Architecture of a Synchronous High-Bandwidth Real-Time Network with Sub-Nanosecond Latency Determinism at the Hardware Level

Zur Erlangung des akademischen Grades eines

DOKTORS DER INGENIEURWISSENSCHAFTEN (Dr.-Ing.)

von der KIT-Fakultät für Elektrotechnik und Informationstechnik des
Karlsruher Instituts für Technologie (KIT)

angenommene

DISSERTATION

von

M.Sc. Vladimir Sidorenko
geboren in Jefremow

Tag der mündlichen Prüfung:	28.07.2026
Hauptreferent:	Prof. Dr.-Ing. Dr. h. c. Jürgen Becker
Korreferent:	Dr.-Ing. habil. Wojciech Zabolotny

Kurzfassung

Mit dem Ziel, immer komplexere physikalische Phänomene zu beobachten und zu untersuchen, entwickeln sich Experimente in der Teilchenphysik hinsichtlich Umfang und Leistungsfähigkeit kontinuierlich weiter, was zwangsläufig zu größeren Mengen an Messdaten führt, die transportiert und gespeichert werden müssen. Herkömmliche Hardwaresysteme reduzieren dieses Datenvolumen durch eine niedriglatente Vorauswahl. Die Effizienz von Hardware-Triggersystemen ist jedoch intrinsisch durch die Komplexität der erforderlichen Auswahlkriterien begrenzt. In modernen Experimenten werden diese Einschränkungen zunehmend zu einem wesentlichen limitierenden Faktor für das Erreichen der physikalischen Ziele. Dies hat zur Entwicklung von Free-Streaming-Auslesekonzepten geführt, die die Fortschritte in der Hochgeschwindigkeitskommunikation und -rechnertechnik nutzen, um die Notwendigkeit einer Vorauswahl durch Hardware zu beseitigen.

Als eines der Experimente, das das Konzept des Free-Streaming-Auslesekonzepts übernommen hat, ist das Compressed Baryonic Matter (CBM)-Experiment stark auf hochpräzise Taktsignale und Zeitsynchronisation zwischen seinen Hardwarekomponenten sowie auf eine reaktionsschnelle und deterministische Datenflusskontrolle angewiesen, um seine wissenschaftlichen Ziele zu erreichen. Diese Anforderungen bilden einen anspruchsvollen Anwendungsfall für ein synchrones, verteiltes Echtzeitsystem (DRTS).

Diese Arbeit stellt eine skalierbare und anpassungsfähige DRTS-Architektur vor. Durch den Einsatz moderner feldprogrammierbarer Gate-Arrays (FPGA) und Multi-Gigabit-Glasfasertechnik sind die Systemkomponenten über bidirektionale Verbindungen verbunden, um Zeit- und Steuerbefehle mit geringer und deterministischer Latenz zu übertragen. Die Punkt-zu-Punkt-Architektur des Systems

ermöglicht Redundanz und Anpassungsfähigkeit für konkrete Anwendungsfälle. Der Zugriff auf die Daten und Rechenressourcen in jedem Knoten des Systems ermöglicht die Implementierung domänenspezifischer Funktionen, z. B. lokalisierte Steuerung oder Domänenstatusaggregation. Darüber hinaus arbeitet das vorgeschlagene Echtzeitnetzwerk synchron mit einer Stabilität der Zeitverschiebungen zwischen den Komponenten im Sub-Nanosekundenbereich, wodurch sich das System für verteilte zeitkritische Anwendungen mit strengen Zeitanforderungen eignet.

Das vorgeschlagene System wurde in einer minimalen Konfiguration implementiert und mit einem automatisierten Charakterisierungsverfahren im Labor getestet. Nach erfolgreicher Designverifikation wurde es im CBM-Experiment als Timing and Fast Control (TFC)-System eingesetzt.

Abstract

Aiming to observe and study more and more complex physics phenomena, particle physics experiments continuously evolve in scale and performance, which inevitably leads to larger volumes of measurement data that must be transported and stored. Traditional hardware systems reduce this data volume through low-latency selection. However, the efficiency of hardware trigger systems is fundamentally limited by the complexity of the required selection criteria. In modern experiments, these constraints increasingly become a significant limiting factor in achieving the physics goals. This has led to development of free-streaming readout concepts that leverage advances in high-speed communication and computing technologies to eliminate the need for hardware pre-selection.

As one of the experiments that adopted the free-streaming readout concept, the Compressed Baryonic Matter (CBM) experiment heavily relies on high-precision clock and time synchronization across its hardware components and highly responsive and deterministic data flow control to achieve its scientific goals. These requirements create a demanding use case for a synchronous distributed real-time system (DRTS).

This work presents a scalable and adaptable DRTS architecture that fulfills the unusually stringent requirements for responsiveness and time determinism. Leveraging the performance of modern field-programmable gate array (FPGA) devices and multi-gigabit optical connections, the system components are interconnected by bi-directional optical links for transporting time and control commands with low and deterministic latency. The point-to-point architecture of the system allows for redundancy and adaptability for a specific application. The access to the data and computing resources in each node of the system allows one to implement

domain-specific functions, e.g. domain-specific control or domain status aggregation. In addition, the proposed real-time network operates in a syntonous way with the stability of the time shifts between components lying in the sub-nanosecond range, which makes the system suitable for distributed time-critical applications with stringent time requirements.

The proposed system has been implemented in a minimal configuration and tested with an automated characterization procedure in the lab. After successful design verification, it has been deployed in the CBM experiment as the Timing and Fast Control (TFC) system.

To all the people who shaped me along the way.

Contents

Kurzfassung i

Abstract iii

Acronyms xi

1 Introduction 1
1.1 DRTS in particle physics 2
1.2 Motivation 4
1.3 Contribution 6
1.4 Outline 7

2 Fundamentals 11
2.1 Distributed real-time systems 12
2.2 Classification of DRTS 13
2.2.1 Hard-, firm- and soft real-time 14
2.2.2 Event-triggered vs time-triggered 16
2.3 Time in distributed systems 18
2.3.1 Long-term clock stability 20
2.3.2 Short-term clock stability 22
2.4 Time synchronization 24
2.4.1 Synchronization condition 25
2.4.2 Synchronization methods 26
2.4.3 Plesiochronous networks 27
2.4.4 Synchronous networks 28
2.5 Triggering in particle physics experiments 32
2.5.1 CBM overview 36
2.5.2 Free-streaming data readout in the CBM experiment 40

3 State of the art . . . 45
3.1 White Rabbit . . . 45
3.2 Time Sensitive Networking . . . 48
3.3 Time and control distribution in physics experiments . . . 50
3.3.1 Mu3e - search for rare muon decays . . . 50
3.3.2 LHCb - precision measurements of matter-antimatter asymmetry . . . 51
3.3.3 ALICE - investigation of quark-gluon plasma . . . 52
3.3.4 Dune - precision studies of neutrino oscillations . . . 53

4 Preliminary research . . . 55
4.1 Requirements and system specification . . . 55
4.1.1 Time stability . . . 57
4.1.2 Message transport latency . . . 59
4.1.3 Scalability . . . 61
4.1.4 Summary . . . 62
4.2 Initial design framework . . . 62
4.2.1 Network topology . . . 63
4.2.2 Communication medium . . . 67
4.2.3 Preliminary concept . . . 70
4.3 Feasibility study . . . 75

5 Approach 1: synchronization through time self-adjustment . 77
5.1 Time synchronization concept . . . 78
5.2 DDMTD . . . 82
5.3 Software PLL . . . 85
5.4 Prototype development . . . 89
5.5 Feasibility analysis . . . 95
5.6 Summary . . . 97

6 Approach 2: time distribution over latency-deterministic links . . . 99
6.1 Time synchronization concept . . . 99
6.2 Transceiver clock domain management . . . 101
6.3 Hardware-level clock distribution . . . 103
6.4 Protocol definition . . . 105

6.5 Reconstruction of the global clock phase . . . 108
6.6 Prototype development . . . 109
6.7 Feasibility analysis . . . 114
6.8 Deployment in the Phase-0 CBM as TFC1 . . . 116
6.9 Summary . . . 119

7 Final architecture: latency-optimized timing and fast control system (TFC2) . . . 121
7.1 Latency-deterministic data transport . . . 121
7.1.1 Analysis of the sources of latency variance . . . 122
7.1.2 Analysis of potential data transport solutions . . . 123
7.1.3 Clock and data alignment . . . 127
7.2 Protocol definition . . . 128
7.3 System architecture . . . 131
7.4 Link calibration . . . 134
7.5 Summary . . . 139

8 System characterization . . . 141
8.1 Characterization goals . . . 141
8.2 Method definition . . . 142
8.3 Test setup . . . 145
8.3.1 Hardware platform . . . 145
8.3.2 Test control software . . . 147
8.4 Baseline measurements with TFC1 . . . 153
8.5 Characterization of the TFC2 data transport . . . 155
8.6 Summary . . . 158

9 Conclusion and Future Work . . . 159
9.1 Conclusion . . . 159
9.2 Future Work . . . 162

List of Figures . . . 165

List of Tables . . . 169

List of Publications . . . 171

Bibliography **175**

Acronyms

AFCK	AMC FMC Carrier with Kintex 7
AGWB	Address Generator for Wishbone
ALICE	A Large Ion Collider Experiment
ASIC	Application-Specific Integrated Circuit
BMC	Board Management Controller
BMON	Beam Monitor
CBM	Compressed Baryonic Matter
CDC	Clock Domain Crossing
CDR	Clock-Data Recovery
CERN	European Organization for Nuclear Research
CMS	Compact Muon Solenoid
CRI	Common Readout Interface
CRU	Common Readout Unit
CTP	Central Trigger Processor
DAC	Digital-to-Analog Converter
DAQ	Data Acquisition
DCA	Device Control Agent

DCSK	Duty Cycle Shift Keying
DCXO	Digitally Controlled Crystal Oscillator
DDMTD	Digital Dual-Mixer Time Difference
DMA	Direct Memory Access
DRTS	Distributed Real-Time System
DTS	DUNE Timing System
DUNE	Deep Underground Neutrino Experiment
ECS	Experiment Control System
FAIR	Facility for Antiproton and Ion Research
FEC	Forward Error Correction
FEE	Front-End Electronics
FIB	Fiber Interface Board
FIFO	First-In, First-Out
FLES	First-Level Event Selector
FMC	FPGA Mezzanine Card
FPGA	Field-Programmable Gate Array
FSM	Finite State Machine
GBT	GigaBit Transceiver
GBT-FPGA	GBT FPGA integration core
GBTx	GigaBit Transceiver chip
GIB	GPS Interface Board
gPTP	generalized Precision Time Protocol

GSI	GSI Helmholtz Centre for Heavy Ion Research
GTX	Gigabit Transceiver
IEEE	Institute of Electrical and Electronics Engineers
IP	Intellectual Property
IPbus	IP-based bus control system
IPMI	Intelligent Platform Management Interface
LHC	Large Hadron Collider
LHCb	Large Hadron Collider beauty
lpGBT	low-power GigaBit Transceiver
lpGBT-FPGA	lpGBT FPGA integration core
LTU	Local Trigger Unit
MAPMT	Multi-Anode Photomultiplier Tube
MAPS	Monolithic Active Pixel Sensors
MGT	Multi-Gigabit Transceiver
MMCM	Mixed-Mode Clock Manager
MRPC	Multi-gap Resistive Plate Chamber
MUCH	Muon Chamber System
MVD	Micro-Vertex Detector
OCXO	Oven-Controlled Crystal Oscillator
OSI	Open Systems Interconnection
PCLK	Parallel Clock
PHY	Physical Layer

PLL	Phase-Locked Loop
PSD	Projectile Spectator Detector
PTP	Precision Time Protocol
QCD	Quantum Chromodynamics
QPLL	Quad Phase-Locked Loop
RICH	Ring Imaging Cherenkov
RMS	Root Mean Square
ROB	Readout Board
RPC	Remote Procedure Call
RTL	Register-Transfer Level
SCLK	Serial Clock
SIS100	Schwerionensynchrotron 100
SIS18	Schwerionensynchrotron 18
SLR	Super Logic Region
SPLL	Software PLL
SSI	Stacked Silicon Interconnect
STS	Silicon Tracking System
SyncE	Synchronous Ethernet
TCXO	Temperature-Compensated Crystal Oscillator
TDMA	Time-Division Multiple Access
TFC	Timing and Fast Control
TOF	Time-of-Flight

TRD	Transition Radiation Detector
TSN	Time-Sensitive Networking
TTC-PON	Timing, Trigger and Control Passive Optical Network
TTS	Trigger and Timing distribution System
UTC	Coordinated Universal Time
VCXO	Voltage-Controlled Crystal Oscillator
VISA	Virtual Instrument Software Architecture

1 Introduction

Distributed real-time systems (DRTS) have become indispensable in many applications that address challenging technical problems through coordinated operation of multiple components. Already today, a broad range of domains in everyday life relies on such systems, and ongoing research continues to extend their reach into emerging areas such as smart power grids and microgrids, as well as the coordinated control of autonomous machines including unmanned aerial vehicles, cars, and robots [1–7]. To perform increasingly complex tasks efficiently and to satisfy the ever-growing requirements of these applications, DRTS face a rising demand for faster and more accurate coordination among their distributed components. This challenge has driven extensive research into deterministic data exchange mechanisms for DRTS [8–12].

Determinism in the context of DRTS is a twofold concept: it may refer to the predictability of the system state after certain input events or the predictability of its temporal behavior, such as response time. The latter aspect is especially critical for applications in which the DRTS interacts with moving objects, where the velocity of the objects directly tightens the requirements on timing predictability and the precision of the system response. To guarantee temporal predictability, the perception of time must be coherent across all components of the distributed system. This makes synchronization of distributed devices a fundamental prerequisite for time-deterministic operation. Consequently, synchronization in device networks has become a central research topic [13–18].

Since synchronization inherently relies on the exchange of timing information between physical devices, its quality is tightly coupled to the predictability of data transport latency in the network [19, 20]. For this reason, requirements for latency

determinism and synchronization accuracy emerge concurrently in modern DRTS, with both becoming increasingly stringent as applications demand higher levels of predictability, responsiveness, and precision.

1.1 DRTS in particle physics

Studying the intricate details of the composition of our universe has required an enormous evolution in experimental technology. Over the past century, the technical instruments of particle physics have advanced from modest table-top setups to overwhelmingly large and sophisticated accelerators that span multiple square kilometers. The largest accelerator today, the Large Hadron Collider (LHC), is built as a ring with a circumference of 27 km [21–23]. To probe matter in forms not naturally found on Earth, immense amounts of energy must be communicated to tiny quantities of common chemical elements, such as carbon or gold, or smaller particles, such as protons. By accelerating these particles to predefined energies and colliding them, new matter states and reactions are produced and subsequently studied with custom and highly complex detectors. These detectors capture the particle fragments generated in collisions and allow researchers to investigate phenomena and theories of fundamental importance to modern physics. Thus, both the accelerator and the detector form crucial components of high-energy experiments, each with its own technical challenges and operational constraints.

Within the accelerator, particles are guided and accelerated inside a beam pipe until they reach certain conditions to be finally extracted to one of the experiments connected to the accelerator. At that point, the particles are directed toward a target, where they collide with another material sample. For accelerators to function correctly, a multitude of auxiliary systems surrounding the beam pipe ensure accurate particle positioning, sufficient cooling, stable power delivery, and efficient acceleration. All these systems must be precisely timed, given that particle beams travel at velocities comparable to the speed of light. Accelerator control therefore represents a prominent use case for DRTS, where stringent requirements for time synchronization and long-distance coordination must be met.

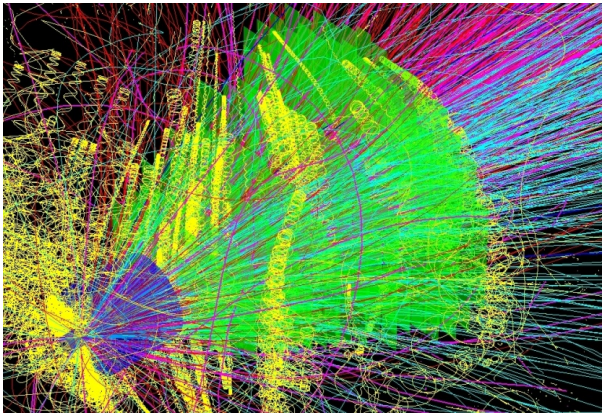


Figure 1.1: A visualization of a simulated central Au+Au collision with its products. (Credit: CBM collaboration)

When the accelerated particles are extracted and collided with a target, a vast number of smaller particles is produced, as illustrated in Figure 1.1. The resulting collision products are meticulously recorded by a heterogeneous detector system individually designed for each experiment. These detectors measure the position, energy, and momentum of millions of particle fragments as they traverse the volume of the experiment. Designing such detectors and collecting data from millions of measurement channels is a remarkable technical challenge. Particularly relevant to this dissertation is the challenge of transporting the great amounts of data from their sources to a storage, where they can later be analyzed for phenomena tied to the scientific objectives. Modern experiments routinely produce terabytes of data, with these numbers continuing to grow [24, 25]. To handle such high throughput near radiation zones, readout electronics typically rely on radiation-hard application-specific integrated circuits (ASICs). Once outside the radiation-heavy area, data aggregation is generally handled by field-programmable gate arrays (FPGAs) [26].

The extent of FPGA involvement in data acquisition (DAQ) is significant, as shown in Table 1.1, which compares the number of FPGA boards deployed in various high-energy physics experiments. While this infrastructure is indispensable, it must also

Experiment	Number of FPGA boards
LHCb	500 [27]
CMS	1200 [28]
ALICE	500 [29]
CBM	200 [30]

Table 1.1: Number of FPGA cards in DAQ systems of different experiments.

be paired with sophisticated data reduction strategies. Only a small fraction of raw detector data is directly relevant to the underlying physics goals [31]. As a result, many experiments rely on hardware triggers or feature-extraction methods that pre-select or compress data before forwarding it further into the DAQ chain [25]. Even with these approaches, DAQ systems still require hundreds of FPGA boards to meet performance demands. The real-time data processing carried out in these boards constitutes a major use case for DRTS.

However, such reduction strategies are no longer universally applicable. The Compressed Baryonic Matter (CBM) experiment at the upcoming Facility for Antiproton and Ion Research (FAIR) in Darmstadt, Germany, exemplifies this shift. In CBM, the data selection criteria are so complex that efficient filtering at the hardware level is infeasible. Instead, the experiment employs a free-streaming data acquisition approach, where all raw data are timestamped directly at the source and forwarded without selection to a high-performance computing farm. There, large-scale computational resources handle the necessary data processing and selection. While offering scientific flexibility, this approach places unique challenges on the CBM DAQ system, which must sustain extremely high throughput with precise time alignment across its distributed components.

1.2 Motivation

The CBM experiment will operate with a beam extracted from the future Schwerionensynchrotron 100 (SIS100) accelerator at the FAIR facility using the

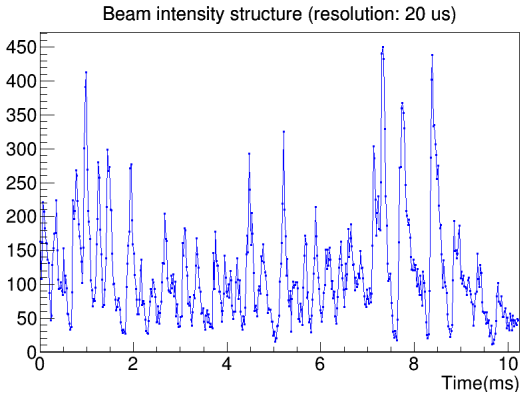


Figure 1.2: Beam intensity fluctuations with slow extraction at SIS18 [34].

slow extraction method [32]. However, experience from the Schwerionensynchrotron 18 (SIS18) accelerator has shown that this extraction technique leads to significant fluctuations in beam intensity on the microsecond timescale [33]. While the CBM DAQ system is designed to continuously sustain the average data throughput over long periods, short bursts reaching up to 400% of the average rate have been observed, as illustrated in Figure 1.2. These bursts, lasting for several microseconds, are sufficient to cause localized buffer overflows within the system.

Such buffer overflows negatively affect the overall performance of the experiment, as they result in uncontrolled and unpredictable partial data losses. Once this occurs, the affected events are corrupted and cannot be meaningfully reconstructed. To maximize the efficiency of data acquisition under these challenging conditions, the throughput within the DAQ system must be actively controlled by a fast DRTS. By dynamically managing local overload situations, the system is able to preserve events that would otherwise be lost due to buffer corruption [34].

Besides, the free-streaming data acquisition with early data timestamping in the self-triggered front-end electronics (FEE) requires accurate distribution of the global time base across the system, as timestamping accuracy directly time measurements in the experiment and the efficiency of event reconstruction in the computing farm [35]. Previous experiments have demonstrated that both

control and timing information can be effectively transmitted together over optical links [36, 37]. With the steady advances in FPGA technology, modern devices now offer sufficient numbers of high-bandwidth, versatile optical transceivers to support such an approach. This makes it feasible to implement a large-scale DRTS that integrates data transport with fast control and timing distribution to ensure a stable and efficient operation of the CBM DAQ system.

1.3 Contribution

Large-scale particle physics experiments have only recently begun adopting the free-streaming approach to data acquisition. Although advanced methods and extensive expertise have been accumulated over decades to reduce and transport the vast amounts of data generated by detectors, new solutions are required to guarantee the stable and reliable flow of continuously streaming data. These solutions must be carefully tailored to the specific constraints of each experiment.

In this context, the central contribution of this dissertation lies in the development of a synchronous, versatile, and highly responsive DRTS capable of meeting the currently defined needs of the CBM experiment, while also remaining sufficiently adaptable and extendable to accommodate future modifications and upgrades. The dissertation presents the development of the Timing and Fast Control (TFC) system, beginning with a broad preliminary study that included prototyping and analysis of multiple potential approaches. Based on these foundations, a distributed TFC architecture was devised that offers competitive performance in both scalability and time distribution. Furthermore, the system design is inherently adaptive, its point-to-point connections between components enable localized, domain-based control, allowing the system to support different experimental configurations and to evolve seamlessly with future upgrades. This flexibility positions the TFC system within the general vision of CBM as a universal measurement apparatus.

Since other subsystems of the experiment are being developed in parallel, the success of the entire project depends on the early integration of preliminary versions

of common functionality. The TFC system is one of these core subsystems, and its development has therefore followed a phased approach. Intermediate functional versions have been integrated into the experiment development setup to support system-level testing and co-evolution with other subsystems.

Beyond the design and implementation of the TFC system itself, significant effort has been dedicated to establishing a rigorous characterization procedure. This methodology ensures that system performance can be systematically verified against the stringent requirements of the experiment. Like the TFC system, the characterization framework has been developed with an emphasis on modularity and adaptability, enabling it to be extended or refined in line with future upgrades to the experiment.

1.4 Outline

This dissertation is organized in 9 chapters that systematically and gradually provide an in-depth overview of the encountered problems and the developed solutions during the development of the highly deterministic distributed control system.

Chapter 2 introduces the reader to the background information that is necessary for the understanding of the fundamental challenges in distributed systems, especially those that are meant to operate with a common notion of time between its components and deterministic data transport delays. The chapter opens with a general introduction to distributed real-time systems in section and their classification based on two different criteria: the strictness of timing constraints and the principle of timely task handling. Afterwards, the chapter delves deeper into the concept of time as it is defined in distributed digital systems and the criteria of its stability. Following deeper into the timing issues in distributed systems, the chapter presents various methods of achieving time synchronization across system components to meet various application requirements. Connecting the earlier presented information with the use case selected for this work, the chapter

is closed with an overview of the CBM experiment and its free-streaming data acquisition scheme, highlighting the need for a synchronized and a deterministic distributed control system for the successful operation of the experiment.

In Chapter 3, modern approaches in applications with similar requirements are reviewed. The chapter begins with an overview of two prominent systems that are used to implement distributed real-time systems: White Rabbit and Time-Sensitive Networking (TSN). Both solutions are widely adopted in various applications, with White Rabbit providing excellent time synchronization in research and finance projects, and TSN serving as the highly standardized and interoperable foundation of more industrial and automotive systems. The chapter continues into an overview of timing and control systems in other experiments that feature free-streaming data acquisition. Timing systems of 4 experiments have been analyzed: Mu3e, Large Hadron Collider beauty (LHCb), A Large Ion Collider Experiment (ALICE) and Deep Underground Neutrino Experiment (DUNE). The experiments have vastly different scales and requirements, yet they all require time synchronization and distribution of latency-bounded messages in their DAQ systems.

Laying the foundation for the contribution of this work, Chapter 4 focuses on the analysis of the DAQ system of the CBM experiment to identify and quantify its requirements. Three different system aspects have been individually considered in the analysis: time stability, message transport latency and scalability. The identified requirements provided sufficient insight to derive the high-level architectural choices, shifting the focus of the chapter towards defining the initial design framework of the TFC system. Thus, the preferred network topology and the communication medium have been determined already at the preliminary research stage and constituted the preliminary concept of the system. Nevertheless, due to insufficient insight into the constraints of two fundamentally different time synchronization approaches present in the prior state-of-the-art analysis, it was not feasible to analytically identify, which of them would be more appropriate to reach the goals of the CBM experiment. Therefore, these constraints must have been identified through additional feasibility studies that involve prototyping of each of the two approaches.

The first of the two aforementioned feasibility studies is covered in Chapter 5. The chapter begins with the introduction of the studied time synchronization principle. The principle involves exchange of time information between system nodes and the nodes adjusting their own time based on this exchange. The study reuses the major enabling feature of this approach, the digital dual-mixer time difference (DDMTD), which is introduced in the beginning of the chapter. This is followed by the description of the software phase-locked loop (PLL) that is built, using the DDMTD as the fully digital phase detector. The software PLL presented in this work features a modified architecture compared to the PLL used in the White Rabbit project, offering additional dynamic flexibility and adaptability during system operation. The rest of the chapter covers the details of the prototype development and the tests carried out to identify the advantages and the constraints of this time synchronization approach.

Chapter 6 covers the feasibility study of the other time synchronization approach. This approach is based on more authoritative distribution of time from a central system node to all other nodes, with the subordinate nodes adopting the received time as correct. Beginning with the general description of the approach, the chapter describes essential aspects of hardware techniques, necessary to ensure the correct operation of this approach. Thus, advanced management of transceiver clock domains is discussed in the chapter, along with the in-depth details of a clock propagation across cascaded optical links. Mapping these advances techniques to a distributed hardware system allows to achieve latency determinism of data transport channels with high precision, impossible to achieve with techniques on higher levels. The chapter continues into the definition of the communication protocol for the prototype and the reconstruction of the global system time by the endpoints. The named reconstruction is critical for correct functionality of the TFC system, as the message transport rate is faster than the global time base of the experiment, which creates ambiguity of the global time phase as perceived by the system nodes. After the overview of the prototype concept, the chapter describes the details of the prototype development and the tests that were carried out to study the feasibility of the approach in the CBM experiment. As the feasibility study resulted in a fully functional prototype of the time distribution system, the chapter

is closed with a report of the deployment and successful usage of the prototype in a preliminary version of the CBM experiment.

Building up on the substantial insight gained in the feasibility studies, Chapter 7 describes the path towards the final system that would meet the requirements of the experiment defined in Chapter 4. Chapter 7 focuses on bringing the data transport latency determinism to the required level and provides an analysis of the intricate sources of latency variance and potential data transport components that can be used. The analysis is followed by a brief overview of the existing data transport interfaces in adjacent applications. Afterwards, the chapter describes mapping of the hardware techniques gained in feasibility studies to the most suitable of the reviewed interfaces and its adaptation for FPGA-to-FPGA communication in the TFC system. The chapter continues into the detailed description of the devised architecture of the system that is capable to deliver the performance according to the specification. Additionally, it describes the communication protocol that ensures correct and stable operation of the system. The chapter is concluded with the custom calibration procedure that is required for clock distribution in a cascaded network with a sufficient precision.

The last chapter before the conclusion of the dissertation, Chapter 8, wraps up the development of the system with an extensive characterization procedure and validation of the parameters of the TFC system developed in this work. Starting with a summary of the characterization goals, the chapter advances to the definition of the characterization method. After the foundation, the focus of the chapter is shifted towards implementation of the method in an automated test setup. At the end of the chapter, the setup is used to first establish a baseline performance with the prototype from Chapter 6, and then to quantify the optimizations introduced by the final setup described in Chapter 7. The results of the measurements are interpreted and presented to the reader, thus closing the chapter before the conclusion of the dissertation.

2 Fundamentals

In this chapter, the fundamental concepts necessary for understanding the major contribution of this work are introduced. First, the chapter provides a general introduction to distributed real-time systems and the key classifications with regard to the strictness of the temporal requirements and the principle of observing the temporal relationships. This is essential for understanding the application requirements and identifying the critical parameters of the system that needs to be developed. Following this introduction, the fundamentals of the concept of time in distributed systems are examined, with a focus on how time is represented in electronic devices and the importance of ensuring the stability of clock generators.

In the following sections, the chapter discusses the criteria, challenges and methods related to network synchronization, highlighting the advantages and disadvantages of different synchronization approaches. Various fundamental types of synchronized networks are introduced here.

The chapter is concluded with an overview of the current use of distributed real-time systems in particle physics experiments, with a focus on the most time-sensitive functions and the technologies that have been developed for such applications. Starting with a discussion of the conventional triggered data acquisition systems, the focus is shifted towards the novel free-streaming readout concept used in the CBM experiment, which sets the context for the main contribution of this work.

2.1 Distributed real-time systems

Implementation of real-time systems in a single devices comes with a significant limitation that the system is bound to a single location. In applications with sensors and actuators naturally separated in space, this results into great complexity of harnessing and signal delivery solutions. Another major limitation of real-time systems is the limited resources of any computational unit. For example, appropriate performance of a real-time system may require complex data processing algorithms that exceed computing capabilities of a single computational unit. To overcome the aforementioned limitations, distributed real-time systems have emerged and have been deeply integrated in various parts of our lives over the past few decades. Distributing the processing tasks in a real-time system across multiple devices drastically increases the available resources, making it possible to implement complex processing algorithms and improve system reliability through redundancy. In addition, the distributed nature makes the system architecture scalable, more flexible in spatial arrangement and opens the possibility of making reconfigurable modular architectures.

Nevertheless, all these advantages of distributed real-time systems come at the cost of added complexity. The computational cluster of any distributed real-time system has a complex structure and contains multiple physical devices. These devices must be interconnected with a real-time network to operate together as one system. As part of a real-time system, such a network extends the general delay model presented above and must address the same temporal requirements, guaranteeing timely data transport between components of the system [38]. Due to communication latency, connections between components can introduce significantly larger delays compared to the internal delays of the components. This also applies to latency jitter, which may be a critical parameter in some applications [10, 39].

Another challenge added by the distributed nature of real-time systems is the inherently independent notion of time in each component [40]. Any computer operates at its own frequency and has no natural awareness of the physical "wall" time. A computer can only derive temporal information from the clock signals

available to it, assuming that the frequencies of these clock signals are well known. Although any frequency generator has a specified frequency value, the real output frequency differs from the nominal value within the tolerance margin typically specified by the vendor.

This slight difference poses no problem if a real-time task is performed by a single node: the vendor-specified time rate, while not perfectly accurate, is sufficiently precise to guarantee the timeliness of task execution between sampling points. However, in a distributed system, the local clocks of the computers will have different, albeit very slight, frequency offsets. Over time, these offsets inevitably accumulate, shifting the observed time of the system components [41]. Such unknown and inconsistent shifts in local time make it impossible for the system components to operate cohesively and ensure the timely collective execution of real-time tasks. Therefore, distributed real-time systems require the establishment of a common, or global, time to function effectively.

2.2 Classification of DRTS

Because DRTS are used in many different fields, these systems are designed with specific requirements and properties based on the functions they need to support. Since there is so much variety, it is important to have a classification system that clearly defines the real-time requirements of each system. This helps in designing real-time systems that meet the required performance and reliability.

Over time, several classification schemes have been developed, each focusing on different aspects such as system dependability, the strictness of real-time constraints, and resource usage. These classification schemes are useful for understanding the different levels of importance and constraints that real-time systems must follow, ensuring that the systems are properly designed for their specific applications.

In this section, two particular classification schemes are discussed. These schemes have been chosen because they are most relevant to the focus of this dissertation and are important for the design and analysis of Distributed Real-Time Systems.

2.2.1 Hard-, firm- and soft real-time

To address the varying levels of timing constraints and the consequences of missing deadlines in different applications, real-time systems are categorized into soft, firm, and hard real-time systems. Each of these three classes has distinct characteristics and applications, which are crucial for understanding their design and implementation.

Soft real-time systems are designed to operate within specific timing constraints, with flexibility to tolerate occasional missed deadlines without leading to failures. These systems aim to respond to inputs within a predefined timeframe, focusing on maintaining an acceptable level of performance and service quality rather than absolute precision. When deadlines are missed, the performance of the system may degrade, but it continues to function, ensuring that the overall operation remains intact.

A common example of a soft real-time system is a traffic light management system. In such systems, timing is critical to ensuring the smooth flow of vehicles and minimizing congestion. However, if a traffic light signal is delayed by a few seconds, it may cause temporary inconvenience or slight delays for drivers, but it will not lead to a significant disruption in the broader traffic network. The system is designed to handle these minor timing deviations gracefully, adjusting operations to maintain functionality and service continuity even under varying conditions.

Firm real-time systems are characterized by the need to meet deadlines but can tolerate occasional misses without causing a complete system failure. These systems require tasks to be completed within a specific timeframe to be effective, but if a deadline is missed, the result may be discarded, and the system continues to

operate normally. The characteristics of firm real-time systems include moderate timing constraints, where deadlines are important, but occasional misses do not lead to catastrophic failures. Missing a deadline results in degraded performance rather than system failure.

These systems are common in applications where timely data processing is crucial but not life-threatening, such as multimedia streaming and online transaction processing. A car parking camera is a good example of a firm real-time system. When the driver is reversing the car, the camera must provide a real-time view to assist in accurate parking. If the video feed lags or briefly freezes, the image may no longer be useful, and the driver might miss the critical moment needed for precise maneuvering. However, the camera continues to function and will still provide a video feed, even if that specific moment's information is lost. This ability to handle occasional timing issues without the system failing entirely is what defines firm real-time systems.

Hard real-time systems are those in which meeting deadlines is absolutely critical. In these systems, the timing requirements are strict, and any missed deadline is considered a system failure, often with serious consequences. The system must ensure that all tasks are completed within their specified timeframes, as any delay can lead to unacceptable outcomes. The characteristics of hard real-time systems include deterministic behavior, where the system's behavior is predictable, and tasks are guaranteed to complete within their deadlines. They are used in safety-critical applications where failure can lead to severe consequences, such as in medical devices, aerospace control systems, and industrial automation.

These systems often incorporate redundancy and fault-tolerance mechanisms to ensure reliability. An example of a hard real-time system is an airbag deployment system in a car. When a collision occurs, the airbag must deploy within milliseconds to protect the passengers. If the system fails to deploy the airbag on time, it could result in severe injury or even death. In this context, there is no tolerance for missed deadlines, and the system is designed to operate with the utmost precision and reliability. This need for absolute adherence to timing requirements is what defines hard real-time systems.

Understanding these categories is essential for designing systems that meet the specific timing requirements of various applications. By appropriately specifying and designing real-time systems, engineers can ensure that they meet the necessary timing constraints and provide the desired level of performance and reliability.

2.2.2 Event-triggered vs time-triggered

Real-time systems can also be classified based on how they handle tasks: some systems are built to react as quickly as possible to events that happen unexpectedly, while others are designed to maintain consistent timing across all operations. These two fundamentally different approaches define two classes of real-time systems: event-triggered and time-triggered [38].

Event-triggered real-time systems are designed to react to specific events or changes in the system environment. These systems perform actions when certain conditions are met or when specific inputs are detected. The timing of operations in an event-triggered system is not predetermined but depends on the occurrence of these events. The system remains idle until an event triggers a response, making it highly responsive to changes as they happen.

The advantages of event-triggered systems include their high responsiveness, as they react quickly to changes as soon as they occur, and their efficient use of resources, since the system only acts when necessary [42]. This also makes event-triggered systems flexible, as they can easily adapt to varying workloads, triggering operations only when required. However, this approach also comes with challenges. The timing in event-triggered systems can be unpredictable, making it difficult to guarantee that all deadlines will be met, especially under high load. Managing multiple events can also be complex, as the system might struggle to prioritize and handle events effectively when many occur close together, potentially leading to delays or missed deadlines. Additionally, the non-deterministic nature of event-triggered systems can make it harder to predict worst-case execution times, which is critical for ensuring real-time performance.

In contrast, time-triggered real-time systems operate based on a predefined schedule, where tasks are executed at specific time intervals or according to a strict timetable. In these systems, the timing of operations is determined in advance, and tasks are initiated at precise moments, regardless of the state of the system or external events. The focus here is on maintaining regular, predictable timing for all operations [43].

In a time-triggered system, all activities are initiated by the progression of real time. Each node in a distributed time-triggered system typically has only one interrupt: the periodic real-time clock interrupt. Every communication or processing activity is initiated at a periodically occurring, predetermined tick of a clock. To function correctly in a distributed environment, it is assumed that the clocks of all nodes are synchronized to form a global time that is available at every node. Each observation of the controlled object is time-stamped with this global time, and the granularity of the global time must be chosen such that the time order of any two observations made anywhere in the system can be accurately established from their time-stamps.

The main advantages of time-triggered systems include high predictability, as tasks are scheduled in advance, making it easier to guarantee that deadlines will be met. The fixed timing also simplifies analysis and verification of the system behavior, which is crucial for safety-critical applications. Time-triggered systems ensure that tasks are performed regularly and consistently, which is important in applications where timing is critical. However, time-triggered systems face their own set of challenges. They might not react as quickly to unexpected events since they operate on a fixed schedule. This can be a limitation in environments where sudden changes require immediate attention. Additionally, time-triggered systems can be less efficient in terms of resource usage, as time may be allocated to tasks even when those tasks are not needed. The rigidity of this approach also makes it less flexible when it comes to handling varying workloads or unexpected changes in the environment.

2.3 Time in distributed systems

As previously mentioned, time synchronization is essential for the proper functioning of many distributed applications and serves as a fundamental requirement for time-triggered architectures. Each device relies on a clock oscillator to represent time, which establishes the basic time reference, known as the system time. This hardware clock is the only tool a device has to measure the flow of time.

