 Ohm resistor. The light blue signal (channel 2) is the pps signal coming directly from the CTRP card. The dark blue signal(channel 1) is the same signal on the other side of the 104 m long cable. The delay due to the cable is approximately 485 ns.

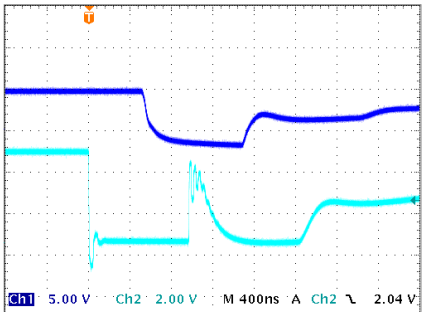

Figure D.6: Test with TG8 card and non-terminated cable. The light blue signal (channel 2) is the pps signal coming directly from the CTRP card. The dark blue signal (channel 1) is the same signal on the other side of the 104 m long cable. Both signals are distorted.

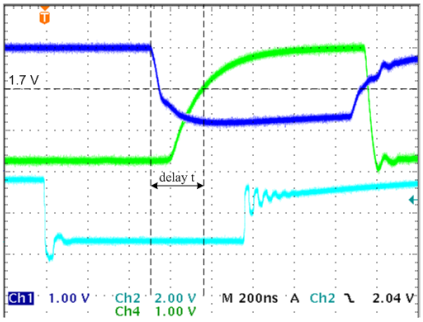

Figure D.7: This is the same image as in Figure D.5 except for the green signal (signal in the middle, channel 4) which is the signal after the opto-coupler. The FMCM triggers the signal on 1.7 V, which gives a delay of approximately 300 ns.

Bibliography

[1] CERN. *CERN Founding Convention.* http://public.web.cern.ch/Public/Content/Chapters/AboutCERN/WhatIsCERN/CERNName/CERNName-en.html, 1954.

[2] CERN. *Cronologie de CERN et de Wolfgang Paulie.* CERN, 2005.

[3] Stephen Myers. *The LEP collider, from Design to Approval and Commission.* Excerpts from the John Adams Memorial Lecture delivered at CERN, 1990.

[4] Markus Zerlauth. *Powering and Machine Protection of the Superconducting LHC Accelerator.* PhD, Graz University of Technology, 2004.

[5] R.Schmidt, R.Assmann, E.Carlier, B. Dehning, R. Denz, B. Goddard, E.B. Holzer, V. Karin, B. Puccio, J. Uythoven, J. Wenninger, and M. Zerlauth. *Protection of the CERN Large Hadron Collider.* The Large Hadron Collider Project, CERN, 2006.

[6] Benjamin Todd. *A Beam Interlock System for CERN High Energy Accelerators.* PhD, Brunel University, West London, 2006.

[7] R. Assmann, M. Brugger, L. Bruno, H Burkhardt, G. Burtin, B. Dehning, C. Fischer, B. Goddard, E. Gschwendter, M. Hayes, J.B Jeanneret, R. Jung, V. Kain, M. Lamont, R. Scmidt, E. Vossenberg, E. Weisse, and J. Wenninger. *Requirements for the LHC Collimation System.* Proceedings of EPAC 2002, Paris, France, 2002.

[8] B.Todd, A.Dinius, P.Nouchi, B.Puccio, and R.Schmidt. *The Architecture, Design and Realisation of the LHC Beam Interlock System.* 10th ICALEPCS Int. Conf. on Accelerator & Large Expt. Physics Control Systems, 2005.

[9] M.Werner, M.Zerlauth, R.Schmidt, V.Kain, and B.Goddard. *A Fast Magnet Current Change Monitor for Machine Protection in HERA and the LHC.* 10th ICALEPCS Int. Conf. on Accelerator & Large Expt. Physics Control Systems, 2005.

[10] Verena Kain. *Power Converter Failure of the Normal Conducting D1 magnet at Experiment Insertions IR1 and IR5, LHC Project Note 322.* CERN, Large Hadron Collider Project, 2003.

[11] M. Zerlauth, B. Goddard, V. Kain, and R. Schmidt. *Detecting Failures in Electrical Circuits leading to very fast Beam Losses in the LHC.* CERN, Large Hadron Collider Project, 2004.