Time synchronization plays a critical role in addressing these discrepancies by aligning the local clocks of devices within a distributed system. Through synchronization, a common time base is established, allowing all devices to perceive and follow the same time. This shared reference is essential for distributed systems to function correctly, as it enables coordinated actions and precise time-triggered operations. Without synchronization, devices could drift out of sync, leading to issues like data loss, miscommunication between system components, and even complete system failure in critical applications, such as in aerospace, telecommunications, or industrial automation.

For any time-based tasks, such as sorting or scheduling, a computer needs a clear reference for time. Unlike humans, who perceive time as a continuous and infinitely divisible flow, digital computers operate in discrete steps. The smallest measurable unit of time for a computer is determined by the period of its system clock generator. As a result, computers do not experience time as a smooth flow but rather in incremental steps, which are defined by these clock cycles.

To measure time, computers count the cycles of their system clock. Each cycle, known as a tick, represents a fixed interval of time that the computer uses to track the duration of operations. The cumulative count of these ticks forms the local time of the computer, but these ticks are abstract units and do not directly correspond to physical time as humans perceive it. As demonstrated in Equation 2.1, to convert the intervals Δt_{loc} measured in ticks into physical time Δt_{real}, such as seconds, the nominal frequency of the system clock f_{clk}, or local clock rate, must be known. This clock rate defines how many ticks occur per second, providing the necessary relationship between the computer internal time and real-world time.

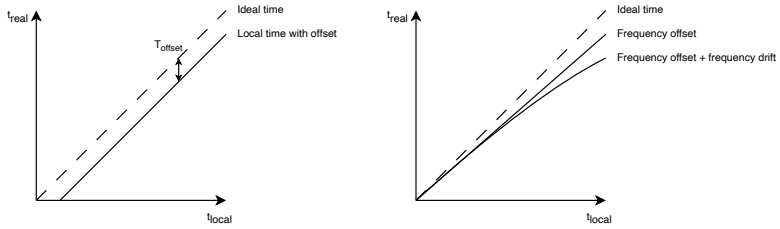


Figure 2.1: Demonstration of the imperfections of the local system clock: time offset (left) and frequency offset and drift (right).

$$\Delta t_{real}(\Delta t_{loc}) = \frac{\Delta t_{loc}}{f_{clk}} \tag{2.1}$$

Knowing the clock rate allows a computer to measure time intervals, which is sufficient for many applications that require only relative timing. However, for cases where the absolute values of local time must be directly and unambiguously related to physical time, this information alone is not enough. In such scenarios, it becomes necessary to establish a clear connection between the local system time and the real-world physical time.

To make this connection, it is essential to know the point in physical time when the computer began counting ticks. This point, referred to as the offset T_{off}, represents the difference between the local time scale and the physical time scale. By determining this offset, the local time can be synchronized with physical time. Assuming the absolute time value t as a special case of a time interval Δt between the current point in time and the beginning of the time scale, T_{off} can be integrated into Equation 2.1 as follows:

$$t_{real}(t_{loc}) = \frac{t_{loc}}{f_{clk}} + T_{off} \tag{2.2}$$

The presented above basic model explains the fundamental relationship between the time flow perceived by a computer system and physical time, as shown in Figure 2.1 (left). Although this model is generally correct, it assumes ideal

conditions for the clock generator that is used as the reference for the local time. However, in systems that are intended to run for longer periods, such as several hours or days, more subtle deviations in the clock frequency become significant and must be considered.

2.3.1 Long-term clock stability

These subtle deviations arise due to inherent inaccuracies in the manufacturing process of the clock generator, which can lead to small errors in the frequency of the oscillator. In addition, aging effects gradually impact the clock stability over time. In systems requiring precise timekeeping, these small frequency variations accumulate over time, causing the local time to drift from the physical time as shown in Figure 2.1 (right).

The frequency error resulting from the manufacturing process of the clock generator remains constant over time and leads to a continuous and predictable drift of the local time. This type of error, known as frequency offset, is generally very small and is measured in parts per million (ppm). The frequency offset can be calculated using the formula $F_{off} = \frac{f_{act} - f_{nom}}{f_{nom}} \cdot 10^6$, where f_{act} is the actual frequency of the clock generator and f_{nom} is the nominal frequency specified by the manufacturer.

With this frequency offset, the actual frequency value can be expressed as $f_{act} = (1 + \frac{F_{off}}{10^6}) \cdot f_{nom}$, which accounts for the deviation from the nominal frequency. Using this adjusted expression for the clock frequency, the basic model from Equation 2.1 can be extended to incorporate the frequency offset:

$$t_{real}(t_{loc}) = \frac{t_{loc}}{(1 + \frac{F_{off}}{10^6}) \cdot f_{nom}} + T_{off} \tag{2.3}$$

This modified equation allows for a more accurate representation of the relationship between the local time and the real physical time, considering the linear time drift introduced by the frequency offset of the clock generator.

In addition to frequency offset, aging of the clock generator introduces another source of systematic frequency instability. Unlike frequency offset, which remains constant, the frequency shift due to aging progressively increases over time at a constant rate D [44]. This leads to the local time of the device drifting at an ever-increasing rate, causing a cumulative time error that becomes more significant over long periods of operation.

The progressive frequency shift over time, denoted as f_{drift}, can be mathematically expressed as $f_{drift}(t) = D \cdot t$. To estimate the time shift caused by the aging process, we can integrate this frequency drift over time, yielding the following equation for the local time drift:

$$t_{loc}(t_{real}) \; = \; \int (D \cdot t_{real}) dt_{real} \; = \; \frac{D}{2} \cdot t_{real}^2 + C \tag{2.4}$$

In this context, since we are only focusing on the additional time drift caused by aging, it is valid to assume $C \; = \; 0$ in Equation 2.4. This simplifies the model, and the real time t_{real} can now be related to the local time t_{loc} as:

$$t_{real} = \sqrt{\frac{2 \cdot t_{loc}}{D}} \tag{2.5}$$

By incorporating this aging effect into the previously derived model of local time (Equation 2.3), we obtain a more detailed equation that accounts for both the frequency offset and the progressive drift caused by aging. The enhanced model is given as:

$$t_{real}(t_{loc}) = \frac{t_{loc}}{(1 + \frac{F_{off}}{10^6}) \cdot f_{nom}} + T_{off} + \sqrt{\frac{2 \cdot t_{loc}}{D}} \tag{2.6}$$

In applications requiring extreme long-term stability of the local clock, the generalized aging model may not provide sufficient accuracy. To address this limitation, it is possible to identify the specific aging pattern of the oscillator in use and apply

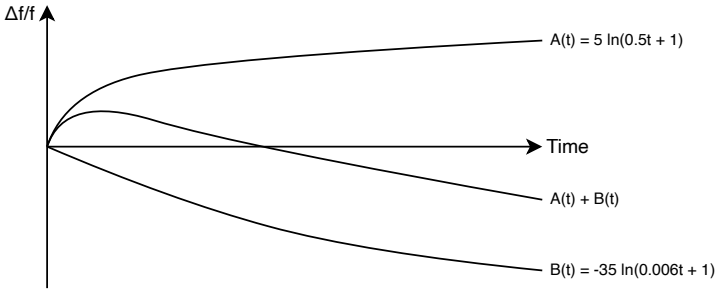


Figure 2.2: Typical aging behaviors.

one of the typical oscillator aging models as shown in Figure 2.2 [45, 46]. These models enable more accurate prediction of aging-related frequency drift and help to minimize the accumulation of local time errors over long durations of operation.

Alternatively, to minimize the effects of aging from the outset, an aging-compensated oscillator can be used. These oscillators are designed to correct or reduce the frequency drift associated with aging, improving long-term stability and reducing the need for frequent recalibration or compensation [47].

2.3.2 Short-term clock stability

In addition to long-term clock instabilities, there are physical processes that cause oscillators to change their phase and frequency on a smaller time scale. These processes include environmental changes such as pressure and humidity, as well as inherent phase noise. These instabilities arise from various sources, including both internal and external factors and introduce significant challenges to maintaining precise synchronization in systems where clock stability is critical.

It is possible to measure the dependency of clock stability on environmental changes and use these measurements to correct the induced errors. However, achieving high accuracy typically requires a tedious calibration procedure for each

individual oscillator, as the sensitivity of oscillators to environmental factors can vary significantly. Specifically, temperature is known to have the greatest impact on oscillator stability. To avoid the need for constant calibration to account for temperature variations, two specialized types of oscillators are commonly used: oven-controlled crystal oscillators (OCXOs) and temperature-compensated crystal oscillators (TCXOs). Although both types are more expensive and physically larger than standard oscillators, they offer significantly enhanced temperature stability, making them ideal for applications where such stability is paramount.

OCXOs and TCXOs achieve this stability through fundamentally different mechanisms. OCXOs include a heating element and control circuitry that maintain the oscillator at a constant internal temperature, independent of the surrounding environment. This design provides exceptional temperature stability, though it comes with increased complexity and power consumption [48]. TCXOs, on the other hand, utilize temperature-sensitive circuitry to dynamically adjust the frequency of the oscillator based on ambient temperature. While TCXOs offer less stability compared to OCXOs, they are more cost-effective and simpler in design, yet still provide better temperature compensation than standard, uncompensated oscillators.

Predicting the exact variance in clock properties due to environmental changes is difficult, as these factors often exhibit quasirandom behavior within a certain margin. This margin is typically specified by the manufacturer for each specific oscillator model. In addition to external environmental factors, oscillators are also subject to internal instabilities caused by processes such as thermal noise, mechanical disturbances, or fluctuations in the power supply. These instabilities tend to occur at higher frequencies than environmental effects, contributing to short-term errors that are more difficult to mitigate. When discussing clock stability, it is common to differentiate between slower, long-term variations caused by factors like aging or gradual temperature changes, referred to as wander, and faster, random variations due to noise and jitter. This distinction is demonstrated in Figure 2.3.

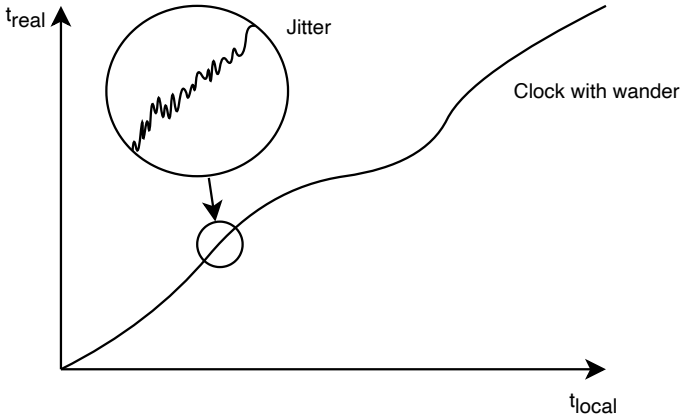


Figure 2.3: Wander and jitter effects on the clock stability.

2.4 Time synchronization

A variety of synchronization methods has been developed to meet the different needs of applications, each with its own precision requirements and constraints. These methods can be organized into categories based on two main factors: the degree of synchronicity they provide and the overhead complexity they require. The degree of synchronicity refers to how closely the nodes or devices can be aligned in time, ranging from loose coordination to very high precision. The overhead complexity involves the resources needed for implementation, such as computational power, communication bandwidth, energy consumption, and system infrastructure.

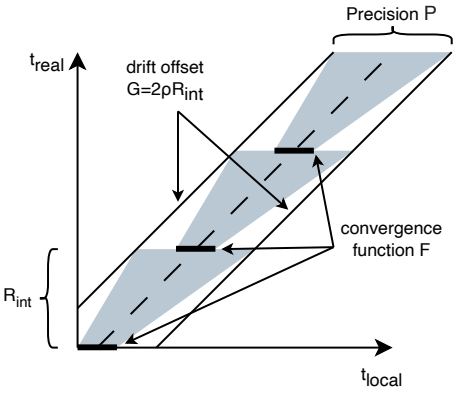


Figure 2.4: Synchronization condition in a plesiochronous network.

2.4.1 Synchronization condition

Frequency differences between nodes in a network cause a linearly increasing time error as the system operates. Factors like frequency drift and phase noise intensify these time discrepancies. To prevent the time error from exceeding a predefined threshold, clocks must be periodically resynchronized within the network. The interval between these corrections is known as the resynchronization interval, and at the end of each interval between these corrections, the clocks are adjusted to bring them back into better agreement.

During the free-running period between resynchronizations, the clocks drift apart due to natural clock variations, as presented in Figure 2.4. The maximum divergence during this period is referred to as the drift offset G, which depends on the length of the interval and the maximum specified drift rate ρ of the clock [41]. The drift offset is calculated as:

$$G = 2\rho R_{int}$$

At the end of each resynchronization interval R_{int}, the clocks are realigned. However, after the clocks are synchronized, they start drifting again until the next interval ends. The amount of divergence between two good clocks during this interval is defined as the drift offset. The synchronization process must ensure that the offset immediately after resynchronization, defined by the convergence function F, combined with the drift offset G, remains within the precision requirement P:

$$F + G \leq P$$

This condition ensures that the clocks stay synchronized within the precision limit throughout the resynchronization cycle. If the combined drift and offset exceed the specified precision, the time error could grow too large, disrupting synchronization. Therefore, the period between clock updates depends on the precision and stability of the clocks, as well as the allowed time difference between the nodes. In the end, the synchronization condition specifies that the algorithm must synchronize the clocks with sufficient accuracy so that, during the next free-running resynchronization period, the deviation between clocks remains within the defined precision interval.

2.4.2 Synchronization methods

This classification helps to choose the most suitable approach for a given application, depending on the balance between the required precision and the system limitations. Applications that need very high accuracy often require more complex methods, while those with lower precision needs can use simpler approaches with less resource overhead. This classification helps to highlight the trade-offs between accuracy and complexity, providing a clearer basis for selecting the right synchronization method (Figure 2.5) [49].

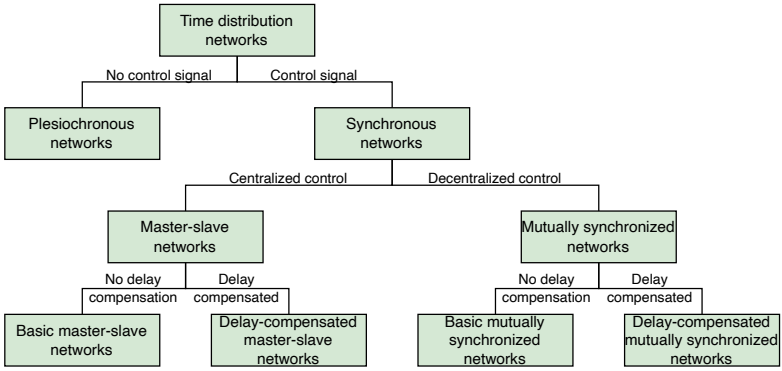


Figure 2.5: Classification of time and frequency distribution networks.

2.4.3 Plesiochronous networks

In a **plesiochronous** network, each node operates with its own precise clock, and no control signals are used to coordinate the operation of these clocks. During the initial setup, clocks are calibrated to minimize the time difference between them, though perfect synchronization is often not feasible. This calibration can either be performed centrally, before deployment, or through a "traveling clock" technique [50]. In the traveling clock method, a highly accurate reference clock is physically transported to each node, where it is used to adjust the local clock. This process ensures that each node is synchronized as closely as possible to the reference time. However, due to the independent nature of the clocks, slight differences in their free-running frequencies remain inevitable.

Plesiochronous networks offer several advantages, most notably their ease of implementation and their robustness against individual clock failures. Since each node operates independently, the malfunction or failure of one clock does not impair the performance of other nodes in the network, providing a certain level of fault tolerance. This decentralized design allows the network to continue functioning even if isolated failures occur, enhancing the overall reliability of the system.

However, plesiochronous networks have also significant drawbacks. The primary challenge lies in the high costs associated with procuring and maintaining highly accurate clocks. Achieving the required level of synchronization precision demands the use of expensive oscillators, and the need for regular updates to prevent time drift further increases the operational complexity.

2.4.4 Synchronous networks

In **synchronous** networks, all clocks are aligned in both time (phase) and frequency to a common reference, ensuring that clocks across different locations generate nearly identical time scales. The methods used to achieve synchronization can be classified into two main types: centralized and decentralized approaches. These methods differ primarily in how control signals are managed to maintain synchronization.

In centralized networks, the master-slave technique is typically employed, where a single master clock dictates the network time scale and frequency. All other clocks are either directly or indirectly synchronized to this master clock, forming a hierarchical structure, as shown in Figure 2.6. In contrast, decentralized networks operate using mutual synchronization, where no master clock exists. Instead, all clocks collaborate and contribute equally to establishing the network time and frequency. This mutual approach allows the network to function without a single point of control.

The master-slave synchronization technique is one of the most widely used methods for synchronizing clocks in networks. In this method, a master clock, such as T1(t), controls the timing for the entire system, while slave clocks, like T2(t), adjust their time based on the master. To synchronize, the master clock sends its time to the slave, as shown in Figure 2.7. The slave compares the time signal from the master, T1(t-τ), delayed due to transmission, with its own time using a time difference detector. The resulting error signal is then used to adjust the slave clock.

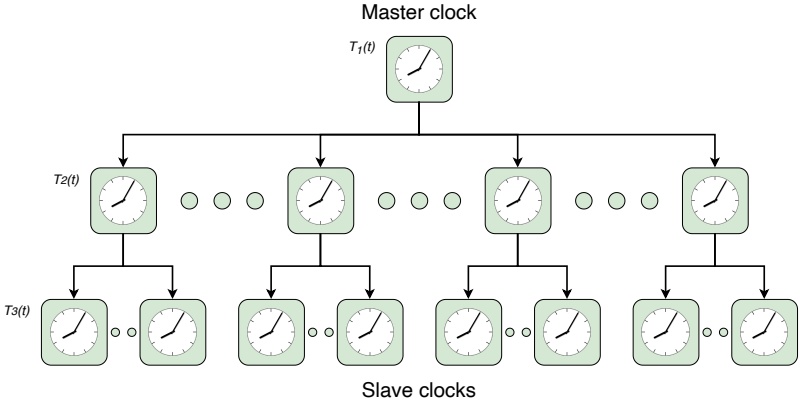


Figure 2.6: A hierarchical master-slave network.

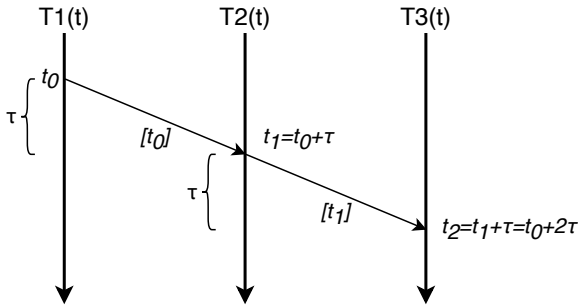


Figure 2.7: Time propagation in a master-slave synchronous network.

In steady-state operation, the slave clock is locked to the delayed time of the master clock, creating a steady time difference between the two clocks due to transmission delay. Any changes in transmission delay directly affect this time difference. In a hierarchical network, where clock T2(t) also serves as the master for clock T3(t), the overall time error between T1(t) and T3(t) is the sum of the errors between clocks T1(t) and T2(t), and clocks T2(t) and T3(t).

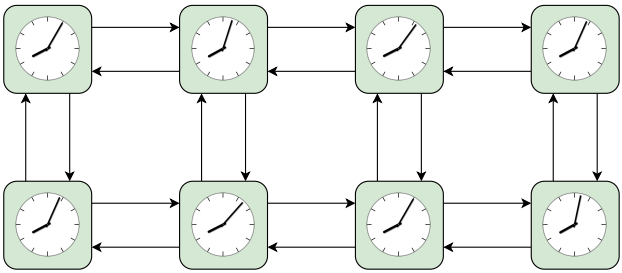


Figure 2.8: A fully-connected mutually synchronized network.

A key challenge in master-slave networks is dealing with transmission delay, which causes a time bias between the clocks. To mitigate this, delay compensation techniques are used. A common method involves estimating the transmission delay and applying this estimate to delay the slave clock before comparing it with the master time. When the estimate is accurate, this eliminates the steady time bias. However, if the estimate is off, a residual time error remains.

One significant drawback in master-slave networks is the potential loss of the master clock. To maintain stability, backup strategies are necessary. One option is to designate alternative master clocks. Another is to let slave clocks operate independently until the master clock is restored, which often requires highly accurate and expensive backup clocks. Because of the risk of losing the master clock, this method may not be ideal for mission-critical applications. However, the master-slave technique remains popular in fault-tolerant networks, like telephone systems, due to its simplicity.

In a mutually synchronized network (Figure 2.8), there is no single master clock responsible for controlling the timing. Instead, the synchronization process is distributed across all clocks, with each one contributing to the establishment of a common time in the network. Every clock continuously adjusts its own time by comparing it to the times of other clocks in the system. The goal is to minimize the time difference between its own clock and a weighted average of the other clocks. This distributed approach ensures that no single clock controls the overall

synchronization process, and all nodes equally participate in maintaining the network timing.

At each node, an error signal is produced by comparing the incoming time signal from a neighboring node with the local clock. This is similar to the error generation process in a master-slave network, but without any central authority. The error signal is then used to adjust the local clock to reduce the time difference. As in the master-slave system, the steady-state time difference between clocks depends on the transmission delays between nodes. Additionally, the network frequency is affected by these delays. Various delay-compensation techniques, similar to those used in master-slave systems, can be applied to reduce the impact of these delays.

In larger networks where each node is connected to multiple neighbors, the challenge becomes more complex. The clock-correction signal must be computed from multiple time differences. Phase averaging is a common solution in this case. Each node receives time signals from all connected neighbors and computes the time difference between the received signals and its own local time. The clock correction is then calculated as a weighted sum of all these differences. In most cases, this simply involves summing the error signals, allowing the clock to adjust its time based on the inputs from all neighboring nodes.

Mutually synchronized networks have several advantages. The lack of a master clock reduces the risk of a single point of failure, making the system more robust. However, one of the main challenges is stability, especially during transient conditions. Closed loops in the network can lead to instability under certain parameter settings, which must be carefully managed. Another potential issue is the vulnerability to malicious nodes. For example, in a network with three clocks, if one clock (node C) behaves maliciously, it can disrupt the synchronization process. By sending manipulated time signals, clock C can prevent the other two clocks (nodes A and B) from synchronizing properly. As a result, clocks A and B may fail to correct their time values and eventually violate the synchronization condition, leading to network failure (as shown in Figure 2.9).

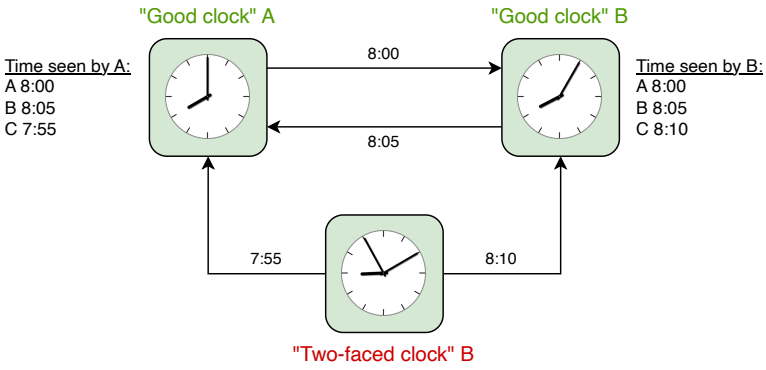


Figure 2.9: A malicious node preventing a consensus in a mutually synchronized network (Byzantine error).

2.5 Triggering in particle physics experiments

In modern particle physics experiments, many of the detected particles are not relevant to the scientific goals of a particular experiment. These measurements, called "background," involve well-known phenomena or interactions outside the experiment focus. Interactions producing such background measurements happen much more often than the interactions of interest, creating a large amount of irrelevant data [31]. Managing this excess background data is essential to accurately identify and study the significant interactions, which are key to making new discoveries in the field. The challenge lies in efficiently distinguishing between this background noise and the rare, meaningful events without overwhelming the data processing and storage systems.

The digitization and initial aggregation of the raw data are managed by ASICs and FPGAs [51]. These specialized hardware components perform preliminary data processing tasks, such as digitization, zero suppression, and initial filtering, before passing the information on to the data storage systems. Given their proximity to the detectors, these components benefit from exceptionally high bandwidth,

allowing them to handle vast parallel data streams. This proximity ensures that data can be processed and transmitted at the same rate it is generated, which is critical given the sheer volume of information produced in high-energy collisions. However, the buffering capacity at this stage is notably low due to the limited resources of each device, meaning that data must be processed and filtered in real-time to prevent loss.

The reliance on hardware for these initial processing tasks introduces certain limitations. ASICs and FPGAs, while incredibly fast and efficient, are designed for specific, fixed functions. This restricts the flexibility and complexity of operations that can be performed near the detector, as they lack the adaptability of software-based systems. This limitation necessitates that only the most basic and essential data reduction operations be carried out at this early stage. Nevertheless, these early operations are crucial, as they reduce the data volume sufficiently to prevent bottlenecks in subsequent processing stages.

As the data moves closer to the storage systems, the available bandwidth becomes more constrained, making efficient data management essential to avoid bottlenecks that could cause data loss [52]. However, at this stage, the increased availability of buffering space and more affordable processing power allows for the implementation of more sophisticated data selection algorithms. These algorithms, often executed on general-purpose processors or high-performance computing clusters, can analyze data in greater detail, applying complex criteria to further refine which data should be preserved.

This gradient in available bandwidth and processing capability throughout the data acquisition chain necessitates the early implementation of data filtering mechanisms. These mechanisms are critical for reducing the volume of data being transmitted and ensuring that only the most relevant information reaches the later stages where it can undergo more thorough processing and eventual storage. The early-stage filtering is vital not only for managing the limited buffering capacity near the detectors but also for optimizing the overall efficiency and accuracy of the data acquisition system in high-energy physics experiments.

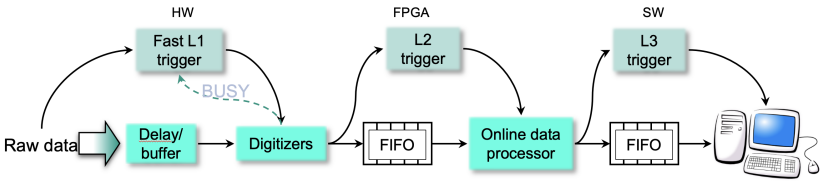


Figure 2.10: An example of a multi-layer trigger system in a particle physics experiment.

To implement this gradual data filtering, high-energy physics experiments employ a multi-layer system of data triggers, as shown in Figure 2.10. These triggers analyze a small fraction of the data to test specific filtering conditions, enabling the system to efficiently decide which data to retain or discard. The triggers are organized in layers, each with different capabilities and decision latencies. Low-level triggers, typically implemented in hardware, perform basic analyses to make immediate select/reject decisions with extremely low latency, often on the order of microseconds. This rapid response is crucial to prevent the downstream data processing stages from being overwhelmed by the sheer volume of information generated near the detectors.

In contrast, high-level triggers, usually implemented in software, are deployed at later stages in the data acquisition chain where there is more computational power and buffering space available. These triggers have access to significantly more detailed information and can employ complex algorithms to refine data selection further. Although these analyses introduce higher latency, the trade-off is justified by the need for more precise data processing at this stage, ensuring that only the most relevant events are stored for long-term analysis. This stage sophistication allows for the implementation of advanced data selection techniques that would not be feasible at the hardware level.

The trigger systems can be highly complex, often incorporating more than two layers, each offering varying processing capabilities and decision latencies. Multiple triggers at the same level can also operate in parallel, each testing different conditions or focusing on different aspects of the data, as illustrated in Figure 2.11. This parallelism enhances the system ability to identify and retain data most

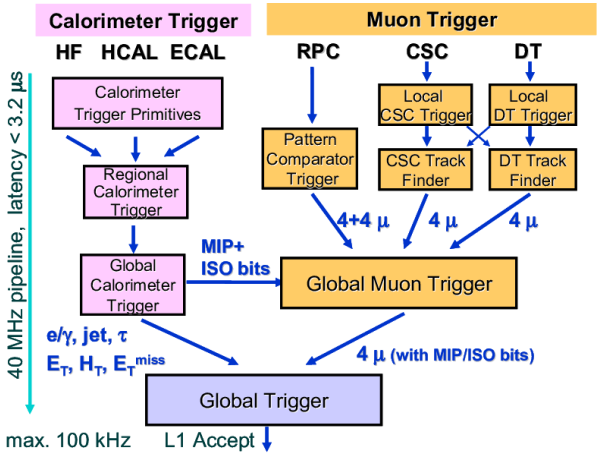


Figure 2.11: A parallelized trigger system in the Compact Muon Solenoid (CMS) experiment (image used under CC BY-NC-SA 1.0) [53].

likely to contain significant interactions, further refining the overall data set that is preserved for detailed analysis. The multi-layered triggering approach thus plays a crucial role in managing the vast amounts of data generated by detectors, ensuring that only the most pertinent information is retained for further study.

Recent advancements in high-speed communication and scalable computing have led to the adoption of streaming data acquisition schemes in some experiments. In a streaming data acquisition system, data is transported unchanged directly to the high-level trigger, where sophisticated software running on high-performance computers analyzes the entire data stream in real-time. This method offers a more thorough and flexible analysis, as the software can access the full data stream and apply complex algorithms to select relevant data in a single stage. This approach can improve the overall efficiency and accuracy of data acquisition by reducing the risk of missing important events due to early-stage filtering. A comparison of a triggered and a streaming data acquisition chain is shown in Figure 2.12.

However, streaming data acquisition is not suitable for all experiments, particularly those that generate extremely large volumes of data. In such cases, the sheer amount of raw data produced makes it impractical to transport everything to the high-level

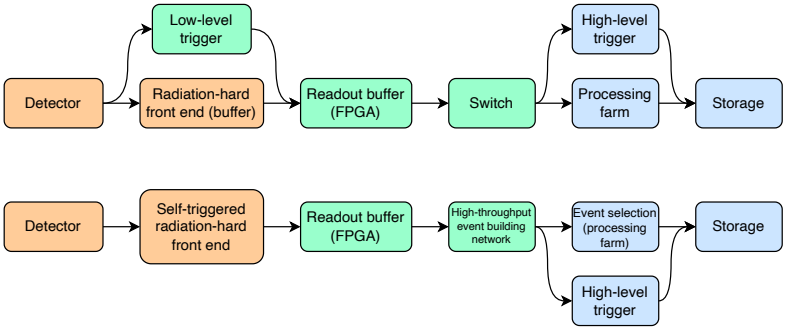


Figure 2.12: A comparison of a triggered (top) and a streaming (bottom) data readout concepts.

trigger without some form of initial data reduction. Therefore, hardware triggers remain necessary for many experiments to perform this initial filtering, ensuring that irrelevant information is discarded early on and preventing overload in the later processing stages.

The benefits of streaming data acquisition include improved accuracy in data selection and the ability to handle event pileups, the situations where multiple interactions occur so close together in time that they overlap in the detector. Advanced software algorithms can unravel these complex events more effectively than hardware-based systems. While streaming data acquisition is not feasible for all experiments, particularly those with extremely high data rates, it offers significant advantages over traditional triggering schemes in appropriate contexts. By providing a more flexible and comprehensive analysis, it enhances the accuracy and adaptability of data selection, ultimately improving the quality of the scientific results in high-energy physics experiments.

2.5.1 CBM overview

Currently under construction in Darmstadt, Germany, the FAIR facility (Figure 2.13) will be a versatile accelerator complex designed to test the limits of the fundamental theories of the modern physics [54]. At its core lies the SIS100



Figure 2.13: Construction of the FAIR facility as of October 2025. (Credit: D. Fehrenz, GSI/FAIR)

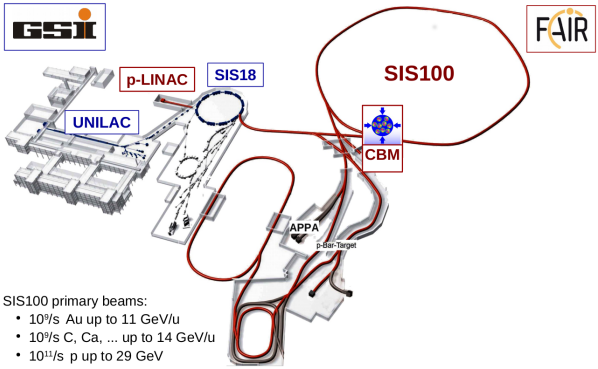


Figure 2.14: FAIR accelerator complex with the location of the CBM experiment. (Credit: CBM collaboration)

synchrotron, a driver accelerator with a circumference over one kilometer. SIS100 will be capable of accelerating all stable elements, from protons to uranium, and will support the truly parallel operation of up to four large-scale research programs [55, 56]. The layout of the FAIR facility, including the location of the CBM experiment, is shown in Figure 2.14.

Among the flagship experiments at FAIR is the CBM experiment (Figure 2.15), which will utilize the high-intensity beams provided by SIS100. As illustrated in Figure 2.16, CBM is designed to investigate the phase diagram of strongly

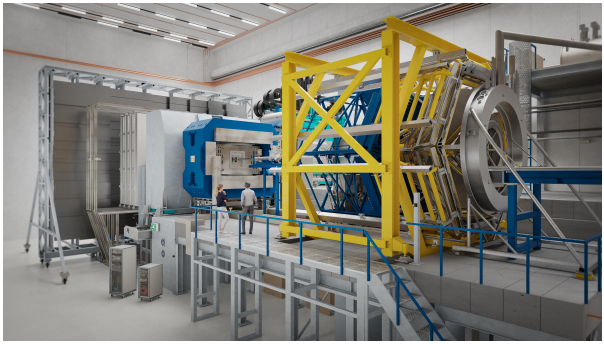


Figure 2.15: Digital visualization of the CBM cave. (Credit: GSI/FAIR, Zeitrausch)

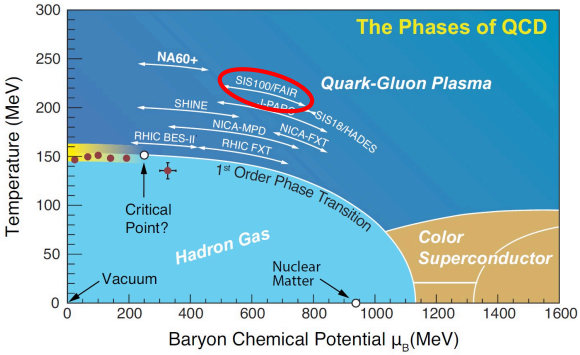


Figure 2.16: QCD phase diagram. The area being explored by the FAIR experiments is highlighted. (Credit: CBM collaboration)

interacting matter described by quantum chromodynamics (QCD), focusing on the region of high baryon densities.

The experiment will explore matter under extreme conditions similar to those believed to exist in the cores of neutron stars and during neutron star mergers, thereby providing insights into celestial bodies otherwise inaccessible to direct observation [57]. The slow beam extraction from the SIS100 accelerator provides CBM with a quasi-continuous beam to probe the high-density baryonic matter with an unprecedented precision [58]. This capability is achieved through the operation

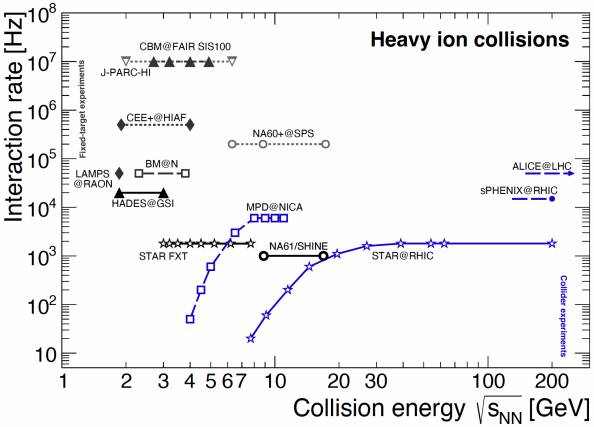


Figure 2.17: Comparison of interaction rates of various experiments, including CBM [59].

of CBM as a versatile multi-purpose detector with exceptionally high interaction rates, surpassing those of previous experiments, as shown in Figure 2.17 [59, 60].

The CBM detector setup is a complex heterogeneous detector system consisting of individual detectors with different fundamental principles of operation and purposes:

- The *Beam Monitor (BMON)* consists of two diamond-based detector stations placed upstream of the target. It provides fast beam-related timing and monitoring signals, including a start-time measurement with a precision of about 50 ps and beam halo observation.
- The *Micro-Vertex Detector (MVD)* is a high-resolution silicon pixel detector designed for precise position measurement close to the interaction point. It comprises four planar stations based on Monolithic Active Pixel Sensors (MAPS) and can be configured either for improved spatial precision or for enhanced tracking capability.
- The *Silicon Tracking System (STS)* is the main charged-particle tracking detector of CBM. It uses eight layers of double-sided silicon micro-strip

sensors and provides trajectory reconstruction and momentum determination within the magnetic field.

- The *Muon Chamber System (MUCH)* combines absorber structures with multiple layers of gaseous tracking detectors. Its purpose is to filter strongly interacting background particles and to identify penetrating particles by reconstructing track segments behind successive absorber stages.
- The *Ring Imaging Cherenkov Detector (RICH)* is used for electron identification and is based on the detection of Cherenkov photons in a gas radiator. Its optical system includes focusing mirrors and photon detector planes based on multi-anode photomultiplier tubes (MAPMTs).
- The *Transition Radiation Detector (TRD)* provides additional tracking information and supports electron identification at higher particle energies. It consists of four detector layers with pad readout and position-sensitive measurement.
- The *Time-of-Flight system (TOF)* is based on Multi-gap Resistive Plate Chambers (MRPCs) and enables particle identification through precise timing measurements with a resolution on the order of 80 ps.
- The *Projectile Spectator Detector (PSD)* is a modular lead-scintillator calorimeter used for event characterization. It measures forward-going energy deposition and provides information related to collision geometry.

2.5.2 Free-streaming data readout in the CBM experiment

In the CBM experiment, achieving high interaction rates is essential for detecting rare particle interactions, such as multi-strange hyperons and charmed hadrons, which occur very infrequently in heavy-ion collisions [30]. These rare events are crucial to the experiment objectives, but because they are so uncommon, observing a large number of collisions is necessary to gather enough data for a meaningful

analysis. With interaction rates as high as 10^7 events per second, the CBM experiment can collect sufficient data on these rare occurrences within a practical timeframe. Without such high interaction rates, detecting these rare events would be far more challenging, and the experiment would require significantly more time to achieve statistically meaningful results.

The challenge of capturing these rare signals is further complicated by the vast number of more common particle interactions that occur during collisions. To extract valuable information from the massive volume of data, the system must be able to filter through millions of events per second. High interaction rates allow CBM to maximize the likelihood of capturing the rare events hidden within this data, ensuring that the experiment can achieve its scientific goals within a reasonable period.

The data rate generated by the detectors and transferred through the DAQ system is expected to exceed 1 TB/s. To efficiently manage the high data volumes generated by these high interaction rates, CBM uses a free-streaming readout system [61]. This approach continuously captures all the data from the detectors without relying on predefined hardware triggers to decide which events to keep. In conventional systems, only specific events that match preset conditions are recorded, potentially missing unexpected or unusual phenomena. The free-streaming system ensures that no data is discarded prematurely, allowing advanced software to later filter out irrelevant events and focus on the ones that matter. This is particularly important for CBM, where rare signals may not conform to simple trigger conditions. By capturing all data and applying complex filtering software later, the system increases the likelihood of detecting important, rare events.

The DAQ system is designed to handle the continuous flow of data with a 1 TB/s rate in the free-streaming environment. The first stage of the system involves the FEE, which convert the signals generated by particle interactions into digital data. These digital signals are time-stamped and sent to readout boards (ROB), which aggregate the data from multiple detectors. The ROBs transmit this data via optical links to the First-Level Event Selector (FLES), the core of the DAQ system [62]. The FLES assembles the data into timeslices based on the timestamps

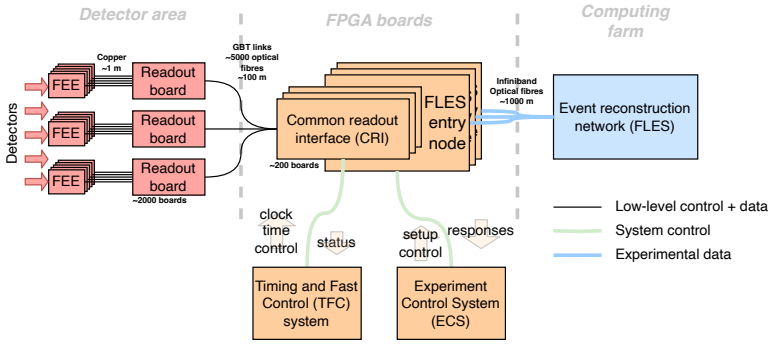


Figure 2.18: The architecture of the CBM DAQ system with the central control systems included.

and processes them in real-time, filtering out irrelevant events and keeping only those of interest for further analysis [63]. This architecture allows the DAQ system to efficiently manage the high interaction rates and handle the enormous volumes of data generated by the detectors. The overall architecture of the DAQ system, including the experiment control system (ECS) and the TFC, is shown in Figure 2.18.

One of the critical aspects of this architecture is synchronization. In CBM, each detector operates independently in a self-triggered mode, generating its own time-stamped data based on particle interactions. For the system to accurately reconstruct individual events, all the data from different detectors must be aligned in time. Synchronization ensures that signals from different parts of the detector system, which may have been recorded at slightly different times, can be combined to reconstruct a complete event. Without proper synchronization, the data from different detectors would be misaligned, making it difficult or impossible to accurately assemble the event. Precise synchronization is essential to maintaining data integrity and ensuring that events are reconstructed correctly.

In a high-rate, free-streaming system like CBM, fast control mechanisms are also required to maintain smooth operation. Fast control allows the system to adjust parameters dynamically in real time to prevent overloads and ensure that data

continues to flow efficiently. For instance, if the interaction rate spikes or certain detectors are overwhelmed by too much data, fast control systems can quickly react and issue a command to adjust the thresholds or stop accepting new data to prevent uncontrolled data loss [34]. This capability is vital for ensuring the DAQ system remains responsive and can continue functioning efficiently, even under varying experimental conditions. Fast control helps maintain a balance between the data output of the detectors and processing capabilities of the DAQ system, preventing disruptions and ensuring that data is collected and processed in an optimal manner.

Fast control also allows the system to react to changes in experimental conditions, such as variations in beam intensity or interaction rates. If necessary, the system can dynamically adjust how it processes data or how it configures detectors to adapt to these changes. This adaptability ensures that the CBM experiment can continue running efficiently and consistently, without manual interventions or lengthy downtimes for system reconfiguration.

3 State of the art

This chapter summarizes the current solutions for achieving real-time operation of the modern distributed systems. Starting off with an overview of two prominent and widely deployed solutions, White Rabbit and Time-Sensitive Networking, the focus of the chapter is narrowed down towards the solutions employed in various particle physics experiments with the real-time networking requirements similar to those in CBM.

For each of the reviewed technologies, the major motivation is presented, as well as its particular principle of operations. For a better understanding of the applicability of the solutions, each of the overviews in this chapter includes also the range of applications and the capabilities with regard to scalability, synchronization accuracy and determinism of the network message transport.

3.1 White Rabbit

Initially developed to synchronize the European Organization for Nuclear Research (CERN) accelerator network, the White Rabbit project set out to provide highly accurate time synchronization combined with high-bandwidth data links and dynamic compensation of propagation delays. The requirements defined for White Rabbit were ambitious: it was to enable time transfer from a central reference node to thousands of endpoints with sub-nanosecond accuracy, support distances of up to 10 km between nodes, scale to more than 2000 network nodes, and guarantee bounded latency for data transfers from the central node to its endpoints [64]. Given these requirements, White Rabbit clearly qualifies as a DRTS.

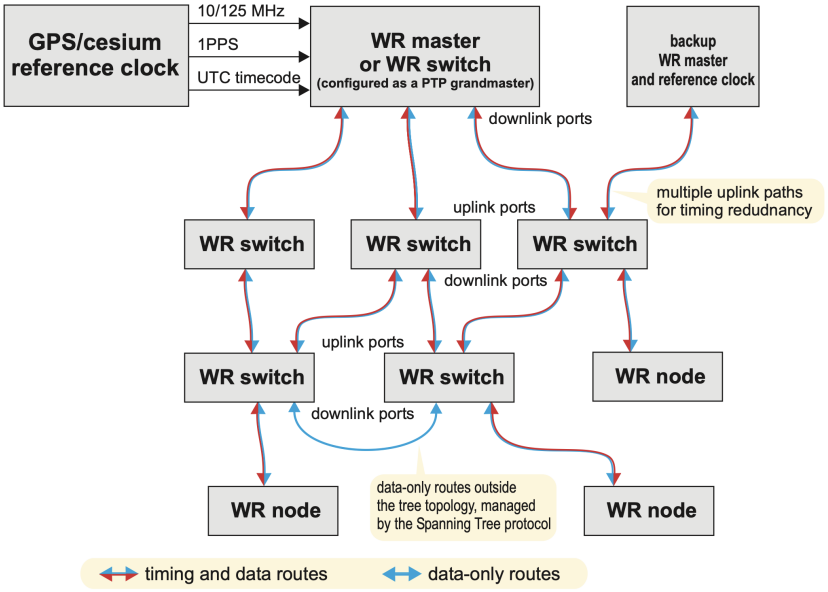


Figure 3.1: Topology of a White Rabbit network. (Credit: Tomasz Włostowski) [68].

The project has been developed building on existing Ethernet-compatible standards and extending them only where necessary. This approach made it possible to leverage the interoperability of Ethernet, its advanced infrastructure, efficient switching techniques, and high bandwidth capabilities [65]. At the same time, White Rabbit maintained backward compatibility and embraced an open-source development paradigm. In particular, it incorporated existing timing standards, Synchronous Ethernet (SyncE) [66] and the Precision Time Protocol (PTP) [67], as the foundation for its synchronization strategy. Because White Rabbit is Ethernet-compatible by design, the timing tree is implicitly established within an Ethernet network of arbitrary topology through the distinction between downstream- and upstream ports of White Rabbit switches, as illustrated in Figure 3.1 [64].

At the physical layer, White Rabbit adopts syntonization from SyncE, which aligns the local oscillator frequencies of all network nodes without interfering

with the data flow. This reduces the timestamping limitations of PTP to a clock phase measurement problem [64]. Using the DDMTD technique, described in Section 5.2, the precision of PTP round-trip delay measurements is enhanced, achieving synchronization accuracy at the sub-nanosecond level. The system is versatile and continuously developed, supporting multiple transmission media and synchronization extensions that influence its overall performance. For instance, strict compliance with the SyncE standard has been shown to limit scalability to four network hops [69], yet deployed networks have demonstrated sub-nanosecond accuracy, with the best reported results achieving below 100 ps root mean square (RMS) over nine hops in the systems where this compatibility is not required [69, 70].

The White Rabbit project continues to be actively supported by numerous scientific institutes and research groups. Significant progress has been made in calibration techniques to further enhance performance [71, 72]. Other research has extended White Rabbit synchronization to much greater distances, reaching hundreds of kilometers [73–76]. Moreover, its application range has broadened well beyond particle physics: White Rabbit is now deployed in telescope arrays [77, 78], quantum communication networks [79, 80], and even finance applications [81, 82].

Despite these strengths, compatibility of White Rabbit with Ethernet-based standards introduces additional implementation complexity and imposes limitations on latency determinism. While time synchronization can be maintained with excellent precision, reported latency determinism reaches only 16 ns peak-to-peak over two hops [70]. As a result, although White Rabbit has been successfully integrated into some DAQ systems [83–85], it does not satisfy the particularly tight requirements of the CBM experiment, which demands fixed-latency transmission of control messages with greater determinism.

3.2 Time Sensitive Networking

TSN is a major initiative designed to overcome the long-standing problem of vendor lock-in in distributed real-time applications, a limitation previously encountered with FieldBus and Industrial Ethernet systems [86]. Originally developed as an extension of the Audio/Video Bridging standards for in-vehicle automotive communication and industrial automation [87], TSN aims to enhance Ethernet with features for reliability and real-time operation, while maintaining the maturity and ubiquity of Ethernet as a networking technology. Its objectives include reduced and deterministic transmission latency, independence from specific transmission media and physical rates, freedom from vendor-specific implementations, and inherent fault tolerance [88]. By achieving these goals, TSN enables real-time applications to benefit from the advantages of Ethernet networks, such as interoperability, high bandwidth, flexibility, and scalability [89].

A key feature of TSN relevant to this thesis is network-wide synchronization, which is essential for many time-critical distributed applications. TSN builds on the generalized Precision Time Protocol (gPTP) principles defined in Institute of Electrical and Electronics Engineers (IEEE) 1588 [67], supporting the formation of a synchronization tree in which a Grand Master clock serves as the root and all other nodes act as Clock Slaves [90]. Synchronization relies on measuring not only the propagation delays of frames along network links but also the residence times within intermediate nodes, which include processing, queuing, and transmission delays [91]. The protocol supports flexible network topologies with multiple potential Grand Master clocks, while the Best Master Clock Algorithm dynamically selects the optimal candidate to serve as the active root of the synchronization tree [92].

Once synchronization is established, TSN enables time-aware traffic management mechanisms. The most prominent of these is the Time-Aware Shaper (TAS), which applies time-division multiple access (TDMA) principles. Each switch port is equipped with transmission gates controlling priority queues, which are opened and closed according to a global schedule. This mechanism effectively isolates

traffic classes in time and provides predictable delays for time-triggered traffic [93]. Another method, the Peristaltic Shaper (PS), divides time into alternating odd and even intervals of equal size, with frames transmitted based on their arrival time [94]. If one interval is overloaded, transmission spills over into the next interval, effectively blocking it until the overload is resolved, thereby bounding the maximum residence time of data frames.

In addition to synchronization-based shapers, TSN also specifies asynchronous methods for bounding latency in high-priority traffic without the requirement of network synchronization. The Credit-Based Shaper (CBS) regulates access to priority queues by assigning and consuming credits depending on queue fill levels and waiting times, ensuring fair and bounded access [88]. Similarly, the Asynchronous Traffic Shaper (ATS) introduces an urgent priority queue at each intermediate node, always served before non-priority queues. Incoming frames must therefore be classified into priority or non-priority traffic upon arrival and directed accordingly. More recent developments, such as the Asynchronous Time-Aware Shaper (ATAS), attempt to map the benefits of TAS onto asynchronous networks. By tracking high-priority streams locally, ATAS estimates the expected arrival of critical frames and temporarily blocks non-priority traffic to reserve bandwidth [95].

Beyond traffic shaping, TSN defines additional features such as frame preemption, traffic replication, and path reservation, which further enhance its reliability for real-time communication [88]. However, these features are not covered as they are rather irrelevant for the context of the thesis.

Reported performance figures of TSN vary across implementations. For example, in a hybrid wired/wireless network with three switches for medical applications, latency of approximately 54 μs with less than 1 μs variance was achieved using gate control and frame preemption [96]. Another study, targeting ultra-reliable low-latency communication over mobile networks, demonstrated consistent latency with variations of only tens of nanoseconds across up to five hops on copper-based 1 Gbps Ethernet links [97]. Synchronization accuracy in TSN networks has generally been shown to be on the order of microseconds [98].

Despite its versatility, TSN is specified exclusively at the medium-access Open Systems Interconnection (OSI) layer. As a result, it does not address synchronization errors or latency variations originating from hardware implementation details. This limitation fundamentally constrains its achievable determinism. Therefore, while TSN provides a valuable general-purpose solution for distributed real-time communication across diverse application domains, it has fundamental limitations that are not compatible with the stringent requirements of the CBM experiment.

3.3 Time and control distribution in physics experiments

3.3.1 Mu3e - search for rare muon decays

The clock and reset distribution system of the Mu3e experiment [99] is being developed to guarantee time consistency and the delivery of time-critical control signals in its triggerless DAQ system [100]. At its core, a central 125 MHz clock must be distributed with an accuracy better than 10 ps to 278 front-end ASICs mounted on 114 front-end boards, as well as to four DAQ switches based on PCIe40 cards [101, 102]. Alongside the clock, the most critical control signal to be propagated is the global reset, which can target individual subsystems or the entire experiment. While the timing requirement for reset distribution is less stringent than for the clock—on the order of 4 ns, it remains an essential element of the system reliability.

Both the global clock and the reset signals are generated within the Clock and Reset Box, which integrates a Si5345 clock generator and a Xilinx Kintex-7 FPGA. From there, the signals are transmitted over two groups of four optical fibers each to an assembly of custom distribution boards. These boards use cascaded NB7L1008M fanout buffers to multiply each signal group into 144 optical outputs, yielding a total of 288 optical fibers connected to the front-end boards and DAQ switches [102]. On each receiving board, one reset fiber and one clock fiber are

connected and the signals are further distributed electrically to the local FPGA and to the front-end ASICs attached to that board.

In addition to the reset functionality, state transitions of the experiment, such as switching between idle and running, are also conveyed via the reset lines. The reset and state change commands are transmitted at 1.25 Gb/s using the inherent features of 8b/10b encoding, avoiding additional protocol overhead. The implemented system has demonstrated excellent performance in time distribution, with the accuracy of the clock distribution reported to be below 5 ps RMS [102].

3.3.2 LHCb - precision measurements of matter-antimatter asymmetry

The TFC system of the LHCb experiment at CERN was developed to distribute the global clock and time-critical commands across its trigger-free DAQ system. To achieve this, the system leverages passive optical splitting, which allows a scalable and efficient fanout of signals to thousands of endpoints. The TFC propagates the 40 MHz bunch clock to the readout electronics, distributes fast control commands, and collects monitoring information from the DAQ system [103]. A representative application of the fast control function is trigger throttling [104]. In this mechanism, the buffer status of the DAQ cards is monitored, and whenever the DAQ system is unable to process the event stream at the incoming detector rate, the rate of the central trigger is reduced. This temporarily relieves the data readout components and prevents buffer overflows.

For data transport, the TFC system employs a protocol largely based on the Timing, Trigger and Control Passive Optical Network (TTC-PON) project [105], embedding the bunch clock directly into the data stream that also carries fast control commands. The DAQ system consists of approximately 2000 front-end boards read out by 480 DAQ cards, all of which act as endpoints of the TFC system. At the top of the hierarchy, the central Readout Supervisor connects to 42 Control Cards via 1:32 passive optical splitters. Both the Control and DAQ cards are implemented on PCIe40 boards [101]. Further downstream, the Control

Cards fan out the signals directly to the front-end boards and to the DAQ cards through an additional layer of 1:32 optical splitters.

Upstream communication from the front-end boards and DAQ cards back to the supervisor is implemented using a time-division multiplexing scheme. While this approach inherently limits the achievable latency of upstream messages, the reported value of 8 μs is sufficient to ensure proper operation of the trigger throttling mechanism. In terms of clock performance, the system was designed to achieve a maximum skew of less than 500 ps between the bunch clocks across all nodes and to limit the total RMS jitter of each recovered clock to below 100 ps. Experimental evaluation confirmed compliance with these requirements, with a maximum measured skew of 254 ps and a total jitter of 76 ps, demonstrating the system suitability for the stringent timing needs of LHCb [104].

3.3.3 ALICE - investigation of quark-gluon plasma

In the triggerless data acquisition scheme of the ALICE experiment at CERN, the Trigger and Timing distribution System (TTS) is responsible for distributing the global clock across the DAQ system and for communicating critical timing, trigger, and control signals to the readout electronics with fixed latency. In addition, the TTS collects busy signals from subsystems to throttle trigger distribution, ensuring reliable data collection at the extreme throughput of 3.6 TB/s [106]. To monitor and regulate the data flow through the readout chain, the system employs heartbeat frames [107]. These frames are generated at regular intervals of approximately 88.92 μs that match the LHC orbit length and serve to delimit the data stream. Depending on operational scenarios, heartbeat frames can be used to downscale the data flow and relieve the DAQ system in cases of overload.

At the center of the heterogeneous TTS architecture is the Central Trigger Processor (CTP), which generates the global clock, timestamps events, and manages the heartbeat frame monitoring and control. The CTP distributes the required timing and control signals via optical links to 18 Local Trigger Units (LTUs). Both

the CTP and LTUs are implemented as custom VME cards, each featuring nine downstream optical connections alongside additional connectivity options.

From the LTUs downstream, different distribution schemes are employed depending on the requirements of individual detectors. For detectors with stringent trigger latency requirements, the LTU is connected directly to GigaBit Transceiver (GBT)-based front-end boards via optical or electrical low-voltage differential signaling links, depending on the detector configuration [106]. In detectors where latency is less critical, the LTU downstream links fan out through 1:64 passive optical splitters to a total of 441 Common Readout Units (CRUs) [107]. These CRUs, in turn, deliver the timing and trigger information to the front-end electronics.

The synchronization performance of the TTS was specified to meet maximum allowable clock jitter between 300 ps and 20 ps, depending on the detector. System evaluation demonstrated performance surpassing these requirements, achieving clock distribution with jitter consistently below 10 ps across all configurations [106].

3.3.4 Dune - precision studies of neutrino oscillations

The DUNE Timing System (DTS) is part of the far detector of the DUNE experiment [108] and is being designed to provide its detector electronics with the common clock, 64-bit time and, potentially, control commands over downstream links with a fixed latency [109]. The global system clock is derived from an external GPS-sourced 10 MHz reference and converted to a 62.5 MHz system clock by the GPS Interface Board (GIB), which acts as the root of the synchronization tree. From the GIB, clock and time are distributed using a custom optical protocol via single-mode fibers to the detector caverns located 1.5 km underground [110].

Each cavern houses two redundant μTCA crates that receive the timing data from the GIB. Within each crate, MicroTCA interface boards distribute the signals over the backplane to up to 12 Fiber Interface Boards (FIBs). Each FIB provides eight SFP slots for further downstream transmission of the timing signals. To

ensure redundancy without requiring physical intervention in the difficult-to-access caverns, the optical links from the FIBs in the two crates are combined using passive optical combiners. After the combiners, the signals are fanned out with 1:8 passive optical splitters and delivered to the detector electronics.

Communication in the DTS is bidirectional, using different wavelengths for upstream and downstream transmission within the same single-fiber cables. The upstream channels are used for reading back status information from the endpoints and for returning synchronization data that allows the GIB to dynamically measure link latencies. A remarkable feature of the DTS is its use of Duty Cycle Shift Keying (DCSK) as the modulation technique. With DCSK, the duty cycle of the transmitted signal encodes binary values, with 25% duty cycle representing a logical `0` and 75% representing a logical `1`. This provides a raw data rate of 62.5 Mb/s and allows the signal to be fed directly into a PLL, eliminating the need for conventional clock-data recovery (CDR) techniques. Combined with 8b/10b encoding for DC balancing, this scheme enables robust recovery of the embedded clock with high precision.

The design goal of the DTS is to achieve an overall synchronization jitter below 5 ns in the detector electronics [110]. Tests performed on a demonstrator setup with a point-to-point optical connection have already shown promising results, with clock and time transfer measured at 12 ps RMS jitter [109].

4 Preliminary research

In the following chapter, the requirements of the CBM experiment and its DAQ system for synchronization and real-time control functionality are analyzed. Based on this analysis, a specification of a TFC system has been derived with quantitative goals for the accuracy of the time synchronization, constraints of the message transport functionality and the system scalability. Having the specification of system requirements and constraints is crucial for the systematic and justified identification of further design approaches and choices.

The chapter continues with the definition of high-level architectural design choices that can be analytically derived from the system specification and fundamental properties of distributed systems. The preliminary design choices are primarily focused on the network topology and the physical communication medium. Eventually, the analytically derived preliminary system concept leads to definition of the further design steps required to design the TFC system that meets the needs of the CBM experiment.

4.1 Requirements and system specification

Large-scale particle physics experiments present a set of interrelated challenges that can compromise data integrity and real-time control if not addressed systematically. First, the extremely short timescales of high-energy collisions demand precise clock alignment throughout the detector, or else signals from a single collision will appear mismatched across front-end modules. Second, communication delays, whether for data or control signals, must be predictable and bounded, because

even a small fluctuation can cause lost event fragments or delayed interventions in time-critical systems. Third, the constant expansion of the experimental setup, driven by higher collision rates and newly added detector layers, creates growing demands on both synchronization fidelity and throughput capacity.

These issues define three core requirements for a data acquisition system: synchronicity, bounded communication latency, and scalability. Each of these requirements has a direct influence on the others. For example, even if all modules are synchronized to a common clock, unpredictable transmission delays can hinder precise alignment of clocks and time-stamps. Similarly, preparing for incremental upgrades while maintaining strict timing guarantees is impossible without carefully designed synchronization and latency control methods that can be scaled or rearranged without compromising the performance.

Considering these interdependencies, there are certain benefits in implementing a dedicated synchronization and control system independent of the main data acquisition framework. By isolating the tasks of clock distribution, real-time control messaging, and system-level coordination, this standalone approach ensures that critical timing and control operations remain robust, even under the rising data loads of evolving detectors. Addressing the three requirements within one specialized architecture allows researchers to ensure precise timing accuracy, maintain bounded communication latency, and seamlessly integrate new detector components or reconfigure existing ones to meet evolving experimental demands. This separation avoids the added burden of handling high data throughput and performing online analysis, which could otherwise jeopardize system performance and add significant engineering challenge given the limited resources of FPGA devices. Ultimately, this methodology serves the central physics objectives of capturing collision events with high fidelity and managing the complex processes needed to operate a state-of-the-art particle physics experiment at scale. These requirements apply to the CBM experiment as well, and the functional requirements for the timing and fast control system are further discussed from this perspective.

The CBM setup is a large and distributed system, with numerous sensor channels and physically separated components. Its heterogeneous architecture requires a

scalable and adaptive solution for synchronization and control. Therefore, the timing and fast control system must be agnostic of the data readout architecture, which may vary from one detector subsystem to the other, and be inherently expandable to serve the number of the Common Readout Interface (CRI) boards that will be deployed in the first version of the experiment and potentially support further upgrades. In order to develop a system that is capable of achieving this complex task, three major requirements have been identified and discussed in detail below: synchronicity, latency determinism and scalability.

4.1.1 Time stability

The first requirement, synchronicity, arises because collisions take place on extremely short timescales, and each subsystem must accurately record the instant at which an interaction occurs. If two modules experience drift in their local clocks, or if cable length and temperature fluctuations cause timing shifts, the signals representing the same collision can become misaligned. This problem worsens in large or densely populated detectors, where each additional link is another possible source of delay. Without a master clock and thorough calibration routines, small discrepancies can assign different timestamps to signals from the same collision, weakening the consistency of the entire dataset. A dedicated synchronization and control system isolates these critical timing functions from the broader DAQ tasks, ensuring that the clock distribution and phase alignment are governed by rigorous procedures and specialized hardware.

The CBM experiment heavily relies on a shared time reference, since it uses time-based event reconstruction to correlate signals across different detectors. In this approach, data from each subsystem are labeled with precise timestamps rather than depending on a global hardware trigger. The first step occurs in the FEE, where each data packet receives a timestamp derived from a globally distributed time base. These timestamps need to be wide enough to cover the full experiment duration without ambiguity, which is challenging due to the strict hardware constraints in the FEE. Limited power, memory, and processing capabilities, combined with

the requirement for radiation hardness, make it difficult to implement both a high-resolution and a long-range timestamp. For that reason, FEE components assign a short timestamp instead, which is later expanded as part of the online processing in the CRI boards.

Thus, synchronization must be achieved at the CRI level, where data streams from multiple FEEs are merged and prepared for higher-level processing. As the FEE and the GBT layers receive clocks and operation mode controls from the CRI boards, global timing information must enter the DAQ system at this point for maintaining accurate event reconstruction across the experiment.

Based on the above analysis of the architecture of the CBM DAQ scheme, the TFC system must provide synchronization on the CRI level, while particular detector subsystems ensure correct operation of the FEE based on the common CRI time. This common CRI time, further referred to as TFC time, is defined by a clock signal and a digital timestamp. It is absolutely critical that the time offset between any two components in the system is stable even in the case of the full restart of the system, as it is not possible to recalibrate the offsets in the system during operation. This is caused by the combination of two factors: first is that once the experiment is deployed and accepts the particle beam from the accelerator, its recalibration becomes extremely difficult, and the second is that detector subsystems may have to be reinitialized during the experiment operation. If the time offset of a subsystem changes after its reinitialization, it becomes impossible to integrate it with the data from other subsystems during event reconstruction. The time offsets themselves are tolerable and do not have to be eliminated in the system, since various detectors inherently have different time offsets that are unrelated to readout electronics, and the data model of the event reconstruction algorithm in the FLES system is designed to compensate for such time offsets. The required time stability is related to the timing performance of the subsystems in the detector setup. For reasonable operation of the CBM detector, it is required that the time in its components does not vary larger than 200 ps RMS. This requirement covers deterministic behavior of time counters as well as stability of the clock phase.

The granularity of time in the system is defined in periods of the 40 MHz global base frequency, due to the architecture of the DAQ system. For this reason, the distributed TFC time must carry a global 40 MHz clock, since the TFC system is meant to be the only source of the common time in the experiment. As all time counters in the experiment are synchronous with this base clock, the local CRI time counters must be incremented at this base frequency and provide unique timestamps to the aggregated data over the lifetime of the experiment. This time is derived from the TFC time information, which means that the distributed TFC timestamps must be unambiguously defined in the clock domain of the 40 MHz TFC clock signal.

In addition to stability of time offsets in the system, it is imperative for robustness of the DAQ system that the CRI boards remain functional after a potential loss of connection with the TFC system. Interruption of the CRI clocks would prevent safe shutdown of the subsystem, which may lead to consequences as far as damaging the sensors or FEE components, as well as losing valuable diagnostic information that would otherwise be valuable to identify and address the cause of failure. Therefore, the TFC interface of the CRI boards must keep the local CRI clocks uninterrupted at all times, including loss of connection with the TFC system.

4.1.2 Message transport latency

The second requirement, deterministic communication latency, focuses on guaranteeing that data packets and control commands pass through the system within tightly bounded intervals. Often, real-time triggers rely heavily on bounded delays, because they must either accept or discard collision data quickly. Nevertheless, many other control and safety functions also demand predictable timing. Monitoring critical system parameters might need to trigger rapid interventions to maintain operational conditions, while certain power or configuration commands must arrive at a precise rate to avoid instabilities. If network paths introduce variable delays, buffers in the front-end electronics can overflow, critical updates may be issued too late, or safety responses may arrive well after a damaging

condition appears. A separate synchronization and control system can prioritize deterministic communication over purely data-intensive transactions, minimizing jitter and ensuring that each control event arrives on time.

Since timestamps are assigned early in the CBM data acquisition process, the data paths do not have strict latency requirements as variable data-transfer delays are tolerable as long as each event is correctly labeled. However, bounded latency remains essential for some control signals, which must arrive in predictable time intervals to maintain efficient operation of the data acquisition system. The most critical problem that requires latency-bounded control is the variation of the data flow through the system. This variation stems from the nature of the slow beam extraction from the accelerator, which is required for the physics goals of the experiment. With slow extraction, the intensity of the beam is not stable over time, which is expected to occasionally cause overflow of the DAQ system [33, 111]. Such overflows inevitably corrupt the data related to physical events captured by the detector. If uncontrolled, data corruption caused by buffer overflow due to intensity surges is unpredictable and may lead to corruption of an unnecessarily large number of events, thus significantly reducing the efficiency of the entire CBM detector. It has been previously demonstrated in simulations that interrupting the data flow through the system in an organized fashion preserves a significant number of correctly collected events not only during the data surges that overshoot the bandwidth of the DAQ system, but also below the readout bandwidth limit [34]. Therefore, timely control of the data flow prevents data loss in the experiment and improves efficiency of the detector.

To provide the fast control functionality, the TFC system must be able to collect status information from all CRI boards in the system and broadcast fast control commands to them. The CRI boards, in turn, propagate the fast control commands to the corresponding FEE components. Given the timing performance of control operations in the FEE, the fast control commands must arrive to the CRI boards with low delay in the order of microseconds. The throttling simulations have been carried out considering the delay of 6 microseconds between status generation in the FEE ASIC and arrival of the corresponding fast control command to the same ASIC. From this latency budget, the latency bound of 3 microseconds can be

safely assumed for the transport delay between CRI and the fast control processor and back. For clarity, the direction of communication from CRI boards to the TFC system is further referred to as "upstream", and the reverse direction, from the TFC system to CRI boards, is called the "downstream" direction.

Aside from the aforementioned requirement for low latency of fast control message transport, the messages traveling downstream must also have deterministic latency. This means that the latency is not only bound in its absolute value, but its variance must be strictly limited as well. This is caused by the fact that fast control messages, be that throttling commands or, for example, a command for a data readout reset, must reach all CRI boards at the same time. This keeps the system coordinated and reduces system dead time while executing fast control commands. The downstream latency variance bound can be defined as equivalent to the synchronization accuracy, which equals 200 ps RMS.

4.1.3 Scalability

The third requirement, scalability, is driven by the evolving nature of accelerators and detectors. As collision rates rise or as detector subsystems are added, data streams increase accordingly. A standalone synchronization and control infrastructure must therefore accommodate growing throughput and expanding networks without sacrificing precise timing or bounded latency. If the core architecture is not designed to handle these changes, the system can experience degradation in timing accuracy or encounter new bottlenecks that undermine future enhancements. Maintaining a clear boundary between the main DAQ system and the TFC functions helps to focus on efficient, low-latency command routes and reliable clock distribution when new readout modules or fibers are introduced. This approach also simplifies long-term upgrades, because adding or reconfiguring detector components becomes primarily a matter of integrating them into a pre-existing, rigorously managed control network.

It is important in the CBM experiment to have an option of expanding the range of fast control commands as the experiment evolves. As the fast control functionality

is combined with time distribution in the TFC system, extension of the fast control functionality must not hinder the performance of system synchronization. For that reason, redundant bandwidth of the transport links is required to provide the necessary margin for future upgrades. Additionally, the format of fast control commands must be sufficiently generic to ensure the flexibility to allow for future complex control.

In the first version of the CBM detector, its DAQ system will comprise about 250 CRI boards. This property defined the required scale of the TFC system, which must serve at least 250 endpoints. Nevertheless, it is additionally important to know how the system performance degrades with topology changes and expansion beyond the limit of 250 endpoints. Therefore, characterization of errors added by system components and topological features is required to make the architecture expandable and adaptive.

4.1.4 Summary

4.2 Initial design framework

Although feasibility study is necessary to identify the appropriate approach to developing the TFC system and fulfilling the requirements of the CBM experiment, some of the architectural design choices can already be made at the initial development stage, based on the state of the art and the requirement analysis. For example, certain common design patterns can be observed in the state-of-the-art solutions. As the requirements of the state-of-the-art works largely overlap with requirements of the TFC system, these architecture design patterns can be potentially used as the design framework for the system development.

The most prominent commonalities in existing solutions for real-time systems are the tree network topology and the use of optical fiber as a medium in applications where relatively high data bandwidth is required. These two properties are analyzed in more detail below, to assess whether they can be reused to form the basis of the

system under development, before the further studies are carried out to devise the details of the TFC architecture.

4.2.1 Network topology

Topology of the distributed system is one of the key system parameters that should be defined early in the development process. It is already defined in the requirements for the TFC system that the topology must be centralized with one controller serving multiple endpoints, and the round-trip communication latency is a critical parameter. Therefore, the topologies are further analyzed with the focus on these parameters.

Generally, distributed real-time architectures have one of the following topologies or a combination thereof: bus, ring, tree, star or mesh [112]. In the bus topology, components are physically connected to the same bus and share it for communication. As no component has an inherent advantage in the communication over the bus, an arbitration mechanism is required to prioritize the access. Communication latency in such systems is dependent on the arbitration mechanism and the occupancy of the bus: a node may have to wait for all the higher-priority communication to finish before it starts sending data itself. Due to the reduced wiring, bus systems are commonly used in space-restricted real-time applications, e.g. automotive or avionic control systems. The shared nature of the bus limits its scalability, as adding more nodes increases the contention for access, which can lead to waiting times to send data and reduced overall throughput.

Ring topology implies that components are arranged in a closed loop, with each node connected directly to its immediate neighbors. Communication occurs in one or both directions around the ring, depending on the specific implementation. As data must pass through intermediate nodes to reach its destination, each node plays a dual role as both a transmitter and a relay. The topology requires a coordination mechanism to avoid data collisions and ensure orderly transmission, typically through token-based or time-division schemes. Like in the bus topology, the wiring requirements are reduced compared to fully connected networks, scalability

is constrained by the cumulative propagation delay and the increasing time needed to travel through the ring as more nodes are added. Ring topologies are commonly employed in scenarios where fault tolerance and deterministic communication are critical, such as in traffic signal control systems or factory automation networks.

In a star topology, each node is directly connected to a central node, which acts as the primary hub for data transmission. Communication is routed through this central device, allowing it to manage traffic and isolate failures to individual links. As a consequence, communication latency in star configurations is generally low and predictable, provided the hub can effectively handle multiple data streams without becoming a bottleneck. This can simplify maintenance and fault detection, since an issue with one connection does not disrupt the rest of the system. However, the hub introduces a single point of failure, and the wiring effort increases as each node requires its own dedicated link to the central device. Star configurations are widely used in local area networks and smaller-scale real-time systems where predictable troubleshooting and centralized control are advantageous.

In the tree topology, nodes are arranged in hierarchical layers branching out from a root node. This structure effectively combines multiple star networks in a layered format, which can help organize large systems by grouping subsets of nodes under intermediate controllers or switches. Traffic typically flows upward and downward along the branches, and the top-level nodes may coordinate overall network activity. Due to this layered structure, communication latency can accumulate as messages travel multiple hops, particularly if congestion occurs in higher-level nodes. Although it can be more scalable than a single bus or ring, the tree topology is sensitive to failures higher in the hierarchy, which may isolate entire sub-trees. Tree configurations are commonly applied in industrial sensor networks and extensive real-time control systems that benefit from hierarchical management and segmentation.

In a mesh topology, each node has the ability to connect with multiple other nodes, forming a web of interlinked paths. This redundancy enhances fault tolerance, as a failure in one link or node may be circumvented by an alternative route. Mesh networks can also reduce transmission delays through parallel paths, especially in

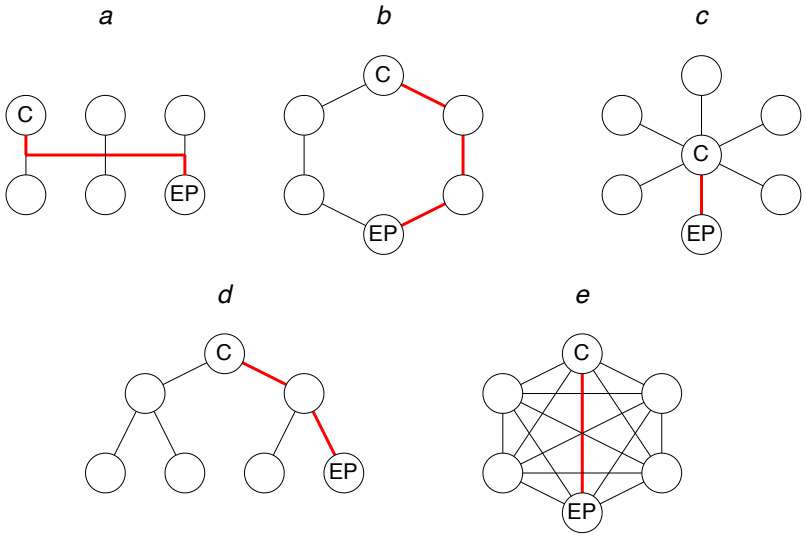


Figure 4.1: Typical topologies of real-time systems: a - bus, b - ring, c - star, d - tree, e - mesh. One controller and one endpoint are placed at the worst-case distances with a route between them highlighted in red.

time-critical scenarios. Nevertheless, the added routing complexity can introduce overhead that affects communication latency, as the network must determine which path is optimal. However, their complexity and wiring costs escalate significantly as the number of nodes grows, and sophisticated routing protocols are often necessary to manage data flows. Mesh layouts are favored in mission-critical applications, such as military or emergency response systems, where continuous operation and robust connectivity are crucial.

All the topologies described above are presented in Figure 4.1. They are assumed to have one controller node and at least one endpoint node, to match the target architecture. The nodes are placed as far apart as possible in the topology, to analyze the worst-case latency scenario in each of them. The highlighted connections indicate the route between the controller node and the most distant endpoint.

Based on Figure 4.1, the ring topology can clearly be ruled out for a latency-critical system, as the data between the controller and the most distant endpoint has to travel the greatest number of hops in this scenario. Moreover, the number of hops increases linearly as more nodes added to a ring network. On the other hand, three other topologies stand out in terms of the number of hops between the controller and the endpoint: star, mesh and bus topologies always have one hop between these nodes. Although all three topologies seem to yield optimal communication latency between the controller and the endpoint nodes, they all have significantly limited scalability.