[12] Matthias Werner (DESY), Markus Zerlauth (CERN), Arend Dinius (CERN), and Bruno Puccio. *Requirements for the Fast Magnet Current Change Monitors (FMCM) in the LHC and the SPS-LHC Transfer Lines.* The Large Hadron Collider Project, CERN, 2005.

[13] Andrew S. Tanenbaum. *Computer Networks.* Pearson Educatio, Inc, 2003.

[14] Edward A.Lee and David G.Messerschmitt. *Digital Communication, Second Edition.* Kluwer Academic Publishers, 1994.

[15] G. Kabatiansky, S. Semenov, and E. Krouk. *Error Correcting Coding and Security for Data Networks.* John Wiley & Sons, Inc, 2005.

[16] M.D Bacon and G.M Bull. *Data Transmission.* Macdonald & Co, London, 1973.

[17] K. Kahle. *LHC machine EMC Workshop (LHC 400/230V electrical distribution system).* http://ab-div-po.web.cern.ch/ab-div-po/Infos/Conferences/EMC%20Workshop/2004_EMC_Kahle_rev3.pdf, 2004.

[18] CERN. *LHC machine EMC Workshop.* http://ab-div-po.web.cern.ch/ab-div-po/Infos/Conferences/EMC%20Workshop/Programme.htm, 2004.

[19] F. Bordry. *LHC machine EMC Workshop (LHC power converters).* http://ab-div-po.web.cern.ch/ab-div-po/Infos/Conferences/EMC%20Workshop/Bordry%20EMC%20LHC

[20] M.A. Rodriguez-Ruiz. *LHC machine EMC Workshop (EMC issues for the LHC tunnel cryogenic system).* http://ab-div-po.web.cern.ch/ab-div-po/Infos/Conferences/EMC%20Workshop/Rodriguez%20ACRemc.v2.pdf, 2004.

[21] R. Losito. *LHC machine EMC Workshop (EMC issues for the LHC tunnel cryogenic system).* http://ab-div-po.web.cern.ch/ab-div-po/Infos/Conferences/EMC%20Workshop/Losito%2025_11_2004_RL_LHC_EMC.pdf, 2004.

[22] F. Szoncsó. *EMC in High Energy Physics.* CERN, ECP, 1996.

[23] International Electrotechnical Commission (IEC). *International Standard IEC 61000-4-1.* IEC, Geneva, Switzerland, 1992.

[24] International Electrotechnical Commission (IEC). *International Standard IEC 61000-4-4.* IEC, Geneva, Switzerland, 1995.

[25] Henry W. Ott. *Noise Reduction Techniques in Electronic Systems.* John Wiley & Sons, Inc, 1988.

[26] Eric Bogatin. *Signal Integrity - Simplified.* Prentice Hall Professional Technical Reference, 2004.

[27] Dallas Semiconductor and Maxim. *Application Note 1856, Signal Integrity vs. Transmission Rate and Cable Length for LVDS Serializers.* Maxim Integrated Products, 2003.

[28] R. Schmitt. *Electromagnetics explained*. Elsevier Science, 2002.

[29] J. Abbott and J. Goldie. *Transmission Line Drivers and Receivers for TIA/EIA Standards RS-422 and RS-423*. National Semiconductor Corporation, 2002.

[30] Dallas Semiconductor and Maxim. *Understanding Common-Mode Signals*. Maxim Integrated Products, 2003.

[31] B&B electronics. *RS-422 and RS-485 Application Note*. B&B electronics, 2006.

[32] VMEbus International Trade Association. *American National Standard for VME64*. VMEbus International Trade Association, 1995.

[33] VMEbus International Trade Association. *American National Standard for VME64 Extensions*. VMEbus International Trade Association, 1998.

[34] SBS Technologies. *VIPC626-ET User Manual*. SBS Technologies, Inc, 2004.

[35] SBS Technologies. *IP-OCTALPLUS422 User Manual*. SBS Technologies, Inc, 2003.

[36] EXAR Corporation. *Datasheet ST16C654/654D*. EXAR Corporation, 2005.

[37] CERN. *FESA homepage*. http://ab-co-fe.web.cern.ch/ab-co-fe/, 2006.