The star topology requires the controller to have connection ports with all other nodes, with the number of ports in the controller growing linearly with the number of nodes in the system. This poses a significant challenge in the systems of the scale required by the application targeted in this work. The mesh topology, in turn, requires extensive connectivity between all nodes, with the number of links in the system growing exponentially with each added node. While it may be technically feasible for smaller systems, the number of required connections grows to an overwhelming magnitude in large-scale systems. Bus systems require only one connection port in each node and the connectivity approach allows for adding a large number of nodes to the network without notable technical challenges. Nevertheless, with all nodes sharing the same bus, communication latency degrades as the network grows. Even with latency-optimized access management techniques, such as TDMA or token circulation, considerable latency can accumulate in large-scale systems where, like in the intended application, all endpoints must be polled for their current status. This latency accumulation negates the advantage of the one-hop access by the controller to all nodes.

Although it does not offer the shortest route between the controller and the furthest endpoint node, the tree topology appears to be the most balanced as it comes to the scalability drawbacks of the bus, star and mesh topologies. Similarly to the star topology, it is inherently centered around the so-called root node, which acts as the network controller. This central node is connected to a number of nodes which are, in turn, connected to another layer of nodes. This hierarchical nature of the network does not pose requirements to the number of links available in

every single node as, for example, the mesh topology. Based on the number of connections available in the nodes, the tree network can be adapted by adding or removing the hierarchy layers. At the same time, latency is only increased with each added layer to the network, and every next layer allows a larger number of nodes added to the network without latency overhead. Thus, the tree topology allows to trade off a small amount of latency for significantly improved scalability, which is why this topology is best suited for implementing the TFC system for the CBM experiment. This choice is also supported by the fact that the vast majority of similar latency-oriented scalable systems analyzed earlier in the state-of-the-art study use the tree network structure as well.

4.2.2 Communication medium

Given the known nature of the information flow required by the application and the operational environment, the communication medium can be defined early in the system design process as well. Being available in a vast variety, communication media can be generally classified into guided and unguided as presented in Figure 4.2 [113]. Regardless of the technology used, guided media transfer information over a conduit which physically contains the carried signal. These media are also known as wired. Unguided, or wireless, media, on the other hand, carry information directly through the environment, e.g. air or water, making the information available to any client within the transmission range.

Being inherently different, guided and unguided media have their unique sets of advantages and disadvantage, and one or the other is typically better suited for each particular application. Guided media generally offer high communication bandwidth, low and stable transmission latency, and are highly immune to interference from the ambience, as signal integrity in these media is physically protected by the boundaries of the media themselves. This reliability and performance comes at a cost of the maintenance overhead, as signal wires must be laid out between network nodes and protected from physical damage. Due to their advantages, wired communication media are commonly used in applications, where large

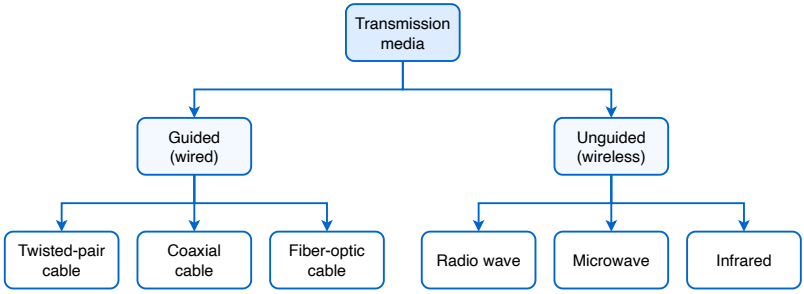


Figure 4.2: Commonly used transmission media classes.

amounts of data must be transferred reliably, such as office networks and the internet, high-resolution television. The high degree of data transfer reliability has earned wired media profound popularity in in-vehicle networks as well.

Two types of guided media are commonly used nowadays: those based on copper wires and on optical fibers. Copper-based data transmission media include various coaxial and twisted pair cable configurations. Coaxial cables are built with a copper core that is used as a conduit for the electrical signal that carries information and a shield that protects the signal from external electromagnetic interference. Another copper-based medium, twisted pair, is built with two intertwined wires and offers lower signal attenuation than coaxial cables and, therefore, better signal integrity over long distances at the cost of the frequency range [113]. This makes the twisted pair more suitable than coaxial cables for transferring high data rates between remote locations, which is why twisted-pair cables have been the medium of choice for the vast majority of local area networks.

Leveraging a fundamentally different approach to transmitting signals, optical fibers offer unprecedented bandwidth and superior latency performance compared to copper-based media [114]. However, since an extremely thin fiber made or glass or plastic is used to carry the signal in the form of light, optical-fiber cables are more fragile than coaxial or twisted-pair cables and require considerable mechanical protection and handling care. As bandwidth as well as low and

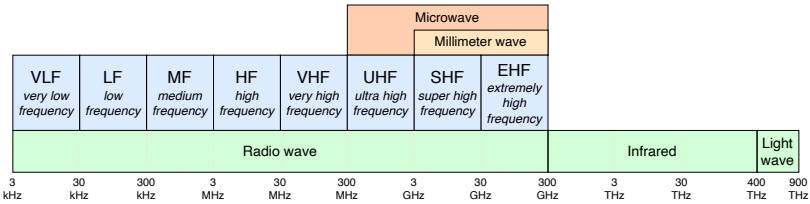


Figure 4.3: Classification of wireless signals by frequency ranges.

deterministic latency are key requirements for the TFC system, optical fiber is the best-suited of the available guided media to use in the system.

In contrast to the guided media, unguided media transmit signals directly through the environment without using wires or any other conductors. Also called wireless, these media make the signal available to any number of receivers within the transmission range, which grants wireless media the unique scalability properties and enables dense star and mesh topologies without a significant technical overhead. Generally, the wireless transmission range is directly proportional to the output power of the transmitter [113]. At the same time, wireless networks are prone to interference from other wireless transmission sources and environmental conditions, which may impair their real-time performance [115]. Nonetheless, due to the ease of scalability, unguided media have been widely used in various broadcasting networks, most prominently TV- and radio broadcasting. Besides, with no need of wires for signal transmission, these media are well-suited for communication with moving object, like cars or robots, and are the only available media for airborne objects, such as planes, spacecrafts and unmanned aerial vehicles.

When discussing wireless transmission media, it is important to note that their properties vary depending on the used frequency range. Figure 4.3 shows the various frequency ranges, or bands, of wireless signals. Signal transmission with lower frequency requires larger antennas, has lower signal attenuation with distance and easily penetrates obstacles, which makes them highly suitable for long-distance communication and broadcasting, such as long-distance navigation in oceans and air [113]. Whereas lower-frequency antennas transmit the signal in all directions, transmission frequencies of the microwave range and above make it possible to use

more compact and focused antennas. Due to this property, higher frequencies can be used for one-to-one radio communication with a minimal interference between signals from different sources. In addition, the higher frequency ranges allow for wider radio bands, thus increasing the channel data rate. The major downside of higher-frequency wireless signals is that they cannot penetrate obstacles as easily as lower-frequency signals, which limits the indoor use of radio communication with higher frequencies.

Of all the wireless frequency ranges, the microwave range stands out not only due to the available bandwidth, but also because of the latency stability, which makes it suitable for real-time applications. Although certainly applicable to synchronous distributed systems, currently available wireless communication technologies in the lower microwave range cannot reach nanosecond stability required in this work [116]. The more promising results have been achieved with the millimeter waves, or the higher microwave frequency range, with the latency stability being notably close to that with the optical fiber communication [117]. Nevertheless, the issue of repeatability and susceptibility to interference and environmental conditions remains open and has yet to be studied.

The choice of the medium to build the TFC system in the CBM experiment is driven mainly by the requirement of sub-nanosecond stability of data transmission. As optical fiber offers superior bandwidth and latency performance, both of which are beneficial for achieving the scientific goals of the experiment, this medium is the most suitable to carry time information and fast control commands in the TFC system. This choice is additionally supported by the dominance of optical fiber connections in the state-of-the-art solutions.

4.2.3 Preliminary concept

Considering the analysis of the requirements, potential network topologies and communication media, a preliminary concept of the TFC system has been devised. The preliminary concept defines the fundamental architectural design choices and constitutes a foundation for further development of the system.

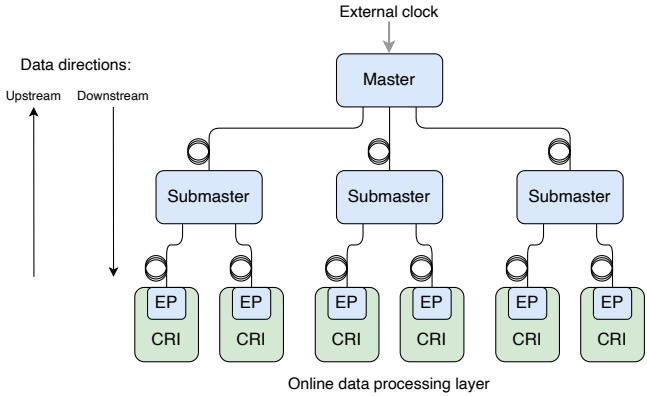


Figure 4.4: General architecture of the TFC system.

Although the physical connections within the TFC system are point-to-point links between pairs of devices, the overall network structure forms a hierarchical tree topology. In this architecture, each node at an intermediate level serves not only as an endpoint for its direct communication link but also as a forwarding agent for traffic between higher and lower layers. Thus, while every cable connects exactly two devices, data often travels multiple point-to-point links through intermediate nodes to reach its destination. This hierarchical organization enables the system to scale efficiently to hundreds of nodes without requiring an impractical number of direct connections to the central timing master. Eventually, this hierarchical tree topology, depicted in Figure 4.4, ensures latency-efficient communication between the 200 endpoints and the central controller.

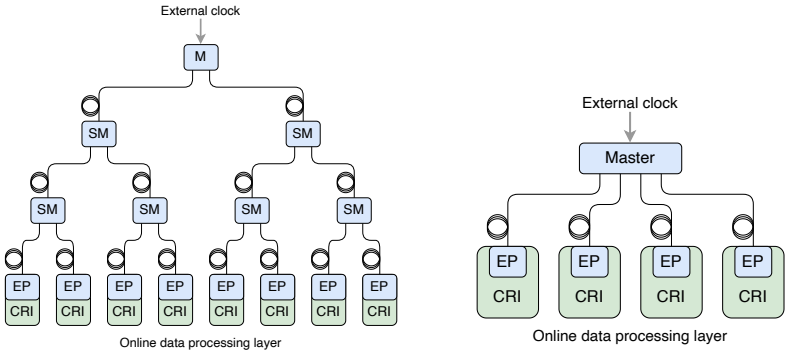
The TFC architecture defines three types of FPGA nodes: Master, Submaster and Endpoint. All nodes in the TFC system are hierarchically interconnected with bidirectional multigigabit optical links, which carry clock, time information, and fast control traffic over the same links. In the context of each TFC node, connections in the direction of the Master are called upstream connections, or uplinks, and all connections in the direction of Endpoints are called downstream connections, or downlinks. Thus, for example, only Master and Submaster nodes have downstream connections and only Submaster and Endpoint nodes have

upstream connections. Similarly, the data transport from Endpoints to the Master node is considered to travel in the upstream direction and the data being broadcast from Master towards Endpoints is said to be transmitted in the downstream direction.

The Master node is the central control node of the TFC system. It receives an external reference frequency and defines the system wide clock rate and time in the form of an electrical clock signal and a time counter value, respectively. Both the rate and time are distributed to the rest of the system. In addition, the TFC Master collects status information from all endpoints, processes it and issues fast control commands when needed.

The Master node is connected with a number of downstream links to TFC Submaster nodes. The purpose of the TFC Submaster nodes is to form the hierarchy of the TFC system and thus guarantee its scalability. Each Submaster node receives the clock and time information and the fast control commands from the TFC Master on its upstream link and distribute them via downstream links to endpoints. At the same time, this node receives Endpoint status information from all its downstream links, aggregates that information and sends upstream in the direction of the Master node. Notably, layers of Submaster nodes can be stacked, thus ensuring the architecture highly scalable and adaptable.

While all TFC Master and Submaster nodes in the TFC architecture are based on individual FPGA boards, TFC Endpoints are meant to be embedded into the CRI designs as an Intellectual Property (IP) core. It acts as an interface between CRI boards and the TFC system functionality and is responsible for serving one upstream link connected to a corresponding Submaster node or, as an option, directly to a TFC Master. The Endpoint receives global clock and time from an upstream TFC link, as well as provides the status information of the CRI board to the TFC system and receives fast control commands. Since the TFC Endpoint is the primary source of the clock and time of each CRI board, the Endpoint continues to produce valid output even in case of a TFC link failure, to enable failure diagnostics or a safe shutdown of the DAQ system.



(a) Configuration with two layers of Submasters.

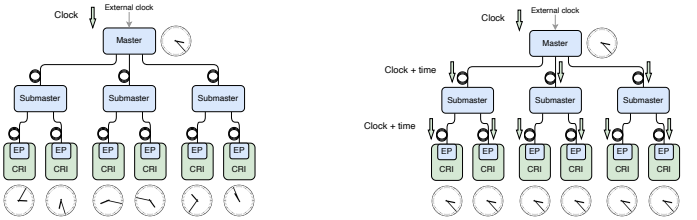
(b) Configuration with a direct connection between Master and Endpoints.

Figure 4.5: Alternative configurations of the TFC architecture.

TFC system architecture is designed to be easily scalable and adaptable. Therefore, the number of submaster layers can vary to adapt the system configuration depending on the needs of the use case. Layers of Submasters can be stacked to scale the system for a greater number of supported Endpoints or to adapt to various numbers of available downlinks in each specific hardware platform, as shown in Figure 4.5a. Similarly, if the Master node has a sufficient number of downlinks to connect to all Endpoints in the system directly, Submaster nodes can be omitted overall (Figure 4.5b). Assuming an even depth of the TFC network, where all Endpoints have an equal number of hops to the Master, and equivalent number of downlinks N_{dl} in all Master and Submaster nodes, the total required number of hops N_{hops} in the network to serve N_{EP} Endpoints can be calculated using equation 4.1.

$$N_{hops} = \lceil \log_{N_{dl}} N_{EP} \rceil \tag{4.1}$$

Directly after startup, time counters in all CRI boards have arbitrary values. To guarantee a common notion of time in the entire TFC system, the system-wide time originates in the Master node, regardless of the presence of the external reference,

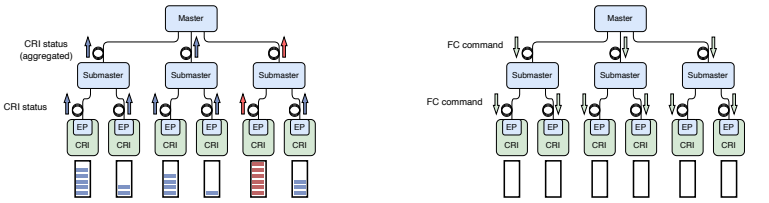


(a) Local clocks before synchronization. **(b)** Local clocks after synchronization.

Figure 4.6: Synchronization of local node clocks in the TFC system.

and is applied to the entire system. During the system initialization, clock and time are simultaneously transmitted from the Master node to Submasters over all downlinks available in the TFC Master. Once the Submasters have successfully received the complete time information, they propagate it further to the Endpoints. In this case, the time information is propagated to further Submaster layers in the cascaded fashion until it reaches the Endpoints. Once the Endpoint received the time information from the Submasters, they apply it to their local clock generators and time counters, after which the system is considered initialized. The time initialization procedure is depicted in Figure 4.6. The time information is continuously broadcasted to the Endpoints during the normal operation of the system, to verify that the local time flow stays stable over extended periods of time. If, as a result of failure, the local time of a TFC Endpoint is no longer consistent with the global TFC time, the problem is immediately reported to the user, so that the cause of the failure can be investigated. Nevertheless, local clock and time counter, although no longer consistent with the TFC time, remain uninterrupted to guarantee functional stability of the CRI board.

Fast control commands in the TFC systems are issued by the TFC Master based on the momentary status of the system. In order to enable the centralized control of the system, status information about all CRI boards is transmitted via TFC Endpoints to corresponding Submaster nodes. These nodes aggregate this status information received from the downstream links and forward it upstream towards the TFC Master either via direct upstream connections (Figure 4.7a) or via cascaded layers of Submasters, depending on the specific configuration of the



(a) Status collection from TFC Endpoints. **(b)** Distribution of a fast control command.

Figure 4.7: Demonstration of a fast control cycle using an example of clearing First-In, First-Outs (FIFOs) in case of local overflow.

TFC System. Having received the status information from all TFC Endpoints in the system, customizable fast control logic in the TFC Master processes it and issues a fast control command if needed. The fast control logic in the Master node is adaptive by design and the fast control commands can be easily extended and adjusted in the future according to the needs of the experiment. Once a fast control command is issued, it is transmitted to the Submaster nodes via the downstream links. Similarly to the time distribution, the Submaster nodes forward the fast control command in a timely fashion to the Endpoints (Figure 4.7b), where the command is applied to each corresponding CRI board at a predictable point in time.

4.3 Feasibility study

Synchronization is a core feature of the TFC system and its concept must be carefully devised to meet the stringent requirements of the experiment. Two major fundamental approaches to network synchronization can be distinguished from the state of the art. In the first approach, slave nodes adjust their own time flow based on a mutual exchange of time information with a master node. This approach can be implemented using general-purpose communication links with variable latency and synchronization accuracy is then limited with the link delay symmetry and accuracy of time shift measurements in each node individually.

With the other approach, the time from the master node is simply transmitted to slave nodes and applied by the nodes directly as local time. Although the synchronization procedure in this case is very straightforward, accurate synchronization depends on communication links with fixed latency. Additionally, the clock must be communicated to the slave node, so that time rate is consistent in the network. Thus, synchronization accuracy in the system is directly connected with the latency determinism of the links over which time is distributed.

Both of the above approaches have been widely adopted and demonstrated similarly significant synchronization accuracy, and analysis of the state of the art does not reveal any clear advantage of using either of them in the TFC system. For this reason, additional feasibility study is required to closely investigate each of the two approaches and analyze its synchronization capabilities in combination with other requirements and constraints of the TFC system. For the feasibility study, a prototype of a system with each synchronization approach has been developed building up on state-of-the-art techniques. Based on these prototypes, design constraints and fundamental limits of each synchronization approach have been explored in the context of the demands of the experiment in order to select one that is better suited for application in the TFC system.

5 Approach 1: synchronization through time self-adjustment

In this synchronization approach, network nodes adjust their local time based on timing information received from the time master and link delay measurements. To achieve synchronization, timestamping of messages exchanged between the master and slave is required. This approach is inspired by the Christian's algorithm and its derivatives that extend it with advanced timestamping techniques and link delay models. The fundamental principle of this method for achieving synchronization between nodes is discussed below.

This synchronization method has no strict requirement for the parameters of the hardware or communication links and can be implemented using general-purpose networking interfaces. However, optimizations of the delay model and timestamping techniques, statistical error mitigation and link delay asymmetry compensation may significantly improve performance and robustness of synchronization.

To study applicability of the synchronization method based on time self-adjustment for the TFC system in the CBM experiment, a minimal prototype was developed. The purpose of the prototype is to implement the features that are most critical to the DAQ system of the experiment and verify that the methods delivers sufficient synchronization accuracy in combination with a low-latency transport channel for time-critical fast control messages. Additionally, the prototype is meant to assess the compatibility of the approach with the hardware constraints of the experiment and explore its practical limitations.

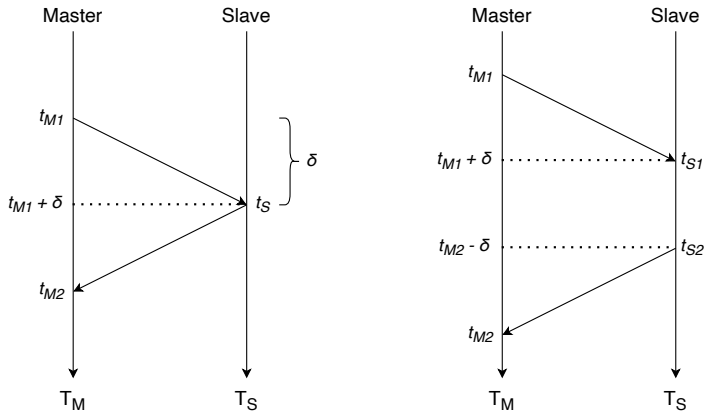
5.1 Time synchronization concept

In the self-adjusting synchronization method, timing slaves derive the amount of their own time offset from samples of round-trip communication of timestamped messages with the timing master (Figure 5.1). In the basic implementation of the method shown in Figure 5.1a, the master node sends its time t_{M1} to the slave that has an unknown time offset relative to the master time ($T_S = T_M + T_{offset}$). At the point t_S when the slave node receives the message, the master time contained in it is outdated due to the propagation delay δ of the communication link and cannot be directly compared with the local time t_S for time adjustment. In order to find out the propagation delay, the slave node sends a reply message to the master. As the master receives the reply, it sends the time t_{M2} of reply reception back to the slave. After this, the slave node has all the timestamps t_{M1}, t_{M2} and t_S. Assuming that the communication link has a symmetric propagation delay and the slave responds to the message immediately, the one-way propagation delay can be expressed with Equation 5.1 and the time offset between the slave node and the master node would be defined by Equation 5.2.

$$\delta = \frac{t_{M2} - t_{M1}}{2} \tag{5.1}$$

$$T_{offset} = t_S - (t_{M1} + \delta) = t_S - \frac{t_{M1} + t_{M2}}{2} \tag{5.2}$$

In practice, however, it is rare that nodes can reply immediately. To allow for the delays due to local scheduling, queueing or buffering taking place, the response delay must be taken into account while calculating the time offset of the slave node (Figure 5.1b). Therefore, a slightly more complex computation is needed to derive the time offset. Assuming a symmetric propagation delay in the communication link as in the simpler case above, time relation between the master and the slave nodes can be expressed in two equations:



(a) A simple delay model. **(b)** A more complex delay model with delayed response.

Figure 5.1: Link synchronization between master and slave nodes.

$$\begin{aligned} t_{S1} &= t_{M1} + T_{offset} + \delta \\ t_{S2} &= t_{M2} + T_{offset} - \delta \end{aligned} \tag{5.3}$$

Solving equations 5.3 as a system for T_{offset} lets us calculate the offset completely independently of the variable response delay in the slave node:

$$T_{offset} = \frac{t_{S1} - t_{M1} - t_{M2} + t_{S2}}{2} \tag{5.4}$$

Thus, having these four timestamps, the slave node can calculate its own time offset and apply the necessary correction to its local time.

The concept of time synchronization in the prototype employs an enhanced link delay model inspired by the White Rabbit project [118]. The enhanced model includes high-precision link delay measurement and allows to achieve sub-nanosecond synchronization accuracy, which is essential in the CBM experiment.

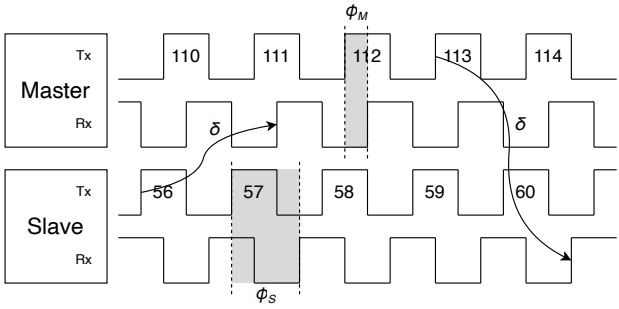


Figure 5.2: Clock relation between Master and Slave clocks.

The high-precision delay measurement helps to not only estimate the round-trip delay based on the time counts that are synchronous with the local clock, but to take into account delays on the finer, sub-nanosecond scale. This is achieved by measuring the phase shifts ϕ_M and ϕ_S inside the FPGA between the receiver clock and the local clock in each node (Figure 5.2). The measured phase shift serves as an extension of the timestamps used for synchronization to the sub-nanosecond range, and the one-way link propagation delay can then be calculated using Equation 5.5 [118].

$$\delta = \frac{((t_{M2} + \phi_M) - t_{M1}) - (t_{S2} - (t_{S1} + \phi_S))}{2} \tag{5.5}$$

Before delving into the details, it is important to note that the local clock and the Tx clock of each node are assumed to be synchronous without any phase shift. Thus, the Rx and Tx datapath delays are ignored for the feasibility study, and the accurate phase measurement only addresses the fine sub-clock component of the link delay. This is done to keep the prototype simple and only demonstrate the feasibility to compensate for fine phase shifts in the system. After the purpose of demonstration is fulfilled, extension of the phase measurement technique to other fine delays in the system is considered trivial in this study. For the same reason, contribution of link asymmetry to clock offset is considered negligible in the current work.

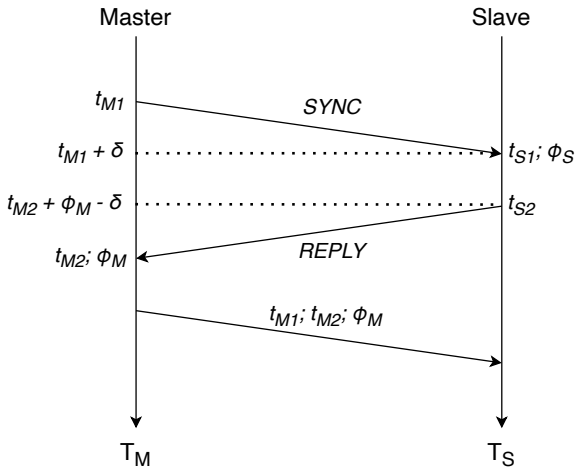


Figure 5.3: The synchronization procedure with the enhanced timestamping.

The enhanced synchronization procedure is shown in Figure 5.3. The synchronization procedure begins the same way as in Figure 5.1b, with the Master node sending a synchronization request SYNC to the Slave, recording the time of sending the message t_{M1}. No need to extend the timestamp at this point due to the assumption of a zero phase shift between the local clock and the Tx clock. Upon reception of the SYNC message, the slave records the value of its local time counter t_{S1} along with the phase shift value ϕ_S between its local clock and the receiver clock. With an uncertain and irrelevant delay, the slave replies to the Master and records the time t_{S2} as the message is sent. Similarly to the slave, the master node records the local time t_{M2} together with the phase shift ϕ_M when the REPLY message arrives. In the end, the master node transfers the recorded timestamps t_{M1}, t_{M2} and ϕ_M to the slave. The slave can then derive its time offset according to Equation 5.6 [118], which, after taking into account Equation 5.5 is transformed into Equation 5.7.

$$T_{offset} = t_{M1} - (t_{S1} + \phi_S) - \delta \tag{5.6}$$

$$T_{offset} = \frac{t_{M1} - (t_{S1} + \phi_S) + ((t_{M2} + \phi_M) - t_{S2})}{2} \tag{5.7}$$

On its own, the measurement of the aforementioned clock shift cannot help significantly for better node synchronization if the nodes use independent free-running clock sources, even if they are rated for the same frequency. The frequency offset within the manufacturing tolerance would cause a clock phase drift in the nodes, leading to a constant change of the phase shift value. This makes it impossible to establish a direct time relation between the Master and the Slave nodes with the desired accuracy. Therefore, the frequency between nodes must be aligned before the synchronization takes place to achieve the sub-nanosecond accuracy. Frequency alignment, or syntonization, removes the frequency skew and, subsequently, the constant time drift between nodes, which keeps phase shifts ϕ_M and ϕ_S constant.

Thus, to achieve the necessary synchronization accuracy, the synchronization procedure must be complemented with dedicated hardware techniques to ensure precision clock phase and frequency alignment. Building up on the previously proposed concept of the timing system for the CBM experiment, this is handled by an all-digital PLL with a clock generator and a software implementation of loop control [119]. The details of the used techniques are discussed in the sections below.

5.2 DDMTD

The key component that enables sub-nanosecond accuracy in the exchange-based time correction synchronization approach, is the DDMTD module [120]. Inspired by a technique to measure fine signal phase shifts in the analog domain, this component allows to measure clock shifts on the sub-clock scale without using higher-frequency signals or analog components. The measurement accuracy is limited mainly with the jitter of signals in the system, which can be mitigated using averaging or other statistical methods.

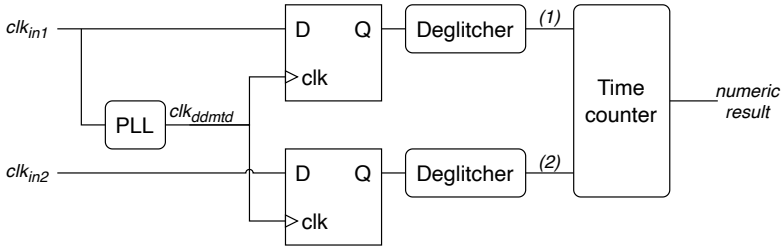


Figure 5.4: Schematic diagram of a DDMTD.

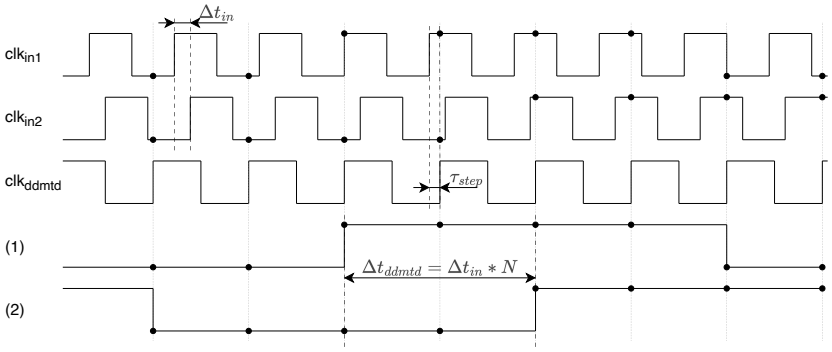


Figure 5.5: Clock and phase stretching in a DDMTD for $N = 8$.

Leveraging the caliper principle, this technique measures a fine phase shift between two syntonous clock signals by sampling them with a third clock signal with a slightly shifted frequency 5.4. Clocked by a slightly slower clock, the D flip-flops capture different points in the periods of the corresponding clocks, thus effectively stretching the clocks while preserving the initial shift as shown in Figure 5.5. As the flip-flops capture the input clocks near their edges, stretched output signals are distorted with glitches due to the jitter in all three clocks. For that reason, deglitching is necessary on these signals before the phase shift can be measured between the stretched signals with a simple counter.

The most important property of the DDMTD is its stretching factor N. This number defines the precision of phase measurement and how long it takes to

capture each of them. As the period of the third DDMTD clock τ_{ddmtd} is slightly different from the periods of measured clocks, its phase drifts relative to the measured clocks with regular and known intervals τ_{step}, which depends on the period of any of the input clocks as defined in Equation 5.8. Thus, the DDMTD flip-flops effectively scan the value of each measured clock across its period as indicated with dots in Figure 5.5. The complete scan takes exactly N clock cycles of clk_{ddmtd}, which defines N in terms of clock periods as shown in Equation 5.9.

$$\tau_{step} = |\tau_{clk_{ddmtd}} - \tau_{clk_{in}}| \tag{5.8}$$

$$N = \frac{\tau_{in}}{\tau_{step}} = \frac{\tau_{in}}{|\tau_{clk_{ddmtd}} - \tau_{clk_{in}}|} \tag{5.9}$$

Generally, smaller values of τ_{step} increase measurement precision, but they also increase the time that it takes to obtain each numeric value from the DDMTD. Thus, the precision of DDMTD measurements is traded off for the measurement time τ_{beat} that is expressed as follows:

$$\tau_{beat} = N \times \tau_{ddmtd} = \frac{\tau_{in} \times \tau_{ddmtd}}{|\tau_{ddmtd} - \tau_{in}|} \tag{5.10}$$

The fundamental prerequisite for the technique is the slight frequency shift of clk_{ddmtd} relative to the measured clocks, either slightly faster or slightly slower. As can be seen in equations above, the direction of the frequency skew does not affect τ_{beat} or N. However, it changes the direction, in which the periods of the measured clocks are scanned by the flip-flops with intervals τ_{step}. The slower clk_{ddmtd} will preserve the direction of the phase shift, while the faster clock will scan the input clocks in the reverse direction, effectively inverting the phase shift at the output, which must be taken into account while interpreting the output value of a DDMTD. As the clocks are stretched by the flip-flops by the factor N and the phase shift between them is counted in term of clk_{ddmtd} periods, the initial

time shift between input clocks can be derived from the integer DDMTD output δt_{ddmtd} as follows:

$$\Delta t_{in} = \Delta t_{ddmtd} \times (\tau_{ddmtd} - \tau_{in}) \tag{5.11}$$

When used as a phase measurement component in a system, a DDMTD must be configured based on the system constraints of the measurement time τ_{beat}, measurement precision defined by N and the minimum achievable frequency skew with the platform PLLs. These factors have been carefully considered while developing the prototype for the feasibility study to enable the management of fine clock offsets in a distributed system.

5.3 Software PLL

As the key prerequisite for the enhanced exchange-based synchronization described in Section 5.1, syntonization must first be achieved between the master and the endpoint link. Propagated through each link individually, common frequency is established on all nodes in the system.

To align its local clock frequency with the frequency of the data received from the master node, all submaster and endpoint nodes in the system require a PLL. However, using commonly found high-performance hardware PLLs significantly increases dependency of the architecture on specific hardware. Besides, hardware PLLs typically have only limited flexibility in terms of dynamic reconfiguration or phase tuning. To overcome the limitation of the hardware PLLs and decrease the dependency of the system on specific hardware components, a software PLL (SPLL) approach has proven effective [120] and has been adopted for this feasibility study.

Taking advantage of fine clock phase measurements with a DDMTD, SPLL replaces the typical analog circuitry of a conventional PLL with all-digital components to control a local frequency generator as presented in Figure 5.6. As the loop

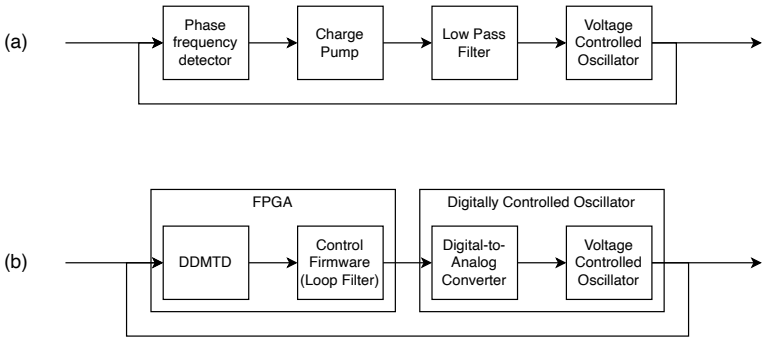


Figure 5.6: Hardware- (a) and software- (b) PLL architectures

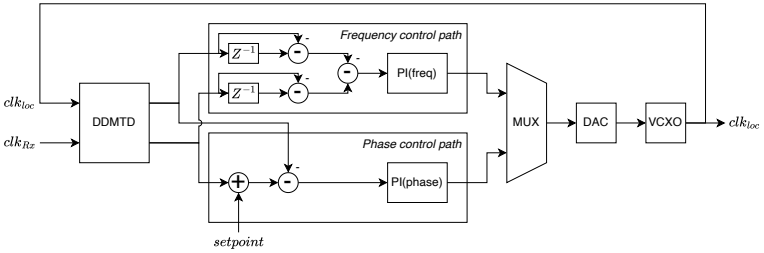


Figure 5.7: Two-phase PI phase and frequency control in White Rabbit.

control is handled in software, the SPLL architecture allows to implement complex control algorithms with no hardware overhead and adapt them to specific hardware or conditions even during system operations. In addition, SPLL practically eliminates the need of external components except those required to generate the local frequency. In order to control a voltage-controlled crystal oscillator (VCXO), the SPLL control output must be converted to voltage with a digital-to-analog converter. This requirement can be even further minimized if a digitally controlled crystal oscillator (DCXO) is used, which can be controlled directly by the SPLL.

Generally, the input phase tracking speed of the loop controller must be balanced to filter out the high-frequency noise while still following the low-frequency drifts of the reference clock. In the prototype for the feasibility study, PI control

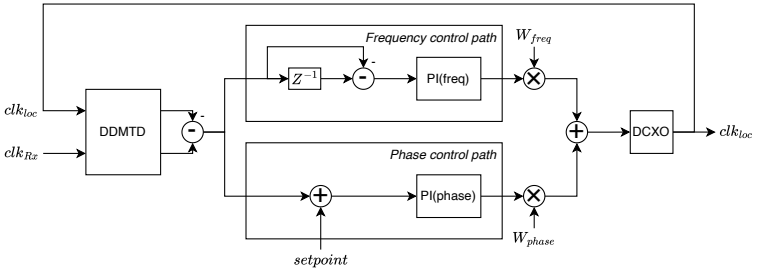


Figure 5.8: Concurrent PI phase and frequency control for potential use in the TFC system.

has been used for tracking the master frequency and phase. In White Rabbit, frequency and phase of the reference clock are tracked by separate controllers in a two-stage approach with the controller output multiplexed between frequency and phase tracking paths (Figure 5.7) [68]. Initially, the output of the loop controller is connected to the output of the frequency-tracking PI controller. When the frequency is aligned, the controller is switched to phase tracking.

While successfully fulfilling its purpose, the total settling time of such two-stage control includes individual settling times of both control paths. To reduce the total settling time while keeping the adaptability and noise-filtering properties, a concurrent dual controller architecture presented in Figure 5.8 has been developed for this feasibility study. The architecture structurally simplifies the White Rabbit approach and features a custom DDMTD implementation that provides time difference between input clocks as the output, instead of individual phase tags for the reference clock and the controlled local clock. This output is forwarded as an error input to both frequency and phase control paths. To track the frequency, the time difference value is subtracted from its previous value, thus tracking the phase drift rate. In the phase control path, the DDMTD output is offset with a user-defined setpoint and is then directly connected to the phase PI controller.

The concurrent PI control in this architecture is possible because the outputs of the frequency and the phase control paths are summed up and forwarded as the adjustment value to a DCXO that generates the local clock. Before adding the

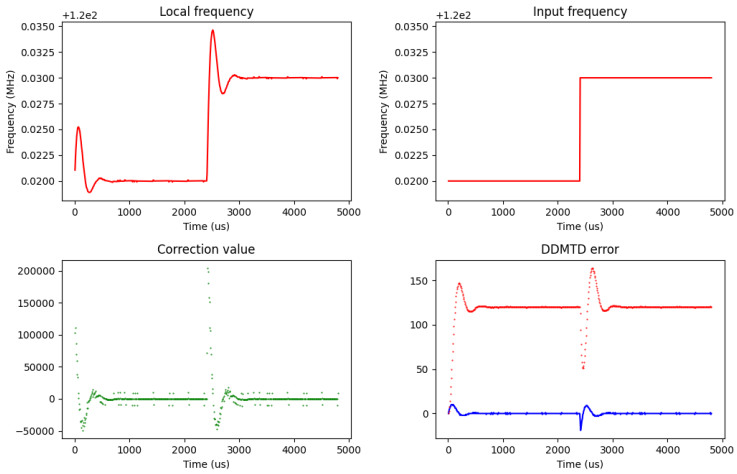


Figure 5.9: Sample simulation of the SPLL parameters.

control path outputs, weights are applied to control the influence of each path on the final control value. When the control coefficients of the PI controllers and the control path weights are properly tuned, the resulting PLL loop control architecture provides effective and highly adaptable phase and frequency alignment of the local clock with respect to the clock received from the corresponding master node.

To validate the control loop architecture and facilitate future parameter tuning, a hardware-aware simulation model has been developed in Python. Apart from the loop parameters, the model takes into account the fundamental properties and delays of the DDMTD, as well as the controls available in the DCXO. Given the large number of parameters that affect the performance of the control loop, the model allows for automated parameter optimization. An example of a simulation with roughly tuned parameters is shown in Figure 5.9.

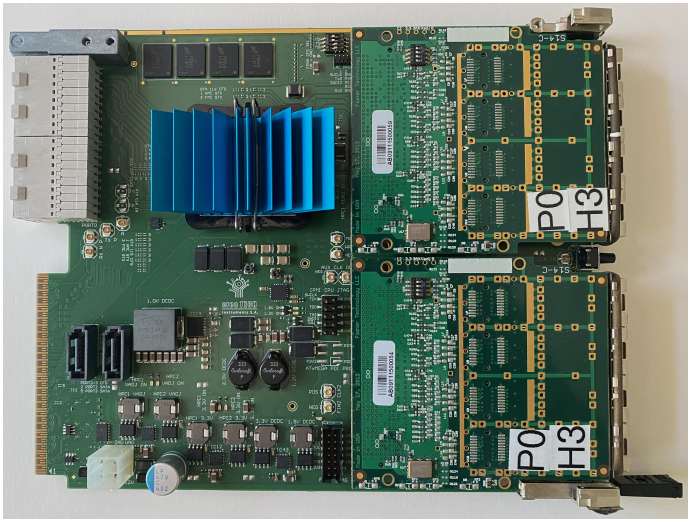


Figure 5.10: The AFCK board used as the TFC prototyping platform with two FM-S14 connectivity boards installed.

5.4 Prototype development

To test the approach to timing distribution where the TFC endpoints reach a common notion of time by a time-negotiation over each TFC link individually, a prototype has been developed. Due to the nature of the data readout in the experiment, the readout boards have a similar set of features to the TFC nodes. For that reason, same FPGA boards have been chosen for a minimal test implementation of the TFC system. At the moment of this feasibility study, AMC FMC Carrier with Kintex 7 (AFCK) boards were used for the prototype CBM setup [121].

The AFCK board features a XC7K325T FPGA by Xilinx and 8 of its 16 Gigabit Transceiver (GTX) channels are routed to two FPGA Mezzanine Card (FMC) connections mounted on the board. Expanded with two Faster Technology FM-S14 cards, each of these 8 transceivers have access to SFP+ slots with optical transceivers, which is required for the multipoint connections between TFC nodes in the hierarchical topology of the TFC system. A key feature of the board in the

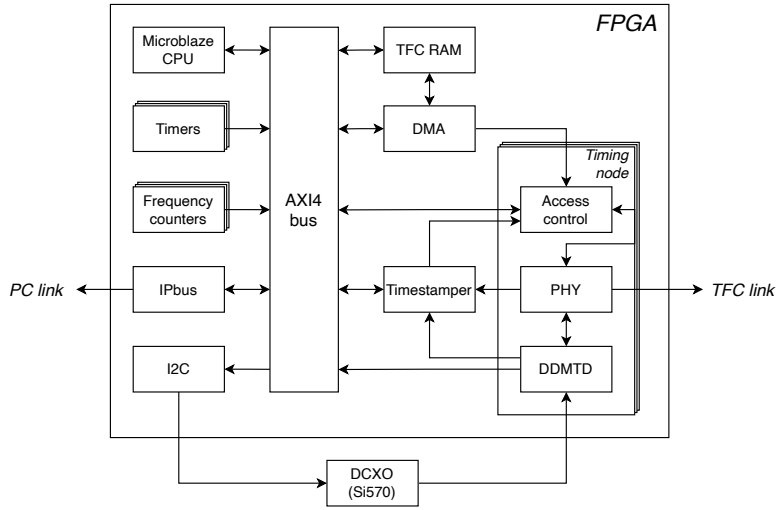


Figure 5.11: Gateware architecture of the prototype.

context of prototyping a synchronized distributed system is its remarkably versatile clocking architecture. The ADN4604 clock crossbar has 16 inputs and 16 outputs can be configured to provide a large variety of on-board clock sources to various FPGA functions, including GTX transceivers and general-purpose clock inputs. Some of the available programmable clock sources are, for example, Si570 DCXO and two IDT8N4Q001 clock generators installed on each FM-S14 extension board.

Although developed as a MicroTCA card, the board can also be operated in a standalone mode. This makes it highly suitable for system prototyping with potential subsequent integration into the final system without the need of switching the hardware platform. An additional benefit for prototyping is an independent from the FMC cards Gigabit Ethernet connection that can be used to control the card from a PC. In the current prototype, the IP-based bus control system (IPbus) protocol has been used to control and monitor the operation mode of the prototype and read out data to evaluate its performance [122].

The gateware architecture of the prototype is built around a *Microblaze soft-CPU core*, with access to all peripherals via an AXI4 bus as shown in Figure 5.11. Using a CPU at the center of the architecture allows to utilize the benefits of hardware-software codesign and efficiently divide tasks between hardware and firmware, with the time-critical and computation-heavy functions handled in hardware on the Register-Transfer Level (RTL) level and complex sequential procedures being implemented in software. Such complex procedures include, for example, initialization of the board hardware, tuning the DCXO via the I2C protocol, handling the synchronization protocol and general control and monitoring of the TFC node operation.

The TFC data transport channel is accessed via a dedicated *TFC RAM* and a *direct memory access (DMA)* controller. This approach allows to efficiently distribute data transport routine between firmware and hardware modules. With the firmware constructing the general frame structure and the payload, the data transport itself is handled with gateware components without blocking the CPU and allowing sufficient control over the time-critical part of the protocol implementation, e.g. hardware timestamping, which is essential for the accurate time exchange between nodes. Thus, with the assistance of the *access control* module, the timestamps are injected into the synchronization frames on the fly.

The core of time awareness in the TFC communication is the *timestamper*. This module defines the local time of the TFC node and this is the counter that Submaster and Endpoint nodes adjust in order to align their own times with the TFC time from an upstream link. This is the central component for both node synchronization and time-deterministic communication. The *Physical Layer (PHY)* module, being the closest module to the transmission medium in the data transport path, detects a special comma character specific for timing frame headers in either Rx or Tx communication channel and issues a pulse to the *timestamper* with a minimum delay to capture the current value of the time counter. As the captured timestamp is immediately presented at the output of the *timestamper* to the *access control* module, the corresponding frame timestamp can be directly integrated into the timing frame. At the same time, a momentary value of the time counter is always available to the *access control* module, and transmission of fast control frames can

therefore be scheduled for a specific time in the future, thus enabling time-triggered data transmission. If scheduled sufficiently in advance to avoid fragmentation of a currently transmitted frame, and fast control frames have higher priority than synchronization frames, transmission of the fast control frame exactly at the specific time is guaranteed. The lower priority of the timing frames does not degrade synchronization accuracy, since the synchronization protocol is immune to variable delay between SYNC and REPLY frames, as described in the beginning of this chapter.

Access control, *PHY* and *DDMTD* modules constitute a *timing node*. The *timing node* block contains all the logic that provides hardware support for time-aware data transport and is dedicated to each link. Thus, in the system configuration with 8 optical links per TFC node, the design contains 8 *timing nodes* in gateware. Such hardware support of time-critical data transport relaxes the requirements for the CPU core used, which does not have to be optimized for real-time performance in this architecture. Additionally, dedicated gateware modules for handling data transport allow for direct data forwarding between upstream and downstream links in the Submaster nodes as defined in the fundamental system architecture in Section 4.2.3.

Gateware designs of the Master, Submaster and Endpoint nodes differ primarily in the number of the required timing nodes. Master and Submaster designs must interface with multiple neighboring TFC nodes and should therefore include as many timing nodes as there are links available on the hardware platform. An Endpoint design, on the other hand, only includes one timing node as it always has to interact with one upstream node, regardless of the hardware platform or its configuration. Although the design of timing nodes is unified, the timing nodes facing upstream are used differently than those facing downstream. This is caused by the fact that the upstream-facing links are used to receive the global TFC time and the local clock rate and the time must align with it. For this reason, upstream-facing timing nodes are used as the receiving side of the synchronization protocol and the recovered clock from the receiver on this link serves as the reference input to the local soft-PLL that is used to track its frequency and, if needed, phase. To avoid conflicting local time references leading to metastable local time conditions,

only one reference timing node can be present in each Submaster or Endpoint node. Regardless of orientation, both upstream- and downstream-facing nodes include a DDMTD to enable accurate phase measurement necessary for implementation of the synchronization protocol.

To align the clock frequency of a TFC node with the rest of the system, the soft-PLL is implemented with a closed control loop consisting of the *DDMTD*, *Microblaze CPU* firmware, an I2C controller and the *Si570 DCXO*. The DDMTD in the upstream-facing timing node provides phase difference measurements between the 120 MHz local node clock and the 120 MHz clock recovered from the receiver data in the corresponding timing node. These phase shift measurements are passed to the CPU firmware, where the DDMTD measurements are used to calculate the adjustment value for the DCXO. A DDMTD helper clock of 119.5 MHz is generated using a Mixed-Mode Clock Manager module, a Xilinx built-in PLL. This helper clock sets the N value to 239, which corresponds to the time difference measurement resolution of 35 ps. While the resolution is likely not optimal for the final implementation of the system, it entails little implementation overhead and is sufficient for the feasibility study.

The minimum period of the control loop is defined by the delays introduced by its various components. The delay of communication with firmware and execution of control calculations are negligible compared with the other components in the loop. At the same time, the delay of the DDMTD measurements are defined by the N parameter and the time to apply adjustments in the DCXO can be calculated from its control protocol. Thus, with the N parameter of 239, measurements of the DDMTD take 2 μs, according to Equation 5.10. To improve accuracy, averaging by the factor of 2 is applied to DDMTD measurements, increasing the measurement delay to 4 μs. The Si570 DCXO is controlled via I2C, using the maximum supported protocol clock frequency of 400 kHz. With all the registers needed for frequency adjustment being written and verified via readback, $\sim$ 630 μs is needed, making this the slowest operation in the loop. With this combination of delays, a loop frequency of 1,5 kHz is feasible for the soft-PLL in the current implementation.

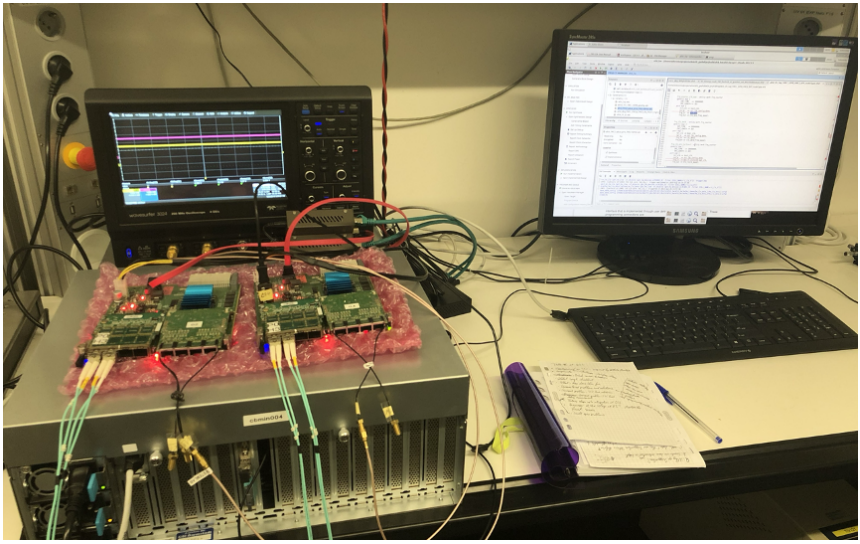


Figure 5.12: Test setup for the prototype of the time self-adjustment approach.

To ensure consistent time intervals between regular operations in firmware, such as soft-PLL cycles and (re-)synchronization of the node itself, timers with various periods are used. In the current implementation, a timer with a period of 650 μs is used to define the time base for the soft-PLL, resulting in the soft-PLL loop frequency of $\sim$1.54 kHz, and a timer with a period of 1 s defines the link resynchronization interval.

Additional monitoring module have been added to the architecture to monitor stability of the prototype operation. Thus frequency counters are used to verify the alignment between Rx and local frequencies and the IPbus interface ensures communication of the TFC node with a PC to control the internal state of the prototype firmware.

To study the advantages and constraints of the approach, an evaluation setup with two prototype TFC nodes has been built: one to represent the TFC Master, and the other as the Endpoint (Figure 5.12). They both implement the same architecture with one timing node each, with the timing node of the Master configured as

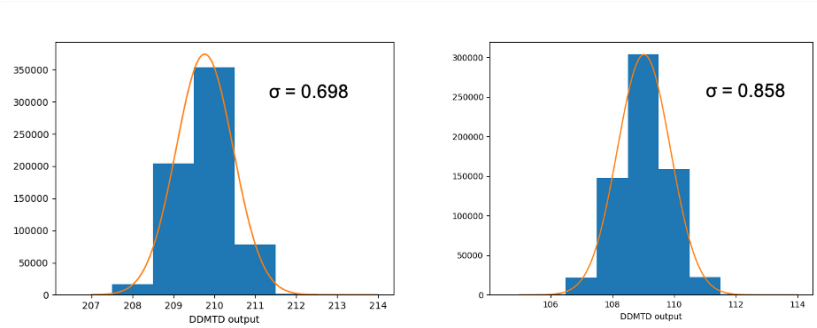


Figure 5.13: Measurement of the real resolution of the DDMTD module.

downstream-facing, and that in the Endpoint configured as upstream-facing and receiving the time from the TFC link. Each AFCK board in the evaluation setup is equipped with one FM-S14 FMC board to serve the TFC link, and one custom debug card to provide probe points for an oscilloscope.

5.5 Feasibility analysis

As the first step of the analysis of the self-adjustment approach, the real resolution of the DDMTD module has been evaluated. Although the nominal resolution with the given N value of 239 can be easily calculated as 35 ps, the clock jitters, drifts and noises in the system inevitably affect the accuracy of phase measurement. For this reason, the consistency of the DDMTD measurements has been measured to verify the actual measurement resolution of the DDMTD. Two measurements have been made on syntonous 120 MHz clocks with two different phase shifts, with 650000 samples in each measurement. The results are presented in Figure 5.13. The acquired histograms have been fitted with a Gaussian function to quantify the measurement accuracy. As a result, a standard deviation of 0.698 DDMTD counts has been obtained in one measurement and 0.858 in the other. These values in DDMTD counts correspond to 24.34 ps and 29.92 ps, respectively, according to Equation 5.11.

Afterwards, the fundamental functionality of the architecture has been successfully verified, with the appropriate scheduling of the regular events and the correct configuration of the hardware components. Nonetheless, adjustment of the Si570 DCXO as part of the soft-PLL in a certain area of the adjustment range has consistently triggered unlocking of the internal Quad Phase-Locked Loop (QPLL) of the Xilinx Gigabit Transceiver, leading to a failure of the TFC link. This indicates insufficient clock stability at the output of the soft-PLL to maintain multigigabit communication. Resolution of the output clock stability issue requires further investigation and, potentially, replacement of the DCXO with a more compatible device. This investigation, however, lies outside of the scope of the feasibility study.

Analysis of the synchronization approach and of the implemented prototype allows to identify general advantages of the approach. One of them is fundamental support of fast control messages of arbitrary length, as synchronization messages can be delayed without harming synchronization accuracy avoiding fragmentation of fast control traffic. Thus, the approach would be preferred over others in applications with unknown or variable payloads. Nevertheless, the fast control messages must be carefully scheduled to reach endpoint nodes at a predictable time. Another advantage of the approach is the vast configurability of the synchronization performance. By adjusting the properties of the DDMTD and loop control algorithms, the loop period of the soft-PLL and its loop bandwidth can be adjusted even dynamically in the runtime, allowing in-field optimizations to adapt to the needs of a target application.

The time self-adjustment approach to FPGA node synchronization has been demonstrated to be generally applicable to the TFC system. However, its use in the TFC system requires careful consideration of the hardware platform, as operability and accuracy of node synchronization is dependent on the used hardware clock generators and, optionally, PLL devices. Besides, the successful application of the approach requires extensive configuration and validation of the numerous parameters of the soft-PLL. To improve the DDMTD resolution, the DDMTD helper clock with a finer frequency offset can be generated using an external PLL or an integrated all-digital PLL proposed by Xilinx [123]. Additionally, the optimal

parameters of the soft-PLL and its transfer function can be accurately estimated using the simulation described at the end of Section 5.3.

5.6 Summary

In the feasibility study of the time self-adjustment approach, a prototype has been developed to study its advantages and limitations. Inspired by the White Rabbit project, the prototype reuses the concept of the DDMTD, which allows each node to measure phase shifts between clocks and derive its own time shift related to the reference clock.

The prototype concept for the feasibility study is based on a CPU core and an AXI-Lite bus, leveraging the advantages of the hardware-software codesign principles. The concept features an improved design of a software PLL that allows for dynamic reconfiguration of the loop parameters, thus offering better loop bandwidth control and faster lock times. Additionally, the software PLL has been adapted for the use of a digitally controlled oscillator, as opposed to the combination of a Digital-to-Analog Converter (DAC) and a VCXO in the original design.

A prototype with the software PLL has been implemented and tested, revealing incompatibilities between the constraints of the experiment and the capabilities of the selected hardware platform. As a result of the feasibility study based on the developed prototype, valuable insight has been gained into the development strategy that leads towards building a TFC system based on the time self-adjustment approach and the scope of work required to achieve full functionality.

6 Approach 2: time distribution over latency-deterministic links

In the second approach in the feasibility study, the common notion of time in a network is achieved by broadcasting the global master time across the network, with the nodes adopting it directly, with no transformations. This way, the accuracy of the synchronization is dictated by the design of the system and its hardware, instead of runtime negotiations, continuously applied corrections and compensations. Therefore, this time distribution method is immune to link asymmetry and does not require complex software control algorithms, provided predictable data transmission delays from the master node to endpoints.

As sources of time discrepancies between nodes are addressed at the hardware-level at the design stage, this approach offers simplified node architecture and reduced protocol overhead, while maintaining system synchronization and determinism with high precision. With the timing information only traveling in the downstream direction, this synchronization method is not affected by system congestions and scheduling, which improves its scalability.

6.1 Time synchronization concept

As opposed to the time self-adjustment method studied in Chapter 5, direct time distribution causes the time to be offset between master and endpoints, since there is no compensation of the round-trip delay involved. Nonetheless, this method is applicable in the TFC system as only time alignment between CRI boards is required, as discussed in Section 4.1. Moreover, latency optimizations of the links

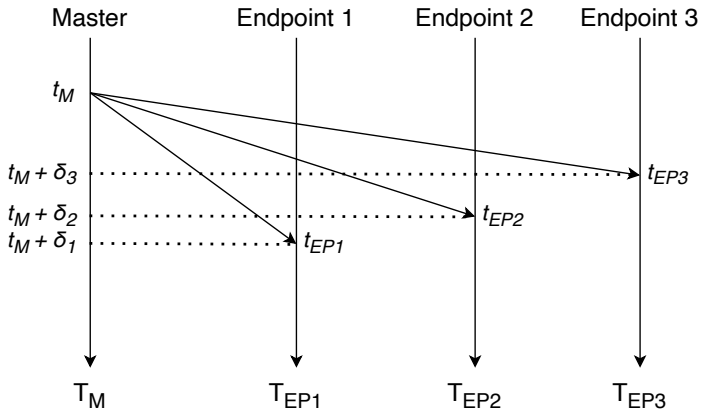


Figure 6.1: Time distribution with a constant transmission delay.

to achieve the required synchronization accuracy improve the overall downstream latency performance for downstream messages, including fast control messages, which is one of the key requirements of the system.

Figure 6.1 shows the general synchronization principle in this approach. With minimum communication between master and endpoint nodes, the nodes override their local time with the time received from the master node. The relationship between the local time in each endpoint node and the global master time is then continuously defined by Equation 6.1, with n being the number of the endpoint node and σ_n being the transmission delay from the master to this endpoint. With the transmission delays σ_n towards each endpoint node being equivalent or having a known and constant time offset, the time between all endpoints can thus be unambiguously reconstructed and aligned.

$$T_{EPn} = T_M + \sigma_n \tag{6.1}$$

With the development focus in this approach shifted towards hardware and gateware design, all clocks in the system must have predictable offsets and all components

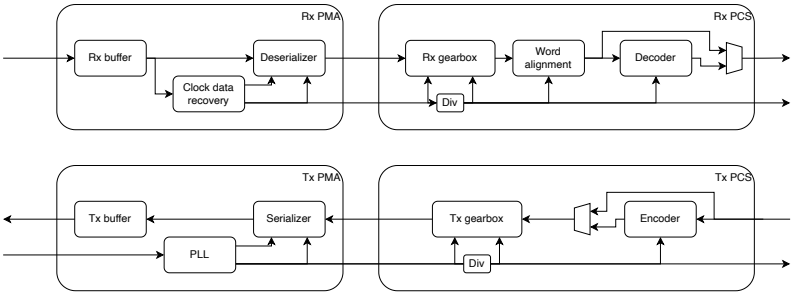


Figure 6.2: General architecture of an FPGA implementation of an MGT IP core. Only basic functions are included, vendor-specific implementations may include additional components.

in the transmission paths of the timing messages must have deterministic input-to-output delay. These aspects have been carefully analyzed and addressed in the feasibility study and described in the following sections.

6.2 Transceiver clock domain management

Predictable latency is the prerequisite for a successful implementation of the synchronization principle in this feasibility study. However, achieving predictable latency in the datapaths of multi-gigabit transceivers (MGTs) is not a trivial task, since high bandwidth comes at the cost of complex architecture of the interface components. In FPGA devices, the PHY is implemented as a hard IP core due to its mixed-signal nature and the high-rate datapaths that operate at frequencies beyond the capabilities of the programmable logic. A general architecture of the FPGA implementation of the MGT PHY core is presented in Figure 6.2.

The data in MGTs are handled independently in the receiver (Rx) and transmitter (Tx) datapaths. As the received data are generated synchronously to the remote clock that is independent of the local node clock, the receiver datapath relies on the clock that is recovered from the receiver link. This receiver clock is extracted from the incoming data stream on the medium by a PLL that locks directly to the data, as part of the CDR circuit.

The bit stream in the receiver arrives at multi-gigahertz rates, which cannot be directly handled by the FPGA logic. For this reason, the received data are deserialized in multiple stages and presented to the user in a parallel form, synchronous to a divided-down version of the clock recovered by the CDR. Thus, the receiver datapath inherently creates its own clock domain, and the received data must be safely passed to the local system clock domain in order to maintain predictable data transmission latency in the system. Even if the frequency of the remote and the local nodes are locked and aligned, the phase shift between the recovered and the local clocks cannot be considered predictable without additional measures.

Similarly, the transmitted data are serialized in multiple stages and fed as a bit stream into the medium. Since the serial data rate on the medium is far beyond the capabilities of the FPGA fabric, the serial data clock is generated by a dedicated PLL, which uses a local node clock as a reference frequency. This means that the Tx serial clock can be generated from the local system clock used in the node design. However, even in this case, the multi-stage serialization process creates clock domain crossings that are handled internally in the IP core and the data must be fed synchronously to the clock that is derived from the Tx serial clock, which may still have an unpredictable phase shift relative to the original local system clock. This creates a separate Tx clock domain that may be syntonous to the local clock by design, but the predictability of the latency upon feeding the Tx data into the transceiver must be handled separately.

For the ease of integrating MGTs into designs that do not require predictable data transmission latency, the transceiver IP cores have built-in FIFOs in both the Tx and the Rx datapaths to handle the clock domain crossings between the MGT datapaths and the local system clock domain. Although typically present in transceiver architectures, these FIFOs introduce uncertain latency in the datapath and do not provide the features to ensure predictable latency and are omitted from Figure 6.2.

While it is possible to configure the MGT IP cores to have a fixed latency in the datapaths within the IP core, the burden of passing the data between the MGT and

local system clock domains lies on the system designer. To handle the data at this level with a predictable latency, separate FIFOs with fill level monitoring are used outside of the core, to ensure that the latency they introduce is known at any time. This allows the system logic to recover the error introduced to the received time by the Rx FIFO and apply corresponding corrections. Besides, this approach allows for a potential control of this latency by introducing and dropping filler messages in the TFC data stream.

6.3 Hardware-level clock distribution

Following the hardware-oriented design principles chosen for this feasibility study, the global hardware clock is distributed from the TFC master to all downstream nodes at the lowest level with as little conversion as possible. This is achieved by leveraging the self-synchronous properties of the multi-gigabit optical communication, which allow each downstream node in the TFC network to recover the upstream hardware clock using the built-in CDR circuitry and reuse the clock as the local system clock and propagate it further downstream when used in a submaster node.

Generally, CDR circuits produce clock signals with relatively high jitter levels, which do not fulfil the stringent jitter requirements for the reference inputs of the FPGA gigabit transceivers. Thus, the recovered clock cannot be directly reused for further downstream transmission and is passed through a jitter-cleaning PLL instead. This way, each TFC node reuses the hardware clock received from the immediate upstream node, which creates a scalable clock distribution path with minimal interference with the local node design. Implementation of this principle is depicted in Figure 6.3.

In order to successfully propagate the global hardware clock through the hierarchical TFC system, the local clock paths in each node must not introduce phase uncertainty and have fixed input-to-output delays. As the CDR PLLs lock directly to the incoming data stream, the phase noise of the recovered clock is inevitably

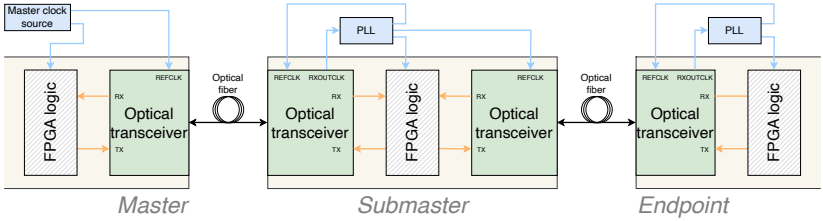


Figure 6.3: Clock distribution via self-synchronous optical links.

affected by the density of bit transitions in the bit stream. While high-frequency phase noise is filtered out by the jitter-cleaning PLL further in the local clock path, low-frequency phase variations must be handled by managing the number of consecutive bits of the same value in the bit stream. In the current implementation, this problem is addressed by using 8b/10b line encoding, which guarantees a bit transition after at most 5 consecutive bits with the same value at the cost of 20% overhead. This overhead is tolerable in the feasibility study, yet it can be reduced by using a similar encoding scheme that uses longer characters, e.g. 64b/66b encoding. The second major component in the local clock path, the jitter-cleaning PLL may introduce phase variance due to uncertain input-to-output delay. This is avoided by enabling the zero-delay mode, which can be commonly found in modern commercial PLLs.

As defined in the requirements for the TFC system in Section 4.1.1, the global clock that must be distributed to the endpoint nodes has a frequency of 40 MHz. Using this frequency as the parallel word frequency of the MGT datapaths significantly limits the data line rate and hence the bandwidth to only 400 Mb/s, which violated another requirement of the system for high link bandwidth as defined in Section 4.1.3. This limitation is overcome by introducing a 120 MHz transport clock. As the frequency of this clock is an integer multiple of 40 MHz, this clock can carry the phase information of the global clock. Reconstruction of the 40 MHz global clock base from the 120 MHz transport clock is handled by each timing receiver individually using a technique described in Section 6.5.

6.4 Protocol definition

At the cost of some protocol overhead, introduction of the intermediate transport clock allows for a protocol that can be scaled up in terms of link bandwidth. In the current prototype, a transport clock of 120 MHz has been introduced, which allows for the link bandwidth of 2.4 Gbps. This way, 4-byte parallel words are available for transmission in the 120 MHz clock domain.

A DC-balanced 8b/10b coding scheme is used on the serial data to ensure optimal performance of the CDR scheme as described in Section 6.3. This coding scheme extends every byte of the transmitted data to 10 bits, providing in addition a possibility of transmitting special non-data characters called K-characters. There are 12 K-characters defined in the coding scheme in total that can be used for the needs of the protocol. One of these symbols, a comma character, is used to align receiver deserializers, so that the parallel received data are correctly presented to the rest of the design.

Before data messages can be exchanged over a TFC link, the link must be successfully initialized. The procedure is meant for the TFC nodes to detect the presence of a valid TFC node on the other end of the link and configure their respective receivers until both of them are ready for communication. In case if it is requires to reset the downstream node in the runtime, the link initialization procedure will be triggered again upon reconnection of the downstream node, i.e. the concept of the TFC link supports hotplugging.

Link initialization involves 3 types of service messages. *SYNC1* and *SYNC2* messages include a comma character and are used to let the upstream and the downstream nodes align their receivers, and the *IDLE* message is a predefined static message indicating that the link is ready for use yet no messages need to be transmitted at the moment.

Upon bootup, the upstream node continuously transmits the *SYNC2* message, regardless of the link status. When a downstream node is connected, it starts sending the *SYNC1* message into the link, thus allowing for the upstream node to

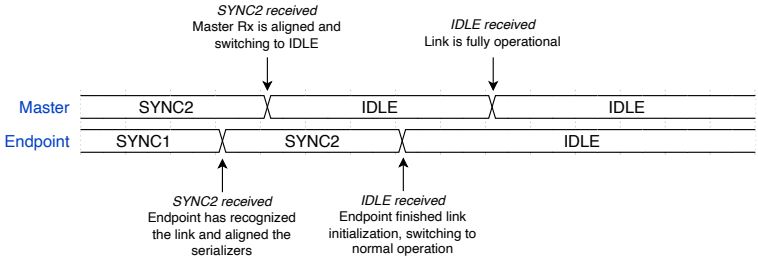


Figure 6.4: Link initialization procedure using direct communication between a master and an endpoint nodes.

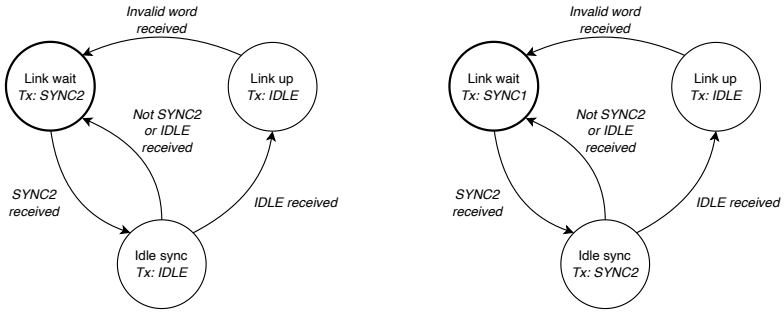


Figure 6.5: State diagrams of the finite state machines used on the upstream (left) and the downstream (right) ends on the link.

align its Rx serializers while adjusting its own receiver. When the downstream node receiver is ready and has successfully received a sufficient number of *SYNC2* messages from the upstream node, it indicates its readiness for the next initialization stage by streaming the *SYNC2* message. The upstream node detects this and acknowledges it by sending *IDLE* messages. The downstream node receives the *IDLE* message and indicates its readiness for normal operation by replying with *IDLE* messages as well. After successful reception of the *IDLE* messages from the downstream node, the upstream node concludes that the link is fully established and is ready for normal operation. The procedure is illustrated in Figure 6.4 using direct communication between a master and an endpoint nodes as an example.

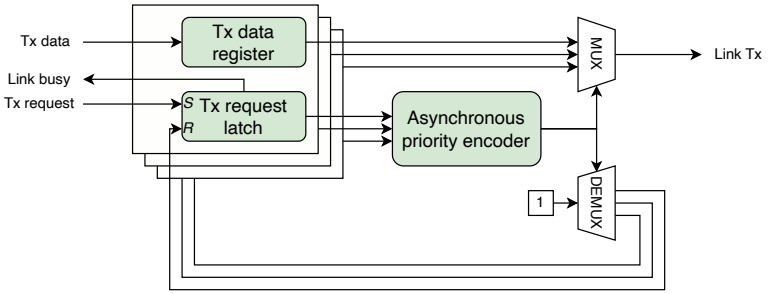


Figure 6.6: A functional diagram of the link prioritization scheme used for shared link access.

This initialization sequence ensures consistent link establishment even in case of a link failure or various bootup orders of the nodes. As the procedure is guided by the upstream node, no timing or control messages are exchanged before the link is fully operational. The sequence is executed by the nodes with two separate state machines on each end (Figure 6.5).

During the normal link operation, different message can be exchanged between TFC nodes. To distinguish between them, 5 most significant bits out of 32 available are used as a message header that includes a type identifier and a static reference pattern to monitor message integrity. The reference pattern is used to detect link loss and prevent reaction of the nodes to accidentally received control sequences. A dedicated protocol checker logic continuously monitors protocol integrity. Thus, for example, in case of a link loss due to physical damage to the cable, the link immediately shuts down and the initialization sequence is triggered, waiting for the node to be reconnected.

As multiple message types must share the TFC link, the link access prioritization is another important feature of the TFC protocol. In this scheme, access to the TFC link must be explicitly requested by the potential senders using a request flag. If the link is busy at the moment of the request, the sender must wait for the link to be released by the higher-priority senders before attempting to send further data (Figure 6.6). An asynchronous priority encoder is used to select an input sender channel with the highest priority for each transmission, with lower

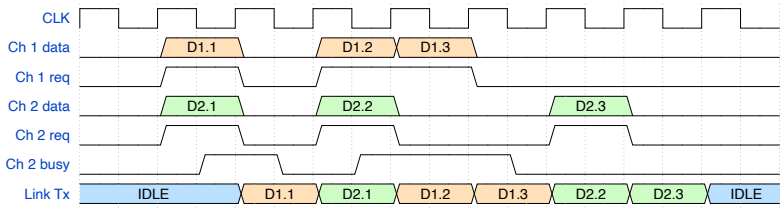


Figure 6.7: An example of prioritized link access with two senders concurrently requesting message transmission.

priority messages being registered and queued. To ensure link access with no dead-time, i.e. when messages can be transmitted from different senders with no interruptions, the access request and link ready flags are asserted on opposite clock edges. An example of link access arbitration with two senders requesting access to the link at the same time is presented in Figure 6.7.

6.5 Reconstruction of the global clock phase

With the TFC link transmitting messages synchronously to the 120 MHz transport clock, only the transport clock can be reliably and unambiguously reconstructed. At the same time, one of the goals of the time distribution function is to distribute the 40 MHz global clock across the experiment data processing boards, so that the phases of their local representations of the global clock have constant phase offsets. Clock division cannot automatically guarantee a fixed phase of the 40 MHz clock, as it is uncertain to which of the three possible 120 MHz clock edges the 40 MHz output clock s aligned to. Therefore, the correct phase of the global 40 MHz clock must be recovered separately.

A counter running in the faster transport clock domain is used to track the phase of the 40 MHz clock in all nodes. Initially aligned such that a counter value of 0 is read on each 40 MHz clock edge, the counter counts from 0 to 2, effectively counting the periods of the 120 MHz clock within a single 40 MHz clock period. These are referred to as subcycles in the context of this work. However, the nodes

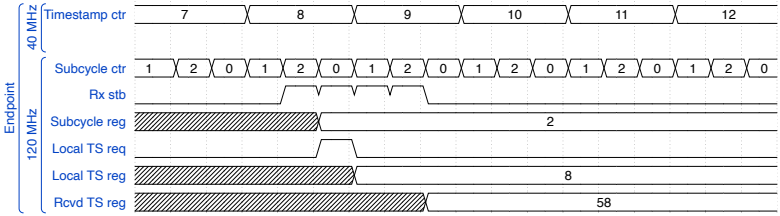


Figure 6.8: An example of registering the phase position of the received 40 MHz clock by an endpoint using a subcycle counter.

start unaware of the position of the global clock, and the phase uncertainty of their local 40 MHz clocks remains at this point.

Being the origin of the global time in the system, the master node implicitly encodes the phase of the 40 MHz in the timing communication by always initiating transmission of a time broadcast on subcycle 0. When each of the downstream nodes detects the time broadcast from the master node, it registers the subcycle value of the first time message and offsets its own subcycle counter to align its local subcycle 0 with the received time transmission. Afterwards, the downstream node must adjust the phase of its 40 MHz clock accordingly. The procedure of registering the phase of the global 40 MHz clock by an endpoint is presented in Figure 6.8.

At the moment of receiving the first timing message, the downstream node registers its local time counter value as well as the time received from the upstream node. Thus, the process of phase reconstruction using the subcycle counter does not disrupt the master time distribution.

6.6 Prototype development

The implementation of the time distribution concept for this feasibility study has been developed using the same hardware platform as the data-processing FPGA boards in the CBM prototype setup at the time. The BNL-712 platform board is

designed specifically for data acquisition in particle physics experiments and meets the general hardware demands of the TFC system as well. Developed as a 16-lane Gen3 PCIe card, the board is equipped with a Xilinx Kintex Ultrascale XCKU115 FPGA [124]. Through eight high-density Avago MiniPOD optical connectors, the FPGA on the board has access to 48 full-duplex optical links that can operate at rates up to 12.8 Gbps. Two jitter-cleaning PLLs are installed on the board to produce reference clocks for the large number of Gigabit transceivers available. In addition, the board supports extension mezzanines for better adaptability to various application-specific interfaces.

Each of the eight on-board MiniPOD connectors aggregates 12 one-way optical fibers that are routed on the board to two high-density MTP connectors, each providing 24 full-duplex optical channels. As a result, point-to-point optical connections between boards require splitting the aggregated links. For this reason, downstream connections from the master and submaster nodes are routed through optical patch panels, which break out the high-density MTP interfaces into individual full-duplex LC connections. On the endpoint side, communication over a single upstream optical link is achieved via a dedicated mezzanine card that hosts an SFP cage, allowing direct connection to a single full-duplex fiber link.

Based on these boards, the control of the prototype TFC nodes is aligned with the hardware control infrastructure of the prototype implementation of the CBM detector setup. All nodes in the system are thus individually controlled via the PCIe interface, with the boards mounted into PCIe slots of commercial off-the-shelf servers as shown in Figure 6.9.

Built with the Stacked Silicon Interconnect (SSI) technology, the Xilinx XCKU115 device is composed of multiple FPGA die slices, or super logic regions (SLRs) [125]. For efficient FPGA resource utilization, node control infrastructure is split into two identical paths, one per SLR (Figure 6.10). Thus, each SLR is interfaced independently with the PCIe bus of the server node using an 8-lane Xilinx PCIe hard IP core. A PCIe-to-Wishbone bridge ensures connectivity between the PCIe bus and the Wishbone bus, which provides register access to individual peripheral

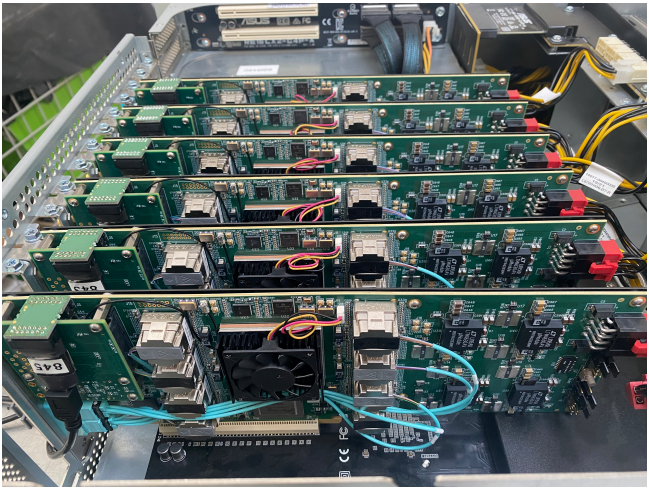


Figure 6.9: Multiple BNL-712 boards mounted in a server node.

components in the gateware. To manage a complex hierarchical structure of register address ranges on the Wishbone bus, the bus configuration is handled by the Address Generator for Wishbone (AGWB) tool [126]. Register files of gateware components can thus be accessed from software using the AGWB built-in Python library together with the experiment Device Control Agent (DCA) framework [30].

The prototype gateware architecture is presented in Figure 6.11, with the focus on the components that directly implement TFC functionality. The control-related components are intentionally simplified in the diagram.

The global time of the experiment originates in the timestamper module of the TFC master node as a 64-bit time counter value. Running in the 40 MHz system clock domain that reflects the base frequency of the experiment, the global time is counted with increments of 25 ns. The time counter can be preset by the user at any time, to align it with any standard time reference, e.g. Coordinated Universal Time (UTC).

The time transfer downstream to other TFC nodes is initiated by the *sync counter*, which determines the period of time distribution in the system.

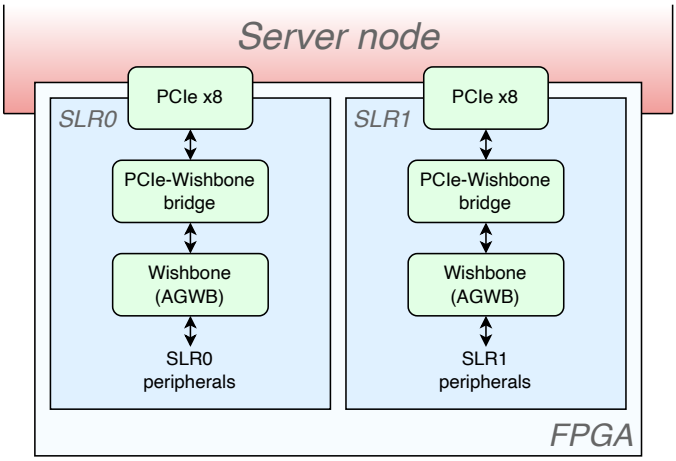


Figure 6.10: Control paths of a TFC node.

To distribute time information to downstream nodes, the timing master module serializes the current counter value. This serialized time value is transferred into the 120 MHz transport clock domain, making it compatible with the data transport interface. In addition to serialization, the timing master module continuously monitors the subcycle value and ensures that the first message of each serialized time value is always transmitted during subcycle 0. This mechanism is essential to enable the functionality described in Section 6.5.

The transport of messages over the TFC link is coordinated by the link access unit, which plays a central role in ensuring latency-deterministic data handling, which is the core feature of the system. This unit manages protocol-level tasks and link arbitration as specified in Section 6.4. It is responsible for the definition and composition of TFC messages, management of transceiver clock domains as described in Section 6.2, and the execution of the link establishment procedure.

To maintain deterministic latency in both the transmit and receive datapaths, the FIFOs in the link access unit are equipped with fill count outputs (Figure 6.12).

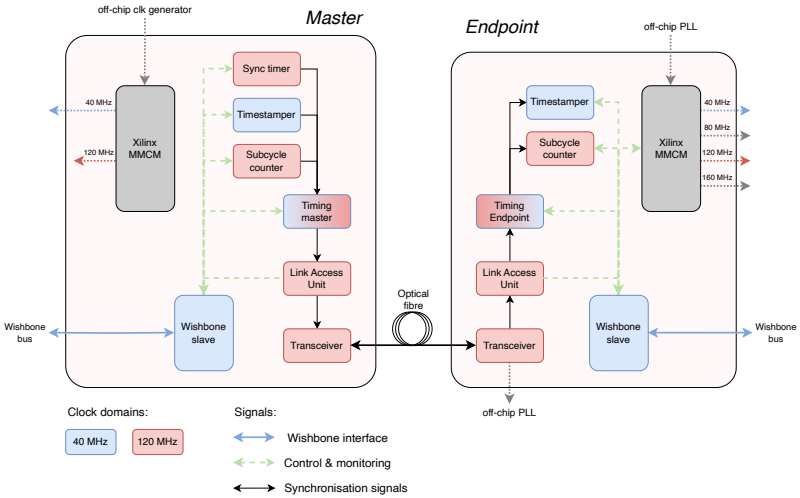


Figure 6.11: Gateware architectures of a master and an endpoint nodes.

This enables precise tracking of the delay introduced by buffering, allowing the receiving node to apply a local time correction accordingly.

Furthermore, the link access unit continuously monitors protocol integrity at the receiver side, allowing for the detection of potential link failures. On the receiving end, a counterpart link access unit accepts the transported messages and forwards them to the appropriate functional recipient.

When a timing message is received, the timing endpoint module registers its payload and deserializes the time information. After receiving all four fragments that constitute a complete time value, the timing receiver calculates a time correction. This correction is transferred from the 120 MHz transport clock domain to the 40 MHz system clock domain and applied to the local time counter of the receiving node.

In the implemented prototype, only timing messages are transmitted. However, the link access unit has already been designed to support shared link access. In this scheme, fast control messages are assigned the highest priority and are granted

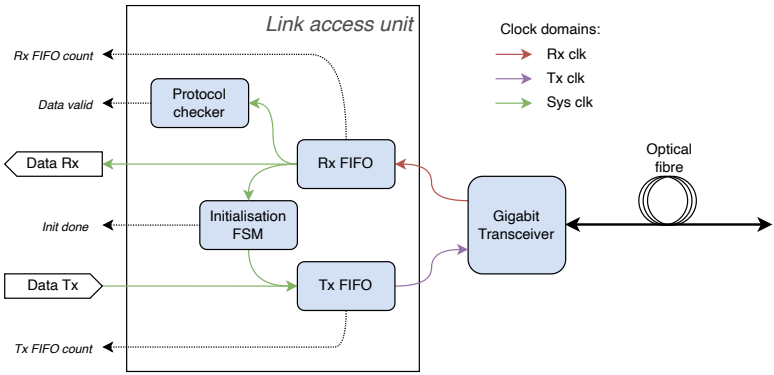


Figure 6.12: Architecture of the link access unit.

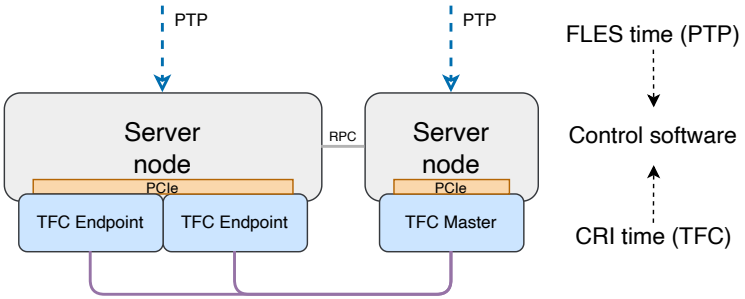


Figure 6.13: Test setup for evaluation of the prototype design.

immediate access to the link due to their stringent latency constraints, while timing messages are allowed to experience bounded delays.

6.7 Feasibility analysis

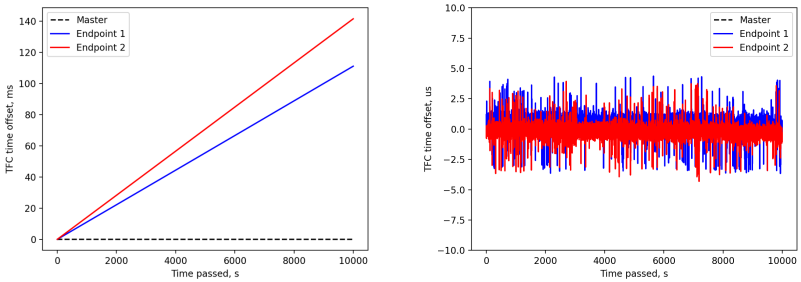
To verify the concept of latency-deterministic time and clock distribution, a minimal TFC system was constructed at GSI Helmholtz Centre for Heavy Ion

Research (GSI) in Darmstadt. The system consists of three TFC nodes: one master node and two endpoint nodes. Both endpoints are hosted in a single server and connected via multimode optical fibers to the master node, which is installed in a separate server (see Figure 6.13). As the focus of this prototype is on functional readiness, the verification of general time alignment across a single TFC link between the master and one endpoint is essential. In addition, the time alignment between the two lateral endpoints is also evaluated.

The nodes are controlled using the DCA software and the infrastructure described in Figure 6.10. As the DCA software supports distributed operation, both server nodes are managed from a single Python script. Access to local control functions of each node is provided through an Remote Procedure Call (RPC) interface over Ethernet.

A common time base is required in the test setup to track the TFC time across all nodes. Since the objective is to verify functional readiness rather than high-precision timing, the time base is expected to provide strong long-term stability. The short-term variation requirements are relaxed because they are typically not sufficiently met by commonly available commercial solutions. For this purpose, the servers were synchronized using PTP, which achieves system-level time synchronization with microsecond-level accuracy. Although this method does not provide high-precision alignment, it is sufficient for tracking long-term TFC time consistency during the test.

During the evaluation, the local TFC time of each node was recorded relative to the corresponding server system time over a test period of 10^4 s. The extended duration ensures that the long-term functional stability of the system can be reliably assessed. As a baseline, the test was initially performed with synchronization disabled. In this configuration, the local clocks of the TFC nodes naturally drifted apart. This allowed the intrinsic drift of independent nodes to be quantified. A second run was then conducted with synchronization enabled in order to evaluate the effect of the time distribution system on reducing time drift between nodes. Although absolute time offsets between nodes are still present, these are not



(a) Asynchronous configuration.

(b) Synchronous configuration.

Figure 6.14: Test results of the TFC prototype.

relevant for the intended evaluation, as explained in Section 4.1.1, and were normalized in the analysis.

In the unsynchronized configuration, the endpoint clocks drifted from the master node by approximately 140 ms and 110 ms, respectively, over the test duration (Figure 6.14a). These values imply that if time synchronization were based only on periodic time value updates once per second without a shared clock frequency, the best possible time stability would be limited to around 14 μs.

In the synchronized configuration, the TFC time remained stable across all nodes throughout the test. The measured deviations remained within 8 μs relative to the master clock (Figure 6.14b). The dominant source of short-term timing variation is believed to originate from the PTP-based time base and the time readout over the PCIe interface. Despite this, the test confirmed the long-term functional stability of the time synchronization approach and demonstrated its readiness for deployment in a mission-critical system.

6.8 Deployment in the Phase-0 CBM as TFC1

During the feasibility analysis, significant advantages of this approach were identified over the approach described in Chapter 5. Thus, it has been confirmed

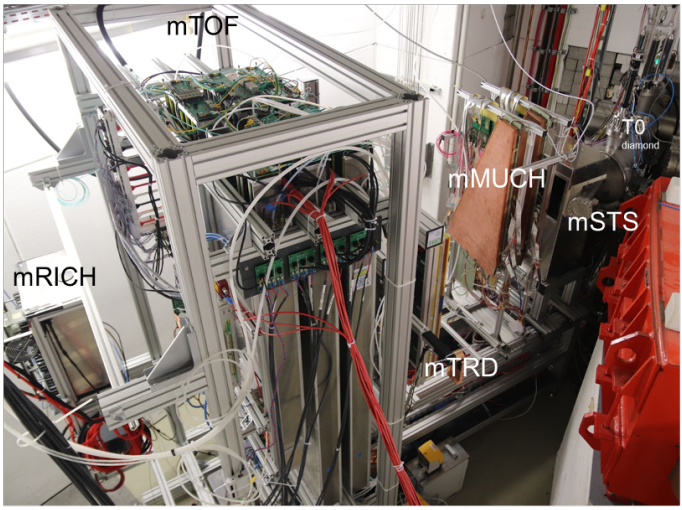


Figure 6.15: The Phase-0 CBM experimental setup. Credit: CBM collaboration.

that addressing the latency variations at the hardware level makes it possible to achieve deterministic time differences in the system with a minimal added architectural complexity, since no complex control infrastructure is required in the system nodes. This advantage significantly improves the resource efficiency of the design and reduces its overall complexity, which played a major role in its subsequent adoption in CBM.

After successful functional verification covered in the previous section, the prototype design of the TFC system was deployed in the Phase-0 setup of the CBM experiment shown in Figure 6.15. This configuration included multiple subdetectors and implemented a complete data acquisition chain. A total of 14 CRI boards was directly connected to a single TFC master node, forming the central timing distribution layer of the system.

The global TFC frequency was synchronized with the reference frequency provided by the particle accelerator via a White Rabbit network. To interface the TFC system with this network, a PEXARIA5 PCIe card was used as a White Rabbit receiver [127]. This card was installed in the same server as the TFC master node

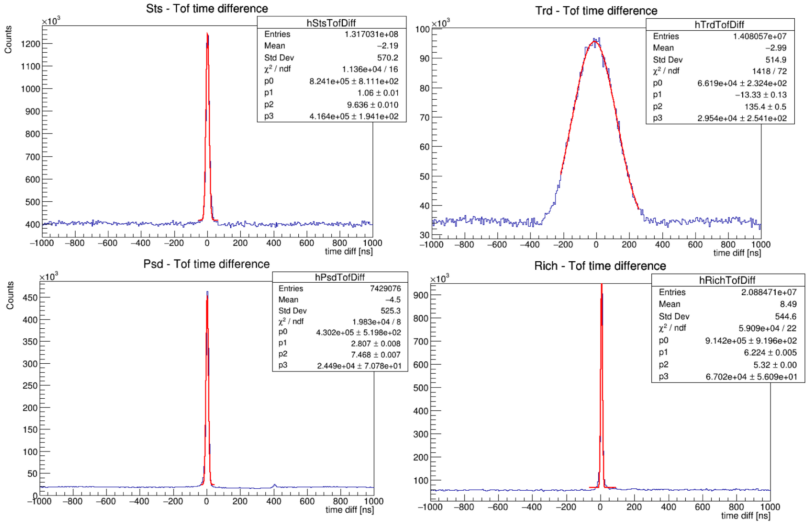


Figure 6.16: Time alignment between various subsystems in the Phase-0 CBM setup. Credit: CBM collaboration [30].

and delivered the global reference frequency over a coaxial cable. The TFC master was equipped with a mezzanine card that received this signal and routed it to the reference input of the local PLL. This configuration ensures coherence between the TFC time base and the accelerator time, which is critical for the coordinated operation of the experiment.

The deployed TFC prototype demonstrated successful synchronization of multiple detector subsystems during the first integrated beam campaign of the CBM experiment at the SIS18 facility at GSI in 2021. In this campaign, several detectors operated together in a common run. With the time offsets between detectors corrected during online data processing, stable synchronization between the 14 CRI boards has been achieved both within individual runs and across multiple runs [30]. Figure 6.16 presents the achieved time stability across the participating subsystems during one representative run.

Based on its successful performance, the prototype was adopted as the official Phase-0 version of the TFC system. Subsequent development efforts have focused on improving this system to meet the target performance requirements and provide a complete characterization. For clarity, the deployed prototype is referred to as TFC1, while the next performance-oriented iteration is referred to as TFC2.

6.9 Summary

The second feasibility study focuses on the hardware-oriented approach to the time synchronization across the network devices and the latency-deterministic message transport. Inspired by the fundamental principles described in the SyncE standard [44], this approach leverages the hardware features of the clocking and data transmission components to ensure the time consistency between the network nodes.

In the feasibility study, a lean architecture of a TFC system has been developed. In this architecture, the time synchronization is incorporated into the data transport protocol and components, with the time consistency in the network being achieved inherently with a minimal interference from the software or the user. The SyncE principle was mapped to the hardware features of the modern FPGA devices, resulting into a detailed concept of clock distribution over cascaded optical links. Together with a custom concept of the numeric global time, the clock distribution prototype provides the necessary functionality for the DAQ system of the CBM experiment to function as a whole with a consistent time base across its components.

The concept has been prototyped and verified in a minimal functional configuration supporting up to 47 endpoints. Having demonstrated significant resource efficiency in the feasibility analysis based on lab tests, the prototype has been incorporated into the preliminary implementation of the CBM experiment and used as the timing system in all subsequent integrated beam tests of the Phase-0 CBM setup. This marked a significant milestone in the development of the experiment, and

the hardware-oriented time distribution prototype has been adopted as the initial version of the TFC system and thereby laid the foundation for the final TFC architecture.

7 Final architecture: latency-optimized timing and fast control system (TFC2)

The functional prototype developed in the preliminary study, as described in the previous chapter, successfully provides timing distribution for the Phase-0 setup of the CBM experiment. Although this prototype confirms the feasibility of the overall system concept, the achieved synchronization accuracy remains significantly below the level required by the experiment. The subsequent development of the TFC system therefore concentrates on three main objectives: reducing transmission latency and its variance, improving synchronization accuracy, and increasing scalability to meet the demands of the full experimental setup. Building on the advantages of the approach identified in Chapter 6, the final TFC architecture is designed to fully exploit this method and to provide the Timing and Fast Control functionality required by the CBM experiment.

7.1 Latency-deterministic data transport

In the data transport paths of the TFC system, information is handled in three forms: a continuous serial bit stream, parallel multi-byte words, and, optionally, larger frames composed from these words. This layered organization allows for flexible handling and fast transmission of data, but also requires multiple stages of serialization and deserialization throughout the system.

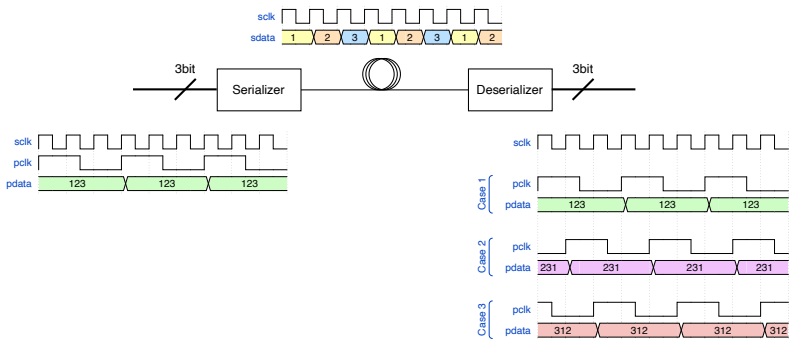


Figure 7.1: Potential data ambiguity after deserialization depending on the parallel clock phase. Numbers in the data represent bit positions in frames rather than bit values.

7.1.1 Analysis of the sources of latency variance

Each transition between serial and parallel representations introduces a new clock domain and potential delay uncertainty within each node. The general input-to-output delay stability of serializers and deserializers is directly determined by the stability of the phase alignment of the parallel and serial clocks. In addition, deserializers must be synchronized before they can correctly function. This is caused by the ambiguity of how the parallel data can be formed from the serial stream, presented in Figure 7.1. Typically, this is achieved by either delaying the serial data stream or by shifting the parallel clock, until the correct alignment is achieved. The former method adapts the input-to-output delay to the variable phase of the parallel clock, which ensures the data integrity but adds variance to the delay introduced by the component. As determinism of datapath delays is critical for distribution of time and fast control commands in the experiment, the approach with the controlled phase of the parallel clock is preferred in the TFC system.

Additional clock domains are also inherently created by the Tx and Rx datapaths, as described in Section 6.2. Although it is possible to pass data between these clock domains using FIFOs as it is done in TFC1, these buffers require external

control logic to ensure fixed input-to-output delay. In addition, data propagation through FIFOs introduce unnecessary latency of multiple clock cycles.

Alternatively to using FIFOs for handling Clock Domain Crossings (CDCs), the data can be passed between clock domains via regular registers. The approach is feasible when the data is passed between appropriately aligned syntonous clock domains, with a guarantee of metastability-free register operation. Although the approach requires control over the phase shift of at least one of the clocks involved in the CDC, it significantly reduces both the latency and the complexity of the datapaths. Simplifying the datapath control logic in turn reduces the resource utilization, which frees FPGA resources for higher-level application logic and improves scalability by making it possible to instantiate more data transport modules within a single TFC node.

Regardless of the method used to handle CDCs in the data transport datapaths, external clock alignment is required to ensure consistent downstream data latency. When this alignment is achieved, register-based CDCs offer significant advantages over the FIFO-based approach and are therefore preferred for use in the TFC system data transport.

If the TFC messages are organized into data frames, these frames must have a fixed size and they must never be fragmented. Such features would introduce unpredictable delays for all types of traffic, including the fast control messages, which are time-critical and must be propagated through the system as fast as possible. As the experiment operates at the base frequency of 40 MHz, the frame size can correspond to the number of bits that can be streamed over the serial line within one base clock period, or 25 ns. This way, the size of one TFC message is maximized without degrading system responsiveness.

7.1.2 Analysis of potential data transport solutions

The requirements for stable and predictable data transport latency discussed earlier closely match those needed for timing and control communication between FPGAs

and front-end electronics in particle physics experiments. In such high-throughput environments, these links are often implemented using custom-designed ASICs, particularly the GigaBit Transceiver chip (GBTx) and its successor, the low-power GigaBit Transceiver (lpGBT) [128, 129]. These chips were developed to provide deterministic-latency communication over optical links and are widely used for connecting FPGAs with radiation-hardened front-end ASICs. To support FPGA integration, both projects offer open-source IP cores for use in FPGA firmware [130, 131].

Because deterministic latency is a key design requirement of the TFC system, the architectures and communication protocols used in these IP cores were reviewed for possible use in the TFC data transport paths. The architecture of the lpGBT FPGA integration core (lpGBT-FPGA) is particularly tailored to meet the demands of the data acquisition in particle physics experiments. In the downstream direction, it can send trigger and control data at 2.56 Gb/s. The upstream link, used for readout, supports data rates of 5.12 Gb/s or 10.24 Gb/s. However, the asymmetric design means that the transmitter and receiver paths are handled differently, including differences in encoding and scrambling. This makes it not directly applicable the TFC system, which requires symmetric, bidirectional communication between FPGAs.

The earlier GBT FPGA integration core (GBT-FPGA), on the other hand, is an integrated core that includes similar features to lpGBT-FPGA, but is designed for a symmetric bidirectional data transmission at 4.8 GB/s. The peak-to-peak uncertainty of the clock and data phase between power-ups has been reported to be 100 ps [130]. These properties make GBT-FPGA highly suitable for latency-optimized communication between FPGA devices, therefore this IP core is further adopted for data transport in the TFC system.

The latency-optimized architecture of the GBT-FPGA IP core has been developed specifically addressing potential points of latency uncertainty, therefore it is compliant with the data transport needs of the TFC system. For example, it uses register-based CDCs and dedicated alignment logic in the deserializer stage to ensure consistent timing. The GBT protocol is based on 120-bit wide frames,

which are split into 40-bit words before being serialized by the Gigabit Transceivers built into Xilinx FPGAs.

To support reliable transmission over optical links, the core uses a scrambler to produce a pseudorandom bit stream. This helps reduce phase bias at the receiver CDR circuit and improves protocol efficiency compared to traditional methods like 8b/10b encoding by supporting shorter headers with larger frame size. As the GBT-FPGA core has been originally designed for high-bandwidth data transmission in the radioactive environment of particle physics experiments, the core also supports optional forward error correction (FEC) based on the Reed-Solomon coding scheme. This makes the data transport immune to occasional bit flips, which is highly beneficial in a mission-critical system like TFC.

As a data frame is transmitted over an optical link served by two GBT-FPGA cores on its ends, it passes two stages of serialization on transmission, and two stages of deserialization on the receiving end (Figure 7.3). In the transmission datapath, manual clock phase control is required to ensure metastability-free transfer of the data between three clock domains: 40 MHz frame clock, 120 MHz word clock and the high-speed 4.8 GHz serial clock. The CDC between the word clock and the serial clock is handled internally by the Xilinx transceiver logic together with the serialization itself. Therefore, the internally produced and aligned word clock is available to the system designer and is not subject for phase control. At the same time, the transition from the frame clock to word clock domains must be handled manually to ensure successful frame serialization without data corruption, as shown in Figure 7.2. For this reason, the phase of the frame clock must be adjusted by external calibration logic as part of the link initialization procedure.

On the receiving end, the serial bit stream is similarly deserialized internally in the Xilinx transceivers. Thus, the received word clock is also internally produced and aligned with the high-speed serial clock, providing the word clock to the designer with a constant phase. Similarly to the transmitter frame clock, the external receiver calibration logic must align the frame clock to avoid data corruption due to metastability on the CDC. The frame clock phase calibration logic is not part

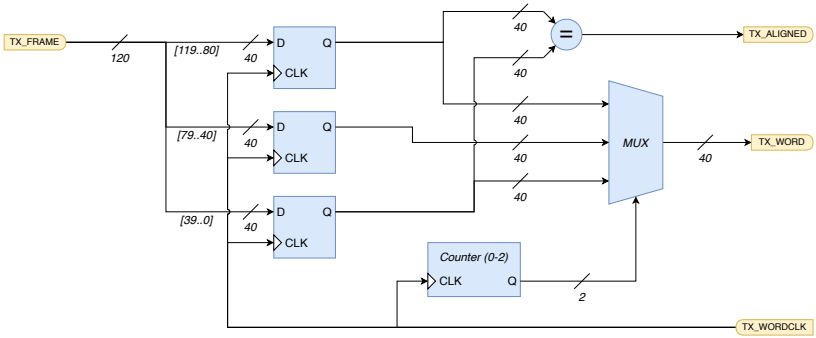


Figure 7.2: Register-based CDC between the frame clock domain and the word clock domains in the transmitter datapath.

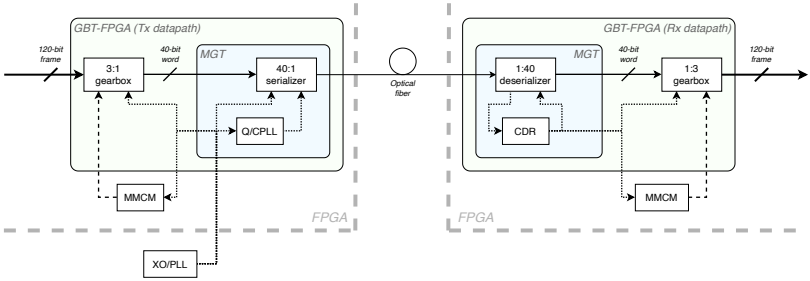


Figure 7.3: Clock generation and serialization/deserialization stages in data transmission between GBT-FPGA cores.

of the GBT-FPGA core and must be handled outside of the core as part of the FPGA firmware architecture.

In addition to the frame clock phase calibration in the receiver datapath, the alignment of the data during deserialization must be performed during link initialization. The core includes a frame aligner to facilitate this process. Using the bit slip mechanism of the Xilinx transceivers, the frame aligner monitors the presence of the static frame header at the right position of the frame and shifts the serial bit stream accordingly, until the header is detected in the correct position in the frame. The register-based CDC in the receiver datapath and the header flag generation

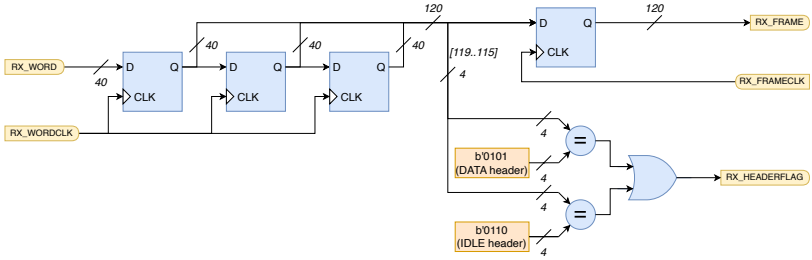


Figure 7.4: Register-based CDC in the receiver datapath and the header flag generation logic.

logic are shown in Figure 7.4. With this method, both deserialization stages are aligned in one procedure, without the need to align the data in each deserializer individually, thus simplifying and speeding up link initialization.

7.1.3 Clock and data alignment

Although the Rx and Tx word clocks in the GBT-FPGA core are syntonous, they are generated independently within the Gigabit Transceiver IP core. Because their phases cannot be directly controlled, they introduce clock domains with a constant but unpredictable phase offset relative to each other. In contrast, the frame clocks in both directions are derived from the transceiver reference clock using Xilinx Mixed-Mode Clock Managers (MMCMs), which allow fine-grained phase alignment with a step resolution of approximately 18.6 ps. Since both Tx and Rx frame clocks must ultimately match the system clock domain, only one of them can be phase-aligned with the corresponding word clock in a latency-deterministic manner. This constraint requires that the same frame clock be used as the system clock and the frame clock in the opposite datapath, where data exchange with the word clock domain must be handled through a FIFO. As a result, only one direction of the GBT-FPGA link can be configured for latency-optimized operation at a time. Given the architecture and timing requirements of the TFC system, the downstream direction must operate with fixed latency, making it the candidate for deterministic optimization.

While FIFO-based datapaths do not require phase alignment and are relatively straightforward to configure, latency-optimized datapaths require more careful handling to ensure data integrity. In the case of the GBT-FPGA core configured with a latency-optimized Tx datapath, the serializers are already initialized in a coarsely aligned state. However, precise phase alignment using the MMCM is necessary to ensure metastability-free data transfer through the gearbox. The alignment status can be verified using the `TX_PHALIGNED_o` output signal provided by the GBT-FPGA core. On the receive side, the situation is more complex: the Rx gearbox can lock to one of three possible phases of the incoming serial clock, which leads to ambiguity in data alignment, as illustrated in Figure 7.1. To resolve this, the Rx gearbox must be reset repeatedly while monitoring the position of the frame header using the `MGT_HEADERFLAG_o` output. When the header is observed at its valid position, correct alignment of the Rx gearbox is confirmed. Alternatively, the Rx gearbox alignment can also be fine-tuned using the same MMCM-based phase shifting technique as used for the Tx path.

7.2 Protocol definition

The protocol supported by the GBT-FPGA framework transmits data in fixed-length frames of 120 bits. Depending on the selected configuration of the GBT core, these frames can carry either 116 or 84 bits of payload data [132]. The four MSBs of each frame are permanently reserved for a static header, which is essential for the alignment of receiver deserializers. The remaining 116 bits are available for user data and can be used in two modes. In the WideBus configuration, all 116 bits are dedicated to payload. Alternatively, the GBT frame configuration protects the data with FEC based on the Reed-Solomon coding scheme. This FEC implementation requires 32 bits, leaving 84 bits for the actual payload.

Given the mission-critical nature of the TFC system within the CBM experiment, the use of forward error correction is advantageous. It provides strong protection against bit errors, ensuring reliable communication of timing information and fast control commands, and effectively making the system robust to bit flips in the

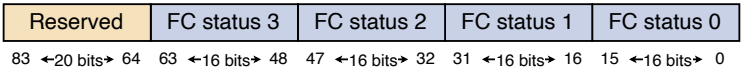


Figure 7.5: Format of an upstream TFC frame.

transmission medium. Therefore, the GBT-FPGA core is used in the GBT frame configuration for data transport in the TFC system.

The upstream communication within the TFC system consists exclusively of status information gathered from the CRI boards. Since the exact parameters to be controlled by the fast control system are still under definition, the data format for fast control messages must be kept generic and flexible to accommodate future changes. However, it is already clear that upstream data must be aggregated efficiently to scale from approximately 200 CRI nodes to the central TFC master. In this context, each TFC submaster is responsible for receiving status information from up to 47 downstream-facing links and compressing or processing it in a way that enables forwarding via a single upstream link, resulting in a link aggregation ratio of 47:1.

The specific aggregation strategy depends on the nature of the controlled parameters and may differ between various types of fast control messages. As a result, intermediate message processing must be carried out by the submaster nodes. To support this, the format of upstream TFC frames defines separate bit fields for multiple categories of status messages, allowing them to be carried in parallel. This structure enables the submaster to process multiple types of status information concurrently, which is crucial for achieving fast and responsive control.

Moreover, in scenarios where only a single experimental parameter is controlled by the fast control system, the parallel structure of the frame format allows for a reduction of the effective aggregation factor. This helps minimize the loss of status information during aggregation and improves the overall quality and granularity of control. The resulting structure of the upstream TFC frame is illustrated in Figure 7.5.

Unlike upstream traffic, downstream communication in the TFC system does not require data aggregation. Each incoming frame from the central master is instead broadcast by the submasters to all 47 downstream-facing links. These frames must include the global time counter value, generated by the master node, as well as the fast control commands. For technical reasons, the global time in the experiment is represented as a 64-bit counter running at the 40 MHz experiment clock. This counter overflows only once every ∼7312 years, making the full 64-bit width redundant for the expected lifetime of the experiment. Therefore, the width of the transmitted time value has been analyzed in the context of counter overflow periods and the realistic operational timespan of the experiment.

The analysis, summarized in Table 7.1, shows that bits 58 through 63 of the counter will not increment over the experiment lifetime and can be omitted from transmission. Consequently, only the lower 57 bits of the time counter are included in the downstream frame and the missing bits are padded with zeros at each receiving node.

Counter width [bits]	Wrap period [years]
54	14,28
55	28,56
56	57,12
57	114,25
58	228,49
59	456,99
60	913,97
61	1827,95
62	3655,89
63	7311,78

Table 7.1: Estimated wrap period of time counters with various widths at the 40 MHz rate.

With 57 bits dedicated to the time counter, the remaining frame payload can be fully used for fast control commands. The combined time and control information

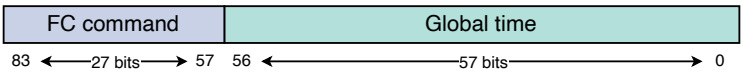


Figure 7.6: Format of a downstream TFC frame.

is transmitted downstream to all CRI boards on every cycle of the 40 MHz global clock. This high-frequency update allows each TFC node to detect local time inconsistencies immediately, thus enhancing system responsiveness. The final format of a downstream TFC frame is shown in Figure 7.6.

7.3 System architecture

In the TFC2 architecture, the general concept of the earlier TFC1 system has been adapted to support latency-deterministic data transport using the GBT-FPGA framework. The architecture is specifically designed to provide precise control and alignment of clocks to ensure fixed and predictable latency in data transmission, as required for correct and reliable operation within the CBM experiment. To meet the scalability demands of the Day-1 CBM setup, which will involve up to 200 endpoint CRI boards, the system architecture has been extended to include a newly designed submaster node.

The updated gateware architecture of TFC2, including the submaster node, is illustrated in Figure 7.7. As with the TFC1 system described in Section 6.6, the TFC2 nodes are installed in PCIe slots of commercial rack-mounted servers. A PCIe-to-Wishbone bridge provides register access from the host server to the internal peripherals of the FPGA-based node design. This configuration enables a hardware-software codesign approach in which the server-based software handles initialization, runtime control, and monitoring, while the FPGA handles time-critical functions.

Data transport within the TFC2 system is implemented using GBT-FPGA cores, encapsulated by the TFC link module. This module is instantiated once per

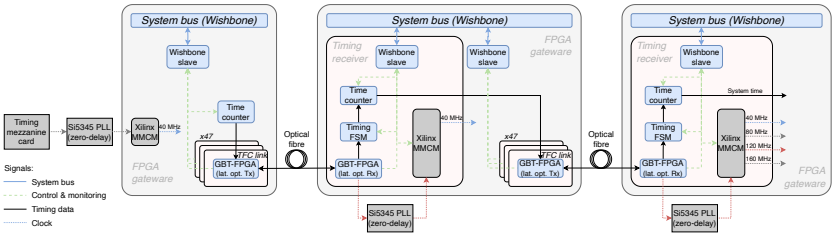


Figure 7.7: Gateware architecture of the TFC2 nodes.

optical connection and is configured differently for upstream- and downstream-facing directions, with the fixed-latency configuration in the downstream direction across the TFC system. In nodes with upstream-facing links, the module provides a recovered clock that is reused as the local system clock. Downstream-facing links, in contrast, derive their clocks from this local clock and align their transmitters accordingly. To ensure stable and predictable latency across all links, every component in the clock paths is configured either in a zero-delay mode or with a fixed input-to-output delay.

Each TFC node contains a time counter module that defines the local time value. This module can be preset with an arbitrary value if required. In the master node, the counter is coarsely aligned to standard UTC time during initialization via the Wishbone interface. This ensures that detector data can be time-stamped with traceable absolute times when archived. For nodes with downstream-facing links, the output of the time counter is directly connected to bits 0-56 of the transmitter data input, following the TFC protocol format described in Section 7.2.

In contrast, submaster and endpoint nodes containing upstream-facing links require a more dynamic approach. Their time counters are managed by a dedicated timing Finite State Machine (FSM), which controls the synchronization process. The FSM begins in the IDLE state, waiting for a command to initiate synchronization with the time received via the upstream link. Once all upstream nodes are initialized and the received time is deemed valid, the FSM is triggered via the register interface to initiate synchronization and transition into the SYNC state. Here, it calculates the difference between the received and local time values and

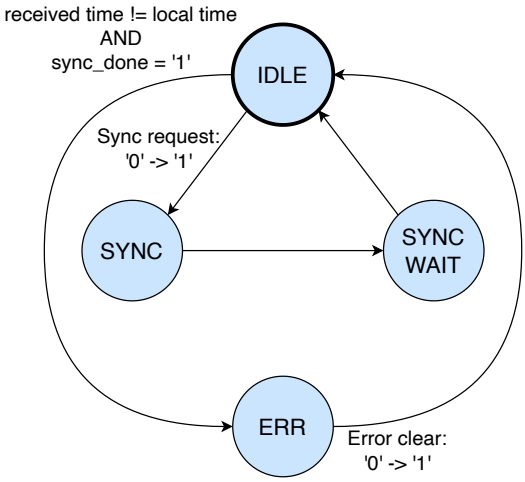


Figure 7.8: State diagram of the timing FSM module.

applies this as an adjustment to the local counter. After a brief wait period in the SYNC WAIT state to allow the update to take effect, the FSM returns to the IDLE state. From this point on, the FSM operates in a monitoring mode, continuously comparing received and local time values. In case a discrepancy is detected, the FSM raises a register flag to alert the operator. Once the link failure has been resolved, the operator explicitly issues a command to the TFC node to recover from the error and restart the time synchronization. This approach prevents automatic fault recovery by design, since even rare failures in the TFC system are unacceptable as they would significantly degrade the quality of the measurements collected in the experiment.

Among all node types in the TFC2 system, the submaster features the most complex gateware. It combines functionalities of both master and endpoint nodes. Similar to the endpoint design, the submaster derives its local clock from the recovered clock of the upstream-facing link, which must be aligned with the receiver datapath as described in Section 7.1.3. At the same time, this local clock must ensure

data integrity across multiple downstream-facing transmitter datapaths. The dual requirement to align the system clock with both receiver and transmitter datapaths is unique to the submaster and imposes specific constraints on clock alignment procedures.

To satisfy these constraints, clock alignment is carried out in two phases: first, alignment is performed for the receiver datapath of the upstream link to enable clock recovery. Afterwards, the transmitter datapaths of the downstream links are adjusted, since they are initialized already in a coarsely aligned state. However, fine clock alignment is required for correct operation of all the fixed-latency datapaths in a submaster node, which is analyzed in detail in the following section.

7.4 Link calibration

Although coarse alignment of the transmitter and receiver gearboxes generally functions reliably, there exist edge cases in which this approach proves insufficient. The gearbox alignment indicators, such as *RX_HEADERFLAG* and *TX_ALIGNED*, only confirm that data is being captured within the valid stability window. However, they do not indicate where exactly within this window the capture occurs. If the clock edge used for data launch or capture lies too close to the boundary of the window, even minor timing deviations caused by thermal drift or jitter may result in incorrect data sampling and hence compromise data integrity across the gearbox. This risk is illustrated in Figure 7.9.

Given the mission-critical role of the TFC system in the CBM experiment, precise phase adjustment of the output clock is required to ensure its long-term reliability. This guarantees that data is consistently captured with a sufficient phase margin by the parallel clock (PCLK), thus maintaining data integrity in the datapath. Fine calibration of the system clock and gearboxes is therefore necessary.

Before achieving optimal alignment of PCLK relative to the serial clock (SCLK), the current phase offset must be characterized. This is done using the dynamic phase shift capability of the MMCM component that generates the system clock.

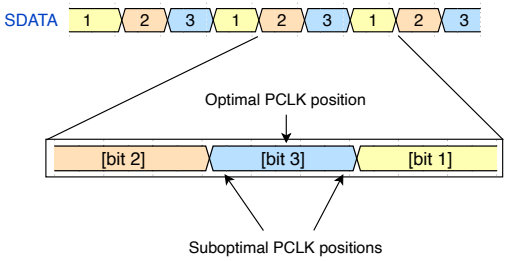


Figure 7.9: Optimal alignment of the parallel clock relative to the serial clock.

This clock, which acts as PCLK, can be shifted in 1344 fine steps, each corresponding to approximately 18.6 ps, over a 25 ns clock period. By scanning these phase steps and evaluating the gearbox alignment flags at each step, the range of phase shifts where the data is stable can be identified. This range is referred to in this work as the "data integrity window".

To demonstrate the feasibility of this approach, a phase scan was performed on a TFC master node across all 47 TFC links. The scan used a phase increment of 12 fine phase shift steps (223 ps), and to accommodate local instabilities at the edges of the integrity window, the *TX_ALIGNED* flag was sampled 500 times per step. The results are shown in Figure 7.10, where the x-axis corresponds to phase shift increments (0-1344) and the y-axis represents the 47 TFC links. The cell color encodes the number of successful alignment reads at the respective phase, with lighter shades indicating unstable alignment across the corresponding gearbox at that phase shift value.

As shown in Figure 7.10, most gearboxes are technically functional without calibration, yet many operate dangerously close to the boundary of the data integrity window. Furthermore, links 15 and 32 demonstrate failed alignment, with data not being captured correctly by the gearboxes.

To address this, an automated link calibration routine was developed. The routine begins by identifying the boundaries of the data integrity windows for all links via phase scanning. The overlap of all stable windows is then calculated, and

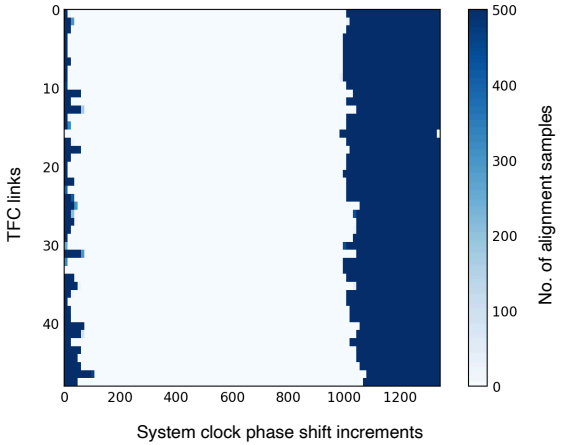


Figure 7.10: Results of the phase scan in an uncalibrated master node.

the center of this overlap is selected as the optimal phase shift for the system clock. Applying this shift maximizes the timing margin across all links. The calibrated result, shown in Figure 7.11, confirms reliable data capture across all 47 transmitters after calibration.

The submaster node introduces an additional layer of complexity, as it must also ensure data integrity in the receiver datapath of its upstream-facing link. In addition to the fine alignment problem, the receiver path introduces clock division uncertainty, as previously discussed in Section 7.1.1 and shown in Figure 7.1. The three possible receiver alignment scenarios are illustrated in Figure 7.12. Note that link 36 is excluded in the plots as it is electrically rerouted to the upstream-facing interface.

Because the receiver and transmitter clock domains are inherently independent in the submaster, their data integrity windows are not naturally aligned. This misalignment means that, after coarse alignment, one edge of the receiver integrity window may lie deep within the transmitter window, leading to potential integrity issues. The goal of submaster calibration is thus to maximize the overlap between

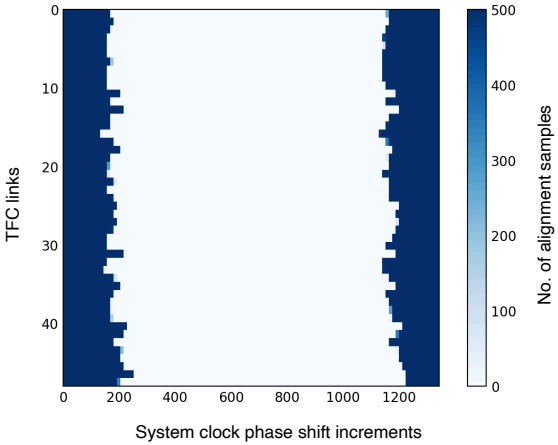


Figure 7.11: Results of the phase scan in a calibrated master node.

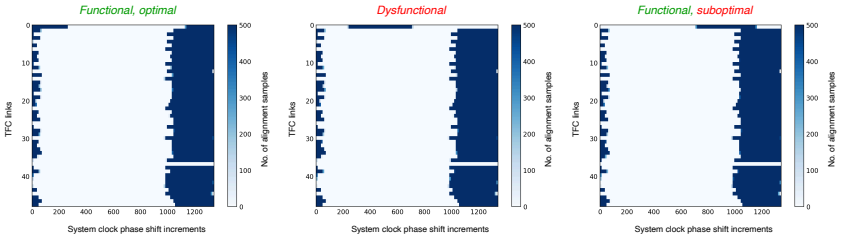


Figure 7.12: Three different possibilities of Rx alignment in the submaster node.

the two windows and select a phase shift that centers the system clock edge within this intersection.

The submaster calibration process first determines the data integrity windows of all links, irrespective of the receiver coarse alignment. Based on these measurements, all potential receiver window positions are computed. A candidate phase shift is selected to maximize the overlap between receiver and transmitter windows. The system clock is then adjusted accordingly, and the integrity windows are scanned again for verification. If necessary, the clock phase is fine-tuned to further improve alignment. The final result is a system clock configured for optimal phase,

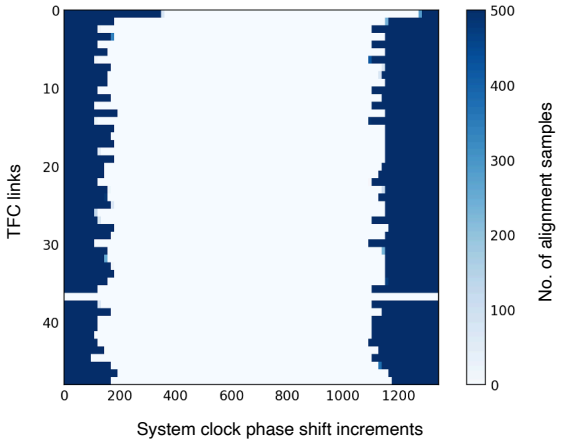


Figure 7.13: Results of the phase scan in a calibrated submaster node.

ensuring reliable, fixed-latency operation across all links. The calibrated result for a submaster node is shown in Figure 7.13.

Compared to the submaster and master nodes, the endpoint node calibration is relatively straightforward and follows the same approach used for transmitter calibration in the master. For all three TFC node types, the clock calibration routine needs to be performed only once per gateware implementation, since the optimal phase shift depends solely on the internal layout of the FPGA design. However, the receiver datapaths must be re-aligned each time a node is initialized, to ensure consistency and correctness after each power cycle.

The calibration procedure described in this section ensures the reliable operation of the latency-deterministic multi-gigabit communication paths, which are susceptible to metastability-related errors. The work described above therefore constitutes a key prerequisite for the reliable operation of the TFC system, which is based on bidirectional links with predictable latency.

7.5 Summary

Building up on the initial prototype of the TFC system described in Chapter 6, the clocking scheme and the datapaths of the data transport infrastructure have been further optimized. The GBT-FPGA core [130] has been incorporated in the TFC architecture to handle the data transport functions. Originally meant for interfacing with a custom ASIC, the core has been adapted for a non-typical use case of latency-deterministic FPGA-to-FPGA communication in the current work. While the latency-deterministic transmission path falls into the usual usage pattern of the core, the components of the receiver datapath had to be modified for latency-deterministic performance. Since the latency-deterministic data paths are designed without FIFOs for clock domain crossing, they are inherently susceptible to metastability-related failures. Therefore, a calibration technique was developed to ensure a deterministic phase relationship between the corresponding clocks, thereby enabling reliable message transmission between network nodes with fixed latency, as required by the DAQ system of the CBM experiment.

As a result, an improved architecture of the scalable synchronized system with a predictable message transport latency has been developed. The unique feature of the architecture is that it fully relies on point-to-point connections between nodes with the high synchronization accuracy and latency determinism and provides the access to the data in all nodes, allowing for domain-based control of the DAQ system. To ensure data and latency integrity in the intermediate network nodes, a novel calibration algorithm has been developed in this work. Thus, the resulting architecture combines the flexibility of the general-purpose networks, such as TSN and White Rabbit, and the high determinism of the hardware-based time distribution systems.

8 System characterization

The TFC system is a common component of the experiment that ensures time consistency across all data readout electronics and orchestrates the overall data acquisition process. Any failure of this system to perform its functions according to the specified requirements would immediately lead to a significant degradation in the performance of the experimental setup, eventually preventing it from achieving its intended physics goals. For this reason, it is crucial to establish a solid and reliable characterization framework for the TFC system. Such characterization must be sufficiently thorough to confirm that the system consistently delivers the performance required for the Day-1 CBM configuration, as specified in Section 4.1.

8.1 Characterization goals

The primary goal of the TFC system characterization is to confirm its ability to provide a stable time base to all CRI boards in the form of a global 40 MHz clock and a 64-bit time counter, ensuring that the variance between boards does not exceed 200 ps. In addition, the system must guarantee low-latency transport of fast control messages between the CRI boards and the TFC master, while maintaining this performance even when at least 200 CRI boards are active in the setup. System scalability is inherently guaranteed by the hierarchical topology of the TFC architecture. With a link aggregation ratio of 1:47 in the submaster nodes, the overall performance of the full-scale setup can be described across three hierarchical layers: one TFC master, one layer of submaster nodes, and a final layer of CRI endpoints. The maximum number of CRI boards that can theoretically be

served in this configuration can be calculated as $N_{MAX} = N_{links}^2 = 47^2 = 2209$, which significantly exceeds the requirements of the initial CBM setup.

Since the scalability of the system is thus guaranteed by design, the focus of characterization lies primarily on verifying the latency performance of timing distribution and fast control message transport. Both of these functions rely on the same downstream data transport, which means their performance is directly linked and determined by the latency of the downstream path and its variance. For this reason, the characterization framework has been developed around precise measurement of the latency stability of the data transport links, as correctly functioning message delivery is the core prerequisite for achieving stable and accurate TFC operation in the experiment.

To assess system performance over extended operating periods, the characterization procedure must be automated to enable the collection of large measurement samples. Furthermore, it must include repeated power cycling of the TFC nodes to confirm that link latency remains stable after complete system restarts. Consequently, two parameters are central to the characterization process: the stability of link latency over long periods of continuous operation, and its reproducibility across successive power cycles.

8.2 Method definition

To achieve accurate results, the characterization of the TFC system must be carried out through direct measurements in real hardware, thereby including all potential sources of variance inherent to the operating environment. This approach follows the concept of hardware-in-the-loop testing, which ensures that the results are representative of realistic system conditions. For this purpose, a minimal TFC setup consisting of two interconnected nodes is used, emulating a single TFC link. The measurement procedure is based on sending a predefined message from one node to another and observing the resulting transport latency. To generate measurable observables, electric pulses are issued at the two key points in time:

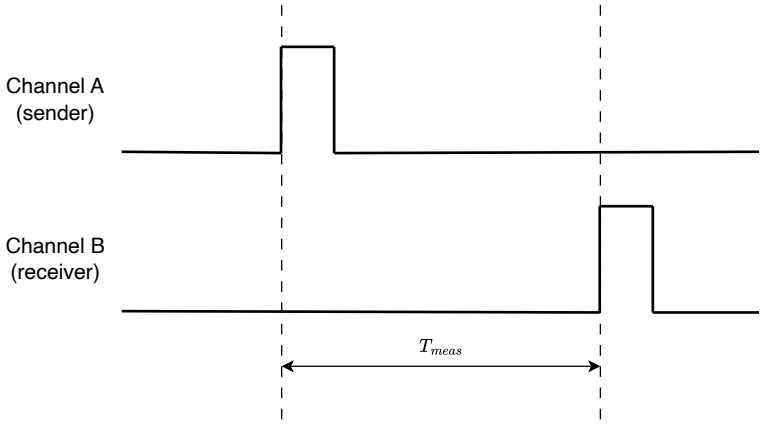


Figure 8.1: The concept of measuring data transport latency through a time distance between electric pulses.

once at the sender when the message is transmitted, and once at the receiver upon its arrival.

The time distance between these pulses is measured using a dedicated lab instrument, such as an oscilloscope or a time interval counter, as illustrated in Figure 8.1, and represents one instance of the data transport latency between two TFC nodes. The emission of pulses can in principle be automated or manually triggered. In this work, the manual triggering approach has been selected in order to retain full control over the measurement process. While this sacrifices some measurement speed, it ensures that every generated pulse is sampled, thereby avoiding any potential selection bias introduced by automated triggering mechanisms.

Since the TFC system must be verified over extended operational periods, including scenarios with system reboots, the complete test loop is automated at the level of the characterization setup. Automation encompasses message generation and pulse emission, acquisition of measurements from the oscilloscope or time interval counter, and power control of both the CRI boards and the servers in which they are mounted. A central test control software orchestrates this process,

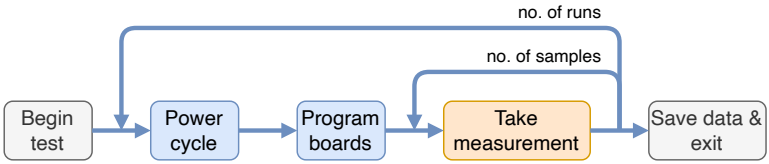


Figure 8.2: Algorithm of the automated characterization of the TFC system.

integrating all necessary functions. This enables the collection of large volumes of measurement data for subsequent analysis, yielding statistically robust insight into the performance of the system over time. Importantly, the hardware under test is installed in the same environment as the CBM DAQ system, ensuring that the obtained results reflect realistic operating conditions.

The complete characterization procedure is summarized in Figure 8.2. At the beginning of each session, both sender and receiver nodes are power-cycled to ensure that the servers start from a clean and deterministic state. The FPGA boards are then programmed with the target gateware and configured, finalizing the necessary configuration for measurements. Once the hardware is prepared, measurements are taken sequentially: a register command triggers the transmission of a test message, the resulting pulse interval is recorded by the measurement instrument, and the result is read back into the central controller. This process is repeated many times to gather sufficient statistics on the latency variance during a single operational run.

After each run, the hardware under test is power-cycled and reconfigured, including the servers, to repeat the procedure under fresh initialization conditions. This repetition ensures that the data collected across multiple runs captures not only the inherent link latency variance during operation, but also the variance introduced by system restarts. The layered methodology of continuous sampling within runs, combined with controlled reinitializations across runs, enables a clear distinction between these two sources of variance and provides a comprehensive characterization of the TFC system.

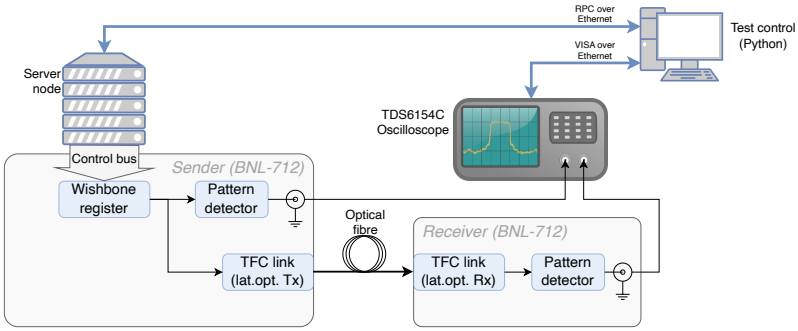


Figure 8.3: Architecture of the characterization setup of the TFC data transport.

8.3 Test setup

To ensure accurate and representative characterization results, the test setup was built on the same hardware platform as that used in the TFC system. The overall architecture of the characterization setup is illustrated in Figure 8.3.

8.3.1 Hardware platform

The setup consists of two BNL-712 boards mounted in separate server nodes and interconnected with an optical fiber. Individually connected with Digilent programming cables to the test control machine, both FPGAs are programmed with the standard TFC gateware. One bit normally reserved for fast control commands is repurposed to trigger measurement pulses simultaneously on the sender and receiver boards. The same internal signal that drives this bit is also routed directly to an output pin of the FPGA, ensuring that the generated measurement pulses accurately reflect the temporal properties of the internal signal path without being influenced by additional logic. To bring the pulses off the FPGA board, one of the available U.FL microcoax connectors is used on each board. Through a U.FL-to-SMA converter, the signal is carried via a 2 m SMA-to-SMA cable and then connected to the oscilloscope using an SMA-to-BNC adapter. The

cables used for signal connectivity in the setup have a minimum bandwidth of 3 GHz, corresponding to a signal rise time of approximately 120 ps. Although the bandwidth of these cables represents the limiting factor in the measurement chain, they are sufficient for the intended characterization, as most of the expected latency variations in the TFC links are confined to frequency ranges below 1 GHz.

For capturing and analyzing the pulses, a Tektronix TDS6154C oscilloscope is used. This instrument provides four input channels with an analog bandwidth of 20 GHz and a maximum sampling rate of 40 GSa/s, which is more than adequate for the precision required in this work. A particularly important feature of this oscilloscope for the present study is its comprehensive remote-control functionality. The TDS6154C supports remote desktop access as well as the Virtual Instrument Software Architecture (VISA) protocol, which is widely used for laboratory automation. VISA supports communication over Ethernet, USB, or the legacy GPIB interface, and enables interaction with the instrument through text-based queries and responses. Within the characterization setup, the most relevant VISA commands are those related to cross-channel edge-to-edge delay measurements, trigger configuration, and the readout of recorded measurement data. The remote desktop interface, in turn, provides a practical way to adjust oscilloscope settings during initial tuning, after which the configuration can be exported and consistently redeployed via VISA. An example of a single measurement pulse captured with the TDS6154C oscilloscope is shown in Figure 8.4.

In each measurement run, the time shift between the transmitted and received pulses is recorded, along with the statistical properties of the complete set of measurements. This allows the dynamic behavior of link latency to be analyzed in detail, distinguishing between random fluctuations and gradual latency drifts. For every run, the full distribution of individual measurements is stored, and the results are summarized by statistical indicators such as the mean latency value, minimum and maximum latencies, and the standard deviation. These parameters are essential for analyzing the statistical properties of the link latency and, in particular, for identifying variance components introduced by system restarts.

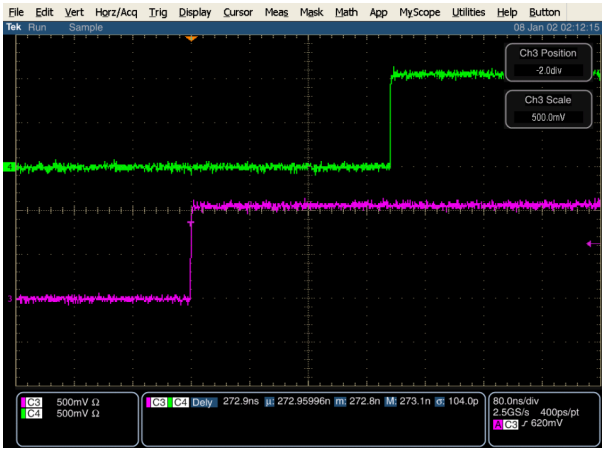


Figure 8.4: An example of a measurement pulse using the Tektronix TDS6154C oscilloscope.

8.3.2 Test control software

In order to reliably implement the characterization algorithm derived in Section 8.2 on a complex hardware setup, a modular control software framework has been developed. The framework has been implemented in Python, due to the wide availability of scientific and hardware control libraries and the straightforward integration of Python with existing automation and data analysis workflows.

To ensure a structured and extendable implementation, the control of different hardware components has been encapsulated in separate Python modules. This abstraction allows the high-level characterization algorithms to remain independent from the low-level control interfaces of the test hardware. The result is a highly parameterizable and modular software environment in which new test procedures can be implemented without much modification needed in the existing infrastructure. The overall architecture of the test control software is illustrated in Figure 8.5.

The test software makes extensive use of the DCA infrastructure deployed in the experiment. DCA provides a software abstraction layer for register-level access

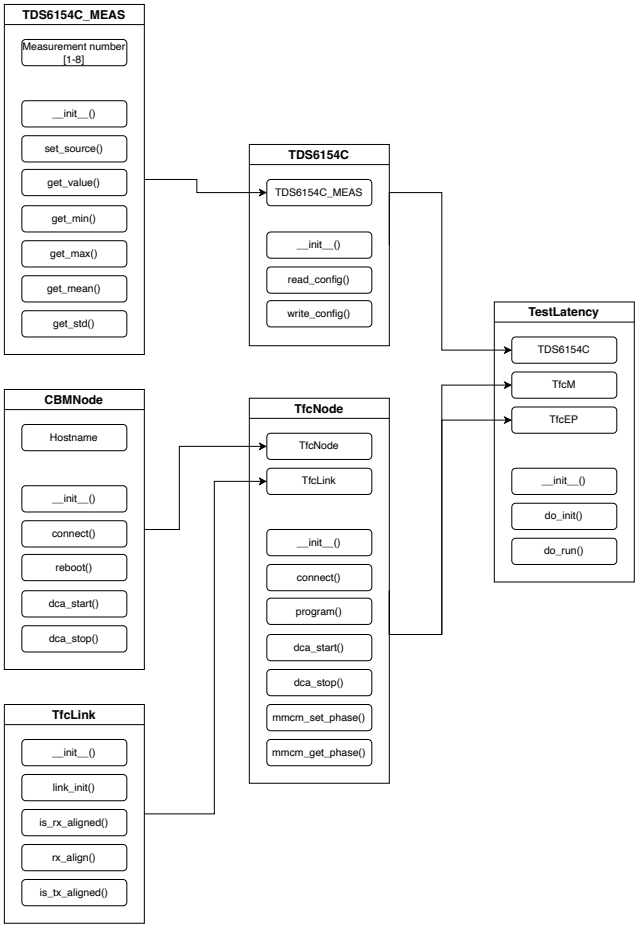


Figure 8.5: Modular architecture of the test control software.

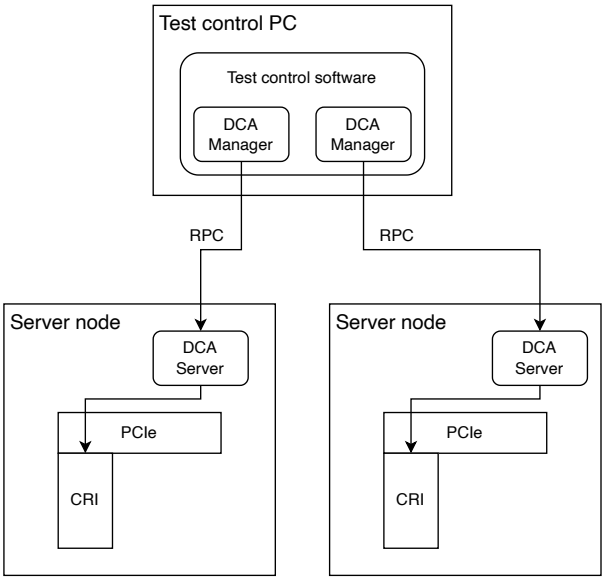


Figure 8.6: Remote connection between the test control software and server nodes using the DCA infrastructure.

and hardware control, with every server node hosting CRI boards running a local DCA manager instance. The test control software interacts with these instances remotely via an RPC interface, enabling transparent hardware control across the distributed setup as shown in Figure 8.6. Although implemented in C++, DCA offers Python bindings, which significantly increase flexibility and makes it an ideal component of the automated, closed-loop test framework.

PC node

The `CBMNode` class models the physical server nodes in which the FPGA boards are mounted. In practice, these servers are remote from the central control machine, and multiple such nodes may be included in a single test setup. The

abstraction maintains a connection to the DCA manager serving all boards in the corresponding server.

To support the automated restart procedures required by the characterization method, the PC node abstraction also implements power control functionality. The commercial off-the-shelf rack-mounted servers used in the CBM DAQ system feature board management controllers (BMCs) with standard remote interfaces, such as Intelligent Platform Management Interface (IPMI) over Ethernet. Leveraging this capability, the control framework issues power commands through a local IPMI tool that communicates with the BMC to switch the remote node on or off. During initialization, the PC node abstraction also starts the DCA manager on the server via SSH, thus fully preparing the machine for participation in the characterization runs.

Oscilloscope

A critical element of the setup is the software abstraction of the Tektronix TDS6154C oscilloscope. This abstraction is implemented using the PyVISA library, which provides a convenient Python interface to instruments supporting the VISA protocol. The `TDS6154C` class instantiates configurable measurement routines, with options for selecting input channels and measurement types. While the present work focuses on cross-channel edge-to-edge delay measurements, the design can be readily extended to other observables such as pulse widths or amplitudes, following the oscilloscope programming manual.

Each measurement instance retrieves both instantaneous values and statistical summaries for the current batch, which are stored for later analysis. The VISA interface also exposes the current sample count, allowing the test framework to monitor measurement progress and maintain full runtime control. Up to eight concurrent measurement routines can be configured per test, enabling future extension of the framework to more comprehensive characterizations.

Common TFC node controls

The unified `TfcNode` class provides board-level control functionality for TFC nodes. Each instance is tied to a corresponding PC node abstraction and is associated with a specific PCIe slot. For automated restart procedures, the class provides methods for programming the FPGA with a given gateware image via Tcl commands sent to the hardware manager on the central control machine, which is connected to the FPGA using a Digilent programming cable.

The class also manages DCA configuration for register access, enabling runtime configuration of the node. In the context of TFC characterization, one of its most important functions is the management of the local clock generator, including the dynamic phase shift features required for precise alignment. This capability is directly linked to the clock alignment and calibration mechanisms discussed in Sections 7.1.3 and 7.4.

TFC link

Since the number of links instantiated per node varies, the `TfcLink` class encapsulates the control and diagnostics of each link individually. This module manages the GBT-FPGA instance that performs the actual data transport and provides methods for link initialization, configuration, and retrieval of diagnostic information. The module also implements link alignment as discussed in sections 7.1.3 and 7.4 and access to the necessary alignment status flags.

Configurable tests

The central part of the test control software is formed by the configurable tests, which implement the high-level characterization algorithms by using the previously defined hardware abstraction modules. Each test is implemented as an individual Python class, defining the flow of the measurement procedure while also ensuring consistent logging of the test progress and structured storage of the results. The

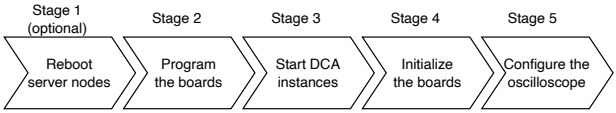


Figure 8.7: Initialization sequence of the tests.

design follows a predefined structure that reflects the general concept of the characterization approach but remains flexible enough to allow extension to new tests in the future.

Every test is split into two phases: initialization (`do_init`) and execution (`do_run`). The initialization phase is carried out in the sequence shown in Figure 8.7. In an optional first step, the server nodes can be rebooted to guarantee a clean start of the hardware setup. This is followed by programming the FPGA boards with the specified gateware and launching the DCA instance on the corresponding machines, thus connecting the programmed boards to the common control infrastructure. Once the hardware has been prepared in this way, the boards are configured with the parameters required for the specific test, such as initialization of TFC links. The initialization concludes with configuring the oscilloscope, ensuring that the measurement setup is fully operational before the test run begins.

The `do_run` phase then implements the course of the test itself and may rely on auxiliary routines defined separately in the class. For latency measurements, this includes functions to trigger the transmission of test messages, perform individual delay measurements, and collect batches of results to form a complete test run. At the highest level, the `do_run` routine executes a predefined number of runs, orchestrating the restart and re-initialization of the test hardware between them in accordance with the procedure outlined in Figure 8.2. This structured approach ensures both reproducibility and scalability of the characterization process.

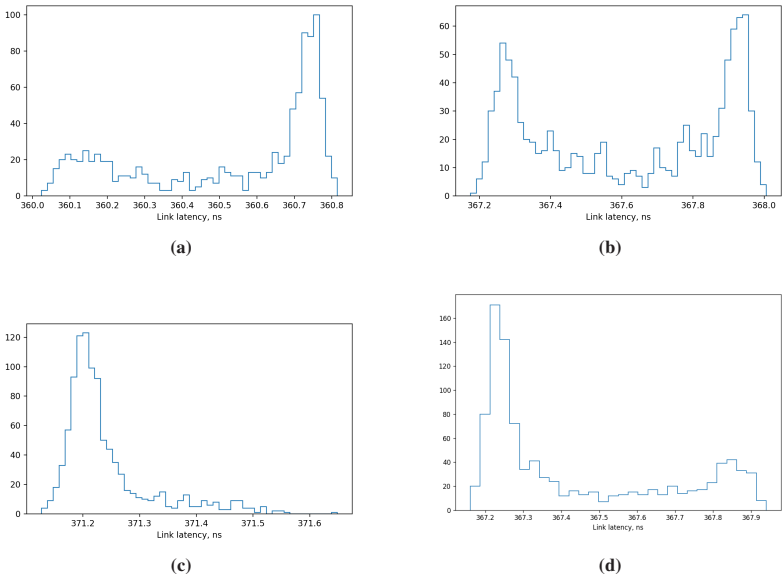


Figure 8.8: Histograms of the in-run latency variation in 4 arbitrarily selected test runs with the TFC1 design.

8.4 Baseline measurements with TFC1

To verify the characterization setup, establish a baseline performance of the TFC prototype, and quantify the improvements introduced in the optimized design described in Chapter 7, the data transport latency of the Phase-0 TFC1 implementation was evaluated. For this purpose, a point-to-point connection between two nodes was characterized in 10 runs, each consisting of 1000 latency measurements. The resulting distributions showed a spread of approximately 1.2 ns within individual runs, as illustrated by the latency histograms of four representative runs in Figure 8.8.

The histograms reveal that the data transport latency exhibits a complex distribution within each run and an inconsistent shape between different runs. In several cases, the tendency to form two distinct statistical peaks can be observed, with the

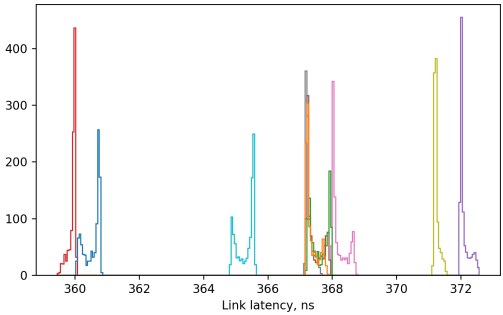


Figure 8.9: A combined histogram of the latency in all runs with the TFC1 design.

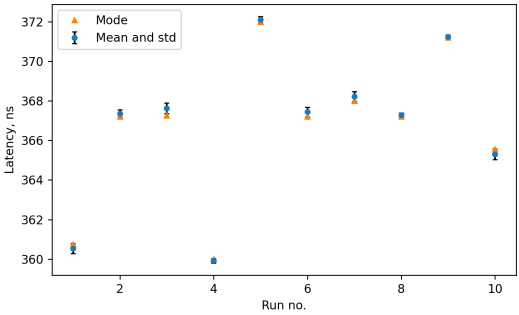


Figure 8.10: Reset-to-reset latency variance of the TFC1 design.

separation between peaks being ∼600 ps. Since this time interval does not correspond to any known time constant in the design, the observed distributions are best interpreted as complex probability density functions. A more detailed decomposition would be required to separate the components and identify the sources of in-run variation. Nevertheless, given that TFC1 was conceived only as an intermediate prototype, this behavior is considered sufficient to establish a baseline for comparison with later designs.

To place the latency variation into the broader context of reset-to-reset behavior, a combined histogram of all 10 runs is shown in Figure 8.9. The overall spread

of latency variation, including both in-run fluctuations and differences between runs, amounts to approximately 14 ns. The effect of system resets alone can be isolated by examining the mean latency values between runs, as presented in Figure 8.10. Given the asymmetric shapes of the distributions, modal values have also been added to the plot to reflect the latency skew caused by uneven statistical peaks. From this analysis, the variation introduced solely by system resets can be estimated to be $\sim$12 ns across the sample of 10 runs, significantly dominating over the in-run variance.

8.5 Characterization of the TFC2 data transport

To assess the final performance of the improved TFC2 implementation and to quantify its improvement relative to the established baseline, a more extensive test was carried out. As in the previous study, the evaluation was based on a one-hop configuration between two nodes, but in this case 50 runs were performed, each consisting of 1000 latency measurements. The results demonstrate a substantial improvement in latency consistency compared to TFC1. Representative latency histograms of four arbitrarily selected runs are shown in Figure 8.11.

Unlike the complex and inconsistent distributions observed with TFC1, the in-run latency variation in TFC2 exhibits a shape that closely resembles a normal distribution. Gaussian fits to the histograms yield a standard deviation of 22-24 ps for the in-run latency. To capture the overall behavior, including both in-run variation and reset-to-reset effects, all histograms were combined into a single distribution, as presented in Figure 8.12. The aggregated results indicate an overall standard deviation of approximately 30.6 ps and a mean latency of 260.42 ns.

At this level of determinism, it is important to recognize that a significant fraction of the observed jitter likely arises from the measurement setup itself, particularly from the use of Xilinx MMCM primitives for generating the GBT-FPGA frame clock. Consequently, the measured jitter represents an upper bound of the actual

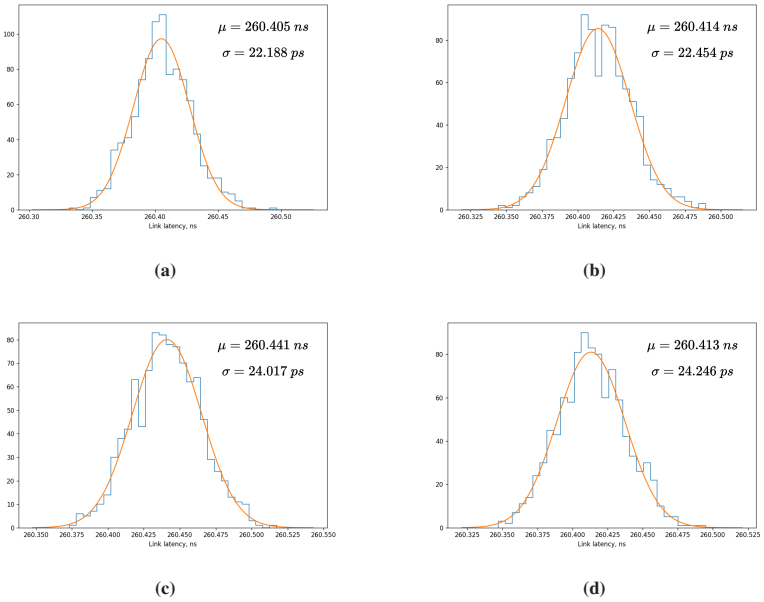


Figure 8.11: Histograms of the in-run latency variation in 4 arbitrarily selected test runs with the TFC2 design.

system performance. Nonetheless, even with these contributions included, the results already satisfy the requirements of the CBM experiment for deterministic data transport latency.

As in the baseline tests, the impact of system resets on latency variation was isolated by analyzing the mean values across measurement runs, as shown in Figure 8.13. The spread of mean latency values across 50 runs was found to be within approximately 70 ps. Remarkably, the mean values tend to cluster into two groups, one centered around 260.41 ns and the other around 260.45 ns. This suggests that the reset-to-reset behavior may follow a more complex statistical function and could be subject to further analysis to identify the underlying sources.

Since the current TFC2 implementation already demonstrates latency performance compatible with the stringent requirements of the CBM experiment, detailed

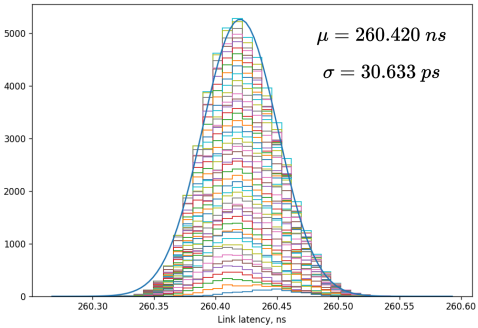


Figure 8.12: A stacked histogram of the latency in all runs with the TFC2 design.

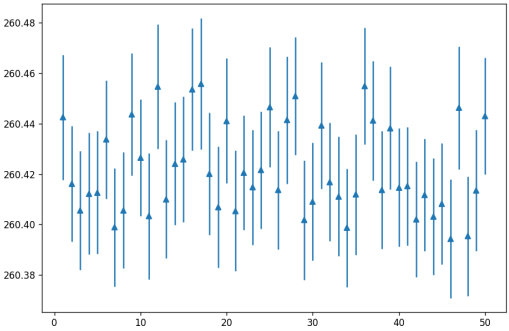


Figure 8.13: Reset-to-reset latency variance of the TFC2 design.

investigation of the remaining reset-to-reset variation has been left for potential future upgrades and is not pursued further in this work.

8.6 Summary

The high dependency of the CBM experiment on the timing synchronization of the readout components dictates the need for a thorough characterization of the TFC system. Therefore, an automated characterization method has been developed, with repeated measurements of the message transport latency and time distribution accuracy. The characterization method has been implemented in a highly configurable fashion, with the test control software being able to control the runtime configuration of the TFC setup under test, as well as the power supplies for a full reboot and reinitialization of the test setup. Therefore, unlike most similar verification procedures, the TFC characterization measures and records the runtime variation of parameters and their variation between full system restarts.

The control software has been built in a modular and extensible architecture, using standard interfaces for lab instrumentation. Thus, the characterization setup is implemented with long-term reusability in mind, to validate further upgrades and reconfigurations of the TFC system.

As a result, the TFC2 implementation has been tested and has demonstrated <25 ps RMS runtime latency variance and ∼70 ps peak-to-peak latency variance between system resets. With both effects factored in, the total variance has been measured to be ∼31 ps RMS. In the final configuration of the system with 2 network hops, this is expected to scale to $31 \times \sqrt{2} =\sim 44$ ps RMS, which fits into the requirement of 200 ps RMS with a considerable safety margin.

9 Conclusion and Future Work

9.1 Conclusion

Modern particle physics experiments are increasingly adopting triggerless data acquisition schemes to improve the precision of physics measurements. In contrast to conventional triggered experiments, where only a small subset of events is preselected for further processing, triggerless approaches enable the evaluation of complex trigger signatures on significantly larger datasets and at higher interaction rates. This, however, places additional demands on the DAQ systems, with specific requirements varying across experiments, as outlined in Chapter 3. In general, compared to triggered experiments, triggerless systems must sustain substantially higher data throughput, ensure accurate synchronization across all detector electronics, and provide centralized, time-critical control to guarantee the reliable operation of the experiment.

The CBM experiment, currently under construction as part of the FAIR facility in Darmstadt, Germany, is one of the first large-scale experiments to adopt such a triggerless scheme. This thesis has presented the development of the TFC system for CBM, which is designed to enable the experiment to cope with the extreme data throughput of 1 TB/s. The TFC system aims to fulfill two key functions: it protects the DAQ electronics from buffer overflows and the resulting congestion and corruption of measured particle interaction events, and it establishes accurate synchronization across all readout electronics.

As part of this work, the detailed requirements of the CBM experiment were analyzed in Chapter 4, resulting in a formal specification for the TFC system and

the derivation of an initial design framework. According to this specification, the TFC system must:

- synchronize 200 DAQ boards with a maximum time variance of 200 ps between any two nodes;
- provide the infrastructure for status collection and fast control distribution with a maximum round-trip latency of 6 µs and a latency jitter of the command distribution not exceeding 200 ps.

Based on a survey of the state of the art, two fundamentally different approaches to meeting these requirements were prototyped and evaluated in Chapters 5 and 6. The feasibility analysis showed that the latter approach provides excellent time-distribution performance and latency determinism while offering significantly higher resource efficiency than the alternative. It was therefore selected as the foundation of the final TFC system architecture.

By employing FIFO-free, latency-deterministic data paths together with a newly developed calibration procedure for automated and reliable clock alignment, reliable bidirectional communication between TFC nodes with low and predictable latency was achieved. This formed the basis of the final architecture described in Chapter 7, which was tailored to meet the stringent requirements of the CBM experiment.

Thus, this dissertation makes three major novel contributions to distributed real-time systems:

- **Hardware-level latency determinism**: unlike White Rabbit and TSN, this system achieves sub-nanosecond latency determinism through direct hardware control of clock domains and register-based CDCs.
- **Novel calibration algorithm**: an automated link calibration routine that scans phase shifts across all links, identifies "data integrity windows", and selects optimal system clock phase that maximizes timing margin across all transmitters and receivers simultaneously.

- **Submaster node architecture**: newly designed intermediate node combining master and endpoint functionality, enabling hierarchical scaling to over 200 endpoints with 47:1 link aggregation ratio while maintaining fixed-latency datapaths.

Given the mission-critical nature of the system, intermediate implementations of the architecture were continuously integrated into the experiment development environment. This allowed other subsystems, including detectors and readout chains, which depend on global synchronization, to progress in parallel with the development of the TFC system.

Because of the tight performance constraints, a rigorous and highly automated characterization procedure was developed and described in Chapter 8. A complete technical framework was built and deployed within the CBM development setup to ensure systematic evaluation of the TFC system. Using this framework, the final architecture was characterized and its performance compared against the baseline established with the initial prototype. The results demonstrated a one-hop data transport latency of approximately 260 ns, with an absolute latency variation of around 30.6 ps RMS. Since time synchronization is performed over the same links as fast control message transport, the synchronization accuracy is directly equivalent to the measured latency variation.

These results confirm that the TFC architecture fulfills the requirements of the CBM experiment with a tangible safety margin, as summarized in Table 9.1. The successful completion of this work thus provides a critical enabling technology for the CBM experiment and demonstrates the feasibility of deploying highly deterministic DRTS in large-scale, triggerless particle physics experiments.

Thus, as a result of this work, the architecture of a TFC system for the CBM experiment has been developed and thoroughly validated. Unlike similar systems reviewed in the state-of-the-art chapter, this system provides high-precision time synchronization and message transport latency determinism on the hardware level in a fully point-to-point system architecture. This architecture adds to the versatility of the system and opens numerous possibilities for complex system

Requirement	Specification	Achieved performance	Safety margin
Time variance between nodes	$\leq$200 ps	$\sim$44 ps	$\sim$4.5$\times$
Latency jitter	$\leq$200 ps	$\sim$44 ps	$\sim$4.5$\times$
Round-trip latency	$\leq$6 µs	$\sim$1.04 µs (2 hops)	$\sim$5.8$\times$
Endpoint count	$\geq$200	2209 (theoretical)	$\sim$11$\times$

Table 9.1: TFC requirements vs. achieved performance.

control strategies, e.g. domain-based control or advanced status data aggregation algorithms in the intermediate nodes.

This dissertation demonstrates that highly deterministic distributed real-time systems can be successfully deployed in large-scale, triggerless particle physics experiments. The TFC system represents a new architectural approach that prioritizes hardware-level determinism over protocol-layer abstraction of the latency model. The successful integration of this system into the CBM experiment validates the design decisions and provides confidence that the scientific goals of the experiment will be achievable. As triggerless data acquisition becomes increasingly common in modern experiments, the techniques and methodologies developed in this work can serve as reference implementations for future distributed timing and control systems.

9.2 Future Work

The development of the TFC system presented in this dissertation establishes a solid foundation for the Timing and Fast Control infrastructure of the CBM experiment. While the current implementation fulfills the Day-1 requirements, several directions for continued development remain open. This section outlines key areas where further research and development efforts should focus to extend the capabilities of the TFC system beyond its initial deployment.

Development of the Fast Control functionality

The hardware foundation established in this work provides a robust platform for implementing advanced fast control functionality. As noted in Chapter 4, the TFC system must allow for expanding the range of fast control commands as the experiment evolves. Future work should concentrate on building upon the existing infrastructure rather than redesigning core components.

The generic command format defined in Chapter 7 can be extended to support increasingly complex control scenarios without hindering the performance of system synchronization. The redundant bandwidth of the transport links, intentionally reserved during design, should be utilized to provide the necessary margin for future upgrades. This approach ensures backward compatibility while enabling forward-looking enhancements.

Expansion of the characterization framework

The rigorous characterization procedure established in Chapter 8 represents a significant contribution to ensuring systematic verification against stringent requirements. However, the characterization framework can be extended to support long-term operation and system evolution.

Extending the characterization methodology to understand the scaling of latency and timing error with added nodes and layers is particularly important. Since the theoretical maximum number of CRI boards that can be served is 2209 (47^2), while the Day-1 configuration requires only approximately 250 endpoints, comprehensive testing across different topology configurations will validate the hierarchical topology performance under various load conditions. This scalability analysis will guarantee informed decisions about network expansion as additional detector subsystems are integrated.

Development of new tests in the characterization software

The modular architecture of the test control software provides a flexible basis for developing additional characterization tests that address long-term system stability. While the current implementation demonstrates excellent short-term performance, understanding how the system behaves over extended periods under varying environmental conditions represents an important direction for future development.

Long-term latency stability testing should extend the existing characterization framework by modifying Section 8.2 sequences to run continuously over hours or days, revealing subtle drift mechanisms invisible during short-duration tests. Besides, different aspects and sources of latency variations require systematic investigation beyond Section 7.1.1. Further studies would include controlled changes of data transport parameters and identification of their contribution to the latency variation. These developments extend the scope from initial validation to fine parameter tuning and sustained operational monitoring.

Performance enhancement for data transport determinism

While the current measurements demonstrate competitive performance with one-hop data transport latency of approximately 260 ns and absolute latency variation of approximately 30.6 ps RMS, opportunities for further optimization remain.

Further datapath optimization should address the sources of subtle latency variation, such as the clock recovery circuitry and the internal datapath of the Gigabit Transceivers. Sufficient understanding of the influence of the internal transceiver parameters and circuits on the resulting latency stability helps to develop techniques to further reduce the latency variations and thus improve the performance of the time distribution and the global data flow control in the CBM experiment and other applications that could benefit from an excellent latency stability.

List of Figures

1.1 A visualization of a simulated central Au+Au collision with its products. (Credit: CBM collaboration) 3
1.2 Beam intensity fluctuations with slow extraction at SIS18 [34]. . . . 5
2.1 Demonstration of the imperfections of the local system clock: time offset (left) and frequency offset and drift (right). 19
2.2 Typical aging behaviors. 22
2.3 Wander and jitter effects on the clock stability. 24
2.4 Synchronization condition in a plesiochronous network. 25
2.5 Classification of time and frequency distribution networks. 27
2.6 A hierarchical master-slave network. 29
2.7 Time propagation in a master-slave synchronous network. 29
2.8 A fully-connected mutually synchronized network. 30
2.9 A malicious node preventing a consensus in a mutually synchronized network (Byzantine error). 32
2.10 An example of a multi-layer trigger system in a particle physics experiment. 34
2.11 A parallelized trigger system in the CMS experiment (image used under CC BY-NC-SA 1.0) [53]. 35
2.12 A comparison of a triggered (top) and a streaming (bottom) data readout concepts. 36
2.13 Construction of the FAIR facility as of October 2025. (Credit: D. Fehrenz, GSI/FAIR) 37
2.14 FAIR accelerator complex with the location of the CBM experiment. (Credit: CBM collaboration) 37
2.15 Digital visualization of the CBM cave. (Credit: GSI/FAIR, Zeitrausch) 38
2.16 QCD phase diagram. The area being explored by the FAIR experiments is highlighted. (Credit: CBM collaboration) 38

2.17 Comparison of interaction rates of various experiments, including CBM [59]. 39
2.18 The architecture of the CBM DAQ system with the central control systems included. 42
3.1 Topology of a White Rabbit network. (Credit: Tomasz Włostowski) [68]. 46
4.1 Typical topologies of real-time systems: a - bus, b - ring, c - star, d - tree, e - mesh. One controller and one endpoint are placed at the worst-case distances with a route between them highlighted in red. 65
4.2 Commonly used transmission media classes. 68
4.3 Classification of wireless signals by frequency ranges. 69
4.4 General architecture of the TFC system. 71
4.5 Alternative configurations of the TFC architecture. 73
4.6 Synchronization of local node clocks in the TFC system. 74
4.7 Demonstration of a fast control cycle using an example of clearing FIFOs in case of local overflow. 75
5.1 Link synchronization between master and slave nodes. 79
5.2 Clock relation between Master and Slave clocks. 80
5.3 The synchronization procedure with the enhanced timestamping. . . 81
5.4 Schematic diagram of a DDMTD. 83
5.5 Clock and phase stretching in a DDMTD for $N = 8$. 83
5.6 Hardware- (a) and software- (b) PLL architectures 86
5.7 Two-phase PI phase and frequency control in White Rabbit. 86
5.8 Concurrent PI phase and frequency control for potential use in the TFC system. 87
5.9 Sample simulation of the SPLL parameters. 88
5.10 The AFCK board used as the TFC prototyping platform with two FM-S14 connectivity boards installed. 89
5.11 Gateware architecture of the prototype. 90
5.12 Test setup for the prototype of the time self-adjustment approach. . . 94
5.13 Measurement of the real resolution of the DDMTD module. 95
6.1 Time distribution with a constant transmission delay. 100
6.2 General architecture of an FPGA implementation of an MGT IP core. Only basic functions are included, vendor-specific implementations may include additional components. 101
6.3 Clock distribution via self-synchronous optical links. 104

6.4 Link initialization procedure using direct communication between a master and an endpoint nodes. 106
6.5 State diagrams of the finite state machines used on the upstream (left) and the downstream (right) ends on the link. 106
6.6 A functional diagram of the link prioritization scheme used for shared link access. 107
6.7 An example of prioritized link access with two senders concurrently requesting message transmission. 108
6.8 An example of registering the phase position of the received 40 MHz clock by an endpoint using a subcycle counter. 109
6.9 Multiple BNL-712 boards mounted in a server node. 111
6.10 Control paths of a TFC node. 112
6.11 Gateware architectures of a master and an endpoint nodes. 113
6.12 Architecture of the link access unit. 114
6.13 Test setup for evaluation of the prototype design. 114
6.14 Test results of the TFC prototype. 116
6.15 The Phase-0 CBM experimental setup. Credit: CBM collaboration. . 117
6.16 Time alignment between various subsystems in the Phase-0 CBM setup. Credit: CBM collaboration [30]. 118
7.1 Potential data ambiguity after deserialization depending on the parallel clock phase. Numbers in the data represent bit positions in frames rather than bit values. 122
7.2 Register-based CDC between the frame clock domain and the word clock domains in the transmitter datapath. 126
7.3 Clock generation and serialization/deserialization stages in data transmission between GBT-FPGA cores. 126
7.4 Register-based CDC in the receiver datapath and the header flag generation logic. 127
7.5 Format of an upstream TFC frame. 129
7.6 Format of a downstream TFC frame. 131
7.7 Gateware architecture of the TFC2 nodes. 132
7.8 State diagram of the timing FSM module. 133
7.9 Optimal alignment of the parallel clock relative to the serial clock. . . 135
7.10 Results of the phase scan in an uncalibrated master node. 136
7.11 Results of the phase scan in a calibrated master node. 137

7.12 Three different possibilities of Rx alignment in the submaster node. . 137
7.13 Results of the phase scan in a calibrated submaster node. 138
8.1 The concept of measuring data transport latency through a time distance between electric pulses. 143
8.2 Algorithm of the automated characterization of the TFC system. . . . 144
8.3 Architecture of the characterization setup of the TFC data transport. . 145
8.4 An example of a measurement pulse using the Tektronix TDS6154C oscilloscope. 147
8.5 Modular architecture of the test control software. 148
8.6 Remote connection between the test control software and server nodes using the DCA infrastructure. 149
8.7 Initialization sequence of the tests. 152
8.8 Histograms of the in-run latency variation in 4 arbitrarily selected test runs with the TFC1 design. 153
8.9 A combined histogram of the latency in all runs with the TFC1 design. 154
8.10 Reset-to-reset latency variance of the TFC1 design. 154
8.11 Histograms of the in-run latency variation in 4 arbitrarily selected test runs with the TFC2 design. 156
8.12 A stacked histogram of the latency in all runs with the TFC2 design. 157
8.13 Reset-to-reset latency variance of the TFC2 design. 157

List of Tables

1.1 Number of FPGA cards in DAQ systems of different experiments. . . . 4
7.1 Estimated wrap period of time counters with various widths at the 40 MHz rate. 130
9.1 TFC requirements vs. achieved performance. 162

List of Publications

[1] CBM Collaboration, I. Kisel, S. Bähr, J. Becker, V. Sidorenko *et al.*, "Event topology reconstruction in the cbm experiment," *Journal of Physics: Conference Series*, vol. 1070, p. 012015, 2018.

[2] CBM Collaboration, P. Senger, S. Bähr, J. Becker, V. Sidorenko *et al.*, "Exploring cosmic matter in the laboratory—the compressed baryonic matter experiment at fair," *Particles*, vol. 2, no. 4, pp. 499–510, 2019, 54.02.03; LK 01.

[3] CBM Collaboration, N. Karpushkin, D. Finogeev, J. Becker, V. Sidorenko *et al.*, "The projectile spectator detector for measuring the geometry of heavy ion collisions at the cbm experiment on fair," *Nuclear instruments & methods in physics research / A*, vol. 936, pp. 156–157, 2019, 54.02.03; LK 01.

[4] CBM Collaboration, V. Mikhaylov, A. Kugler, J. Becker, V. Sidorenko *et al.*, "The very forward hadron calorimeter psd for the future cbm@fair experiment," *The European physical journal / Web of Conferences*, vol. 204, pp. Art.–Nr.: 11 004, 2019, 54.02.03; LK 01.

[5] CBM Collaboration, O. Golosov, V. Klochkov, J. Becker, V. Sidorenko *et al.*, "Performance for proton anisotropic flow measurement of the cbm experiment at fair," *Journal of Physics: Conference Series*, vol. 1690, pp. Art.–Nr. 012 104, 2020.

[6] CBM Collaboration, I. Segal, O. Lubynets, J. Becker, V. Sidorenko *et al.*, "Using multiplicity of produced particles for centrality determination in heavy-ion collisions with the cbm experiment," *Journal of Physics: Conference Series*, vol. 1690, pp. Art.–Nr.: 012 107, 2020.

[7] CBM Collaboration, O. Golosov, V. Klochkov, J. Becker, V. Sidorenko *et al.*, "Physics performance studies for anisotropic flow measurements with the cbm experiment at fair," *Physics of particles and nuclei*, vol. 51, no. 3, p. 297–300, 2020.

[8] CBM Collaboration, P. Senger, S. Bähr, J. Becker, V. Sidorenko *et al.*, "Probing dense qcd matter in the laboratory—the cbm experiment at fair," *Physica scripta*, vol. 95, no. 7, pp. Art.–Nr.: 074 003, 2020.

[9] ——, "Status of the compressed baryonic matter experiment at fair," *International journal of modern physics / E*, vol. 29, no. 02, pp. 2 030 001–1–2 030 001–26, 2020, 54.02.03; LK 01.

[10] BM@N Collaboration, CBM Collaboration, NA61/SHINE Collaboration, J. Becker, V. Sidorenko *et al.*, "Transverse and longitudinal segmented forward hadron calorimeters with sipms light readout for future fixed target heavy ion experiments," *Nuclear instruments & methods in physics research / A*, vol. 958, pp. Art.–Nr. 162 728, 2020, 54.02.03; LK 01.

[11] CBM Collaboration, S. Bähr, J. Becker, V. Sidorenko, K. L. Unger *et al.*, "Cbm collaboration," *Nuclear physics <Amsterdam> / A*, vol. 1005, p. Article no: 122089, 2021.

[12] CBM Collaboration, S. Samanta, S. Bähr, J. Becker, V. Sidorenko *et al.*, "Feasibility studies of conserved charge fluctuations in au-au collisions with cbm," *Nuclear physics <Amsterdam> / A*, vol. 1005, pp. Art.–Nr.: 121 896, 2021, 54.12.03; LK 01.

[13] F. Kreß, J. Hoefer, T. Hotfilter, V. Sidorenko, J. Becker *et al.*, "Hardware-aware partitioning of convolutional neural network inference for embedded ai applications," in *18th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEEXplore, 2022, pp. 133–140.

[14] V. Sidorenko, I. Fröhlich, W. F. J. Müller, S. Bähr, J. Becker *et al.*, "Prototype design of a timing and fast control system in the cbm experiment," *Journal of Instrumentation*, vol. 17, pp. Art.–Nr.: C05 008, 2022, 54.12.02; LK 01.

[15] V. Sidorenko, W. F. J. Müller, W. Zabolotny, I. Fröhlich, J. Becker *et al.*, "Evaluation of gbt-fpga for timing and fast control in cbm experiment," *Journal of Instrumentation*, vol. 18, no. 2, pp. Art.–Nr.: C02 052, 2023, 54.12.02; LK 01.

[16] F. Kreß, V. Sidorenko, P. Schmidt, T. Hotfilter, J. Becker *et al.*, "Cnnparted: An open source framework for efficient convolutional neural network inference partitioning in embedded systems," *Computer Networks*, vol. 229, p. Article no: 109759, 2023.

[17] M. Stammler, V. Sidorenko, F. Kreß, P. Schmidt, and J. Becker, "Context-aware layer scheduling for seamless neural network inference in cloud-edge systems," in *2023 IEEE 16th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC), Singapur, 18th-21st December 2023*. Institute of Electrical and Electronics Engineers (IEEE), 2024, pp. 97–104.

[18] F. Kreß, V. Sidorenko, I. Topko, K. Unger, J. Becker *et al.*, "Vhdl crash course: A multimedia-based teaching approach," in *2024 IEEE 3rd German Education Conference (GECon), Munich, Germany, 05-07 August 2024*. Institute of Electrical and Electronics Engineers (IEEE), 2024.

[19] CBM Collaboration, V. Sidorenko, W. F. J. Müller, D. Emschermann, W. Zabolotny, J. Becker *et al.*, "Time error accumulation in a hierarchical time and clock distribution network with deterministic optical links," *Journal of Instrumentation*, vol. 19, no. 03, pp. Art.–Nr.: C03 014, 2024.

[20] O. Manzhura, M. Caselle, V. Sidorenko, K. L. Unger, J. Becker *et al.*, "The data acquisition system for the panda micro-vertex detector," *Journal of Instrumentation*, vol. 19, no. 03, pp. Art.–Nr.: C03 036, 2024, 54.12.03; LK 01.

[21] F. Lenta, D. Calvo, F. Cossio, V. Sidorenko, J. Becker *et al.*, "Characterization of the radiation tolerant toast asic for the readout of the panda mvd strip detector," *Journal of Instrumentation*, vol. 19, no. 04, pp. Art.–Nr.: C04 047, 2024, 54.12.03; LK 01.

[22] PANDA collaboration, M. Caselle, O. Manzhura, V. Sidorenko, J. Becker *et al.*, “Performance of the daq system of the panda micro-vertex detector,” *Journal of Instrumentation*, vol. 19, no. 12, p. C12015, 2024, 54.12.03; LK 01.

[23] G. Mazza, D. Calvo, F. Cossio, V. Sidorenko, J. Becker *et al.*, “Toast: A 64-channel asic for the readout of the silicon strip detectors of the panda experiment,” *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 1071, p. 170069, 2025, 54.12.03; LK 01.

[24] A. Agarwal, CBM Collaboration, J. Becker, V. Sidorenko, K. L. Unger *et al.*, “Performance of the prototype silicon tracking system of the cbm experiment tested with heavy-ion beams at sis18,” *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 1082, no. 2, p. 171059, 2026, 54.12.03; LK 01.

Bibliography

[1] M. P. Brundage, Q. Chang, Y. Li, J. Arinez, and G. Xiao, "Implementing a real-time, energy-efficient control methodology to maximize manufacturing profits," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2016.

[2] Y.-Y. Chen, Y.-H. Lin, Y.-C. Hu, C.-H. Hsia *et al.*, "Distributed real-time object detection based on edge-cloud collaboration for smart video surveillance applications," *IEEE Access*, 2022.

[3] B. V. Philip, T. Alpcan, J. Jin, and M. Palaniswami, "Distributed real-time iot for autonomous vehicles," *IEEE Transactions on Industrial Informatics*, 2019.

[4] L. Yu, T. Jiang, and Y. Zou, "Distributed real-time energy management in data center microgrids," *arXiv: Distributed, Parallel, and Cluster Computing*, 2016.

[5] J. M. Mendes Filho, É. Lucet, and D. Filliat, "Real-time distributed receding horizon motion planning and control for mobile multi-robot dynamic systems," *IEEE International Conference on Robotics and Automation*, 2017.

[6] M. Abdelkader, Y. Lu, H. Jaleel, and J. S. Shamma, "Distributed real time control of multiple uavs in adversarial environment: Algorithm and flight testing results," *IEEE International Conference on Robotics and Automation*, 2018.

[7] M. Hu, J.-W. Xiao, S. Cui, and Y.-W. Wang, "Distributed real-time demand response for energy management scheduling in smart grid," *International Journal of Electrical Power & Energy Systems*, 2018.

[8] N. Finn and P. Thubert, “Deterministic networking problem statement,” *Request for Comments*, 2019.

[9] E. Grossman, “Deterministic networking use cases,” *Request for Comments*, 2019.

[10] H. Austad and G. Mathisen, “Bounding the end-to-end execution time in distributed real-time systems: Arguing the case for deterministic networks in lingua franca,” *CPS-IoT Week Workshops*, 2023.

[11] N. Finn, B. Varga, P. Thubert, and J. Farkas, “Deterministic networking architecture,” *Request for Comments*, 2019.

[12] Y. Hu, B. Guo, C. Yang, and Z. Han, “Time-deterministic networking for satellite-based internet-of-things services: Architecture, key technologies, and future directions,” *IEEE Network*, 2024.

[13] J.-L. Ferrant, M. Gilson, S. Jobert, M. Mayer, and L. Montini, “Synchronous ethernet and ieee 1588 in telecoms,” 2013.

[14] A. Arestova, K.-S. Hielscher, and R. German, “Simulative evaluation of the tsn mechanisms time-aware shaper and frame preemption and their suitability for industrial use cases,” *2021 IFIP Networking Conference (IFIP Networking)*, 2021.

[15] X. Xu, Z. Xiong, J. Wu, and X. Zhu, “High-precision time synchronization in real-time ethernet-based cnc systems,” *The International Journal of Advanced Manufacturing Technology*, 2013.

[16] J. Ruh, W. Steiner, and G. Fohler, “Clock synchronization in virtualized distributed real-time systems using ieee 802.1as and acrn,” *IEEE Access*, 2021.

[17] J. Schacht, H. P. Laqua, and H. Niedermeyer, “Synchronization of processes in a distributed real time system exemplified by the control system of the fusion experiment wendelstein 7-x,” *IEEE Transactions on Nuclear Science*, 2005.

[18] R. Kumar and N. Monga, "Design and implementation issues of a synchronization in distributed system," *International Journal of Engineering Applied Sciences and Technology*, 2025.

[19] J. Lai, H. Zhou, W. Hu, X. Lu, and L. Zhong, "Synchronization of hybrid microgrids with communication latency," *Mathematical Problems in Engineering*, 2015.

[20] M. Rizzi, M. Lipinski, P. Ferrari, S. Rinaldi, and A. Flammini, "White rabbit clock synchronization: Ultimate limits on close-in phase noise and short-term stability due to fpga implementation," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 65, no. 9, pp. 1726–1737, 2018.

[21] M. S. Livingston, "Early history of particle accelerators," p. 1–88, 1980. [Online]. Available: http://dx.doi.org/10.1016/s0065-2539(08)61061-6

[22] L. Evans and P. Bryant, "Lhc machine," *Journal of Instrumentation*, vol. 3, no. 08, p. S08001–S08001, Aug. 2008. [Online]. Available: http://dx.doi.org/10.1088/1748-0221/3/08/S08001

[23] A. P. Chernyaev and S. M. Varzar, "Particle accelerators in modern world," *Physics of Atomic Nuclei*, vol. 77, no. 10, p. 1203–1215, Oct. 2014. [Online]. Available: http://dx.doi.org/10.1134/s1063778814100032

[24] C. Zampolli, "Alice data processing for run 3 and run 4 at the lhc," p. 929, Feb. 2021. [Online]. Available: http://dx.doi.org/10.22323/1.390.0929

[25] V. V. Gligorov and V. Reković, "Review of real-time data processing for collider experiments," *The European Physical Journal Plus*, vol. 138, no. 11, Nov. 2023. [Online]. Available: http://dx.doi.org/10.1140/epjp/s13360-023-04599-6

[26] S. Ahmad Khan, J. Mitra, and T. K. Nayak, "Fpga based high speed daq systems for hep experiments: Potential challenges," in *Springer Proceedings in Physics*. Springer International Publishing, 2023, pp. 83–89. [Online]. Available: http://dx.doi.org/10.1007/978-3-031-19268-5_11

[27] A. Perro, "Emerging technologies and development methodologies for high-throughput data acquisition systems," Ph.D. dissertation, IN2P3 (Centre de Recherche en Physique des Particules), France, 2025, doctoral thesis; supervisors: Renaud Le Gac, Olivier Leroy, Paolo Durante. [Online]. Available: https://indico.in2p3.fr/event/36521/attachments/93783/143733/PhD_aperro_25-final.pdf

[28] G. Badaro, U. Behrens, A. Bocci, J. Branson, and P. Brummer, "The phase-2 upgrade of the cms data acquisition," *EPJ Web of Conferences*, vol. 251, p. 04023, 2021. [Online]. Available: http://dx.doi.org/10.1051/epjconf/202125104023

[29] V. Barroso, K. Alexopoulos, M. Boulais, S. Chapeland, and F. Costa, "The new alice data acquisition system (o2/flp) for lhc run 3," *EPJ Web of Conferences*, vol. 295, p. 02029, 2024. [Online]. Available: http://dx.doi.org/10.1051/epjconf/202429502029

[30] CBM Collaboration, *Technical Design Report for the CBM Online Systems - Part I, DAQ and FLES Entry Stage*. GSI Helmholtzzentrum fuer Schwerionenforschung, GSI, Darmstadt, 2023. [Online]. Available: https://repository.gsi.de/record/340597

[31] L. Lyons, "Open statistical issues in particle physics," *The Annals of Applied Statistics*, vol. 2, no. 3, pp. 887–915, 2008. [Online]. Available: https://doi.org/10.1214/08-AOAS163

[32] H. H. Gutbrod, I. Augustin, H. Eickhoff, K.-D. Groß, W. F. Henning, D. Krämer, and G. Walter, "Fair baseline technical report (volume 2): Accelerator and scientific infrastructure," GSI Helmholtz Centre for Heavy Ion Research, Tech. Rep., Mar. 2006, volume 2 of the six-volume FAIR Baseline Technical Report (BTR); includes detailed description of accelerator lattice layout, ring topology, and infrastructure for the Facility for Antiproton and Ion Research (FAIR). [Online]. Available: http://www.gsi.de/fair/reports/btr.html

[33] R. Singh, P. Forck, P. Boutachkov, S. Sorge, and H. Welker, "Slow extraction spill characterization from micro to milli-second scale," *Journal of Physics: Conference Series*, vol. 1067, no. 7, p. 072002, sep 2018. [Online]. Available: https://dx.doi.org/10.1088/1742-6596/1067/7/072002

[34] X. Gao, D. Emschermann, J. Lehnert, and W. F. Müller, "Throttling strategies and optimization of the trigger-less streaming daq system in the cbm experiment," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 978, p. 164442, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0168900220308391

[35] V. Akishina, I. Kisel, I. Vassiliev, and M. Zyzak, "Time-based reconstruction of free-streaming data in cbm," *EPJ Web of Conferences*, vol. 173, p. 04002, 2018. [Online]. Available: http://dx.doi.org/10.1051/epjconf/201817304002

[36] J. Anderson, A. Borga, H. Boterenbrood, H. Chen, K. Chen *et al.*, "Felix: a high-throughput network approach for interfacing to front end electronics for atlas upgrades," *Journal of Physics: Conference Series*, vol. 664, no. 8, p. 082050, Dec. 2015. [Online]. Available: http://dx.doi.org/10.1088/1742-6596/664/8/082050

[37] R. Grabit, J. K. Troska, K. Gill, F. Vasey, and G. Dewhirst, "80mbit/s digital optical links for control, timing and trigger of the cms tracker," 2002. [Online]. Available: http://cds.cern.ch/record/592727

[38] H. Kopetz and W. Steiner, *Real-Time Communication*. Cham: Springer International Publishing, 2022, pp. 177–200. [Online]. Available: https://doi.org/10.1007/978-3-031-11992-7_7

[39] A. Ichimura, T. Yokoyama, and M. Yoo, "A time-triggered distributed computing environment for cyber-physical systems based on physical time and logical time," in *TENCON 2018 - 2018 IEEE Region 10 Conference*, 2018, pp. 1510–1515.

[40] L. Lamport, “Time, clocks, and the ordering of events in a distributed system,” *Commun. ACM*, vol. 21, no. 7, p. 558–565, jul 1978. [Online]. Available: https://doi.org/10.1145/359545.359563

[41] H. Kopetz and W. Steiner, *Global Time*. Cham: Springer International Publishing, 2022, pp. 57–85. [Online]. Available: https://doi.org/10.1007/978-3-031-11992-7_3

[42] Z. Chen, D. Li, Y. Huang, and C. Tang, “Event-triggered communication for time synchronization in wsns,” *Neurocomputing*, vol. 177, pp. 416–426, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231215017804

[43] R. Obermaisser, *Properties of time-triggered communication systems*, 01 2011, pp. 25–51.

[44] International Telecommunication Union, “Definitions and terminology for synchronization networks,” ITU-T, Tech. Rep. G.810, August 1996. [Online]. Available: https://handle.itu.int/11.1002/1000/3713

[45] J. Vig and T. Meeker, “The aging of bulk acoustic wave resonators, filters and oscillators,” in *Proceedings of the 45th Annual Symposium on Frequency Control 1991*, 1991, pp. 77–101.

[46] S.-y. Wang, B. Neubig, K. Sato, T. Hosoda, and E. Seydel, “Aging models and parameters of quartz crystal resonators and oscillators,” in *2015 Symposium on Piezoelectricity, Acoustic Waves, and Device Applications (SPAWDA)*, 2015, pp. 382–385.

[47] K.-S. Park, N. Pal, Y. Li, R. Xia, and T. Wang, “A temperature- and aging-compensated rc oscillator with ±1030-ppm inaccuracy from 40 °c to 85 °c after accelerated aging for 500 h at 125 °c,” *IEEE Journal of Solid-State Circuits*, vol. 58, no. 12, pp. 3459–3469, 2023.

[48] F. Tirado-Andrés and A. Araujo, “Performance of clock sources and their influence on time synchronization in wireless sensor networks,” *International Journal of Distributed Sensor Networks*, vol. 15, no. 9, 2019.

[49] W. Lindsey, F. Ghazvinian, W. Hagmann, and K. Dessouky, "Network synchronization," *Proceedings of the IEEE*, vol. 73, no. 10, pp. 1445–1467, 1985.

[50] J. Savory, S. Romisch, L. Hernandez, and K. Maurer, "Local distribution and calibration of timing signals at nist," in *Proceedings of the 48th Annual Precise Time and Time Interval Systems and Applications Meeting*. Institute of Navigation, mar 2017.

[51] J. Toledo, F. J. Mora, and H. Müller, "Past, present and future of data acquisition systems in high energy physics experiments," *Microprocessors and Microsystems*, vol. 27, no. 8, pp. 353–358, Sep. 2003. [Online]. Available: http://dx.doi.org/10.1016/s0141-9331(03)00065-6

[52] W. H. Smith, "Triggering and high-level data selection," pp. 533–554, 2020. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-35318-6_12

[53] J. Varela, "The cms trigger system," p. 392, Apr. 2007. [Online]. Available: http://dx.doi.org/10.22323/1.021.0392

[54] M. Durante, P. Indelicato, B. Jonson, V. Koch, and K. Langanke, "All the fun of the fair: fundamental physics at the facility for antiproton and ion research," *Physica Scripta*, vol. 94, no. 3, p. 033001, Jan. 2019. [Online]. Available: http://dx.doi.org/10.1088/1402-4896/aaf93f

[55] H. H. Gutbrod, I. Augustin, H. Eickhoff, K.-D. Groß, W. F. Henning, D. Krämer, and G. Walter, "Fair baseline technical report (volume 1): Executive summary," GSI Helmholtz Centre for Heavy Ion Research, Tech. Rep., Sep. 2006, volume 1 of the six-volume FAIR Baseline Technical Report (BTR); contains the Executive Summary ... [Online]. Available: http://www.gsi.de/fair/reports/btr.html

[56] O. Kester, P. Spiller, and H. Stoecker, "Fair project at gsi," in *Challenges and Goals for Accelerators in the XXI Century*. World Scientific, Feb. 2016, pp. 611–622. [Online]. Available: http://dx.doi.org/10.1142/9789814436403_0032

[57] P. Senger, "Probing dense qcd matter in the laboratory—the cbm experiment at fair," *Physica Scripta*, vol. 95, no. 7, p. 074003, May 2020. [Online]. Available: http://dx.doi.org/10.1088/1402-4896/ab8c14

[58] H. H. Gutbrod, "The path to heavy ions at lhc and beyond," in *Melting Hadrons, Boiling Quarks - From Hagedorn Temperature to Ultra-Relativistic Heavy-Ion Collisions at CERN*. Springer International Publishing, 2016, pp. 97–106. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-17545-4_13

[59] T. Ablyazimov, A. Abuhoza, R. P. Adak *et al.*, "Challenges in qcd matter physics—the scientific programme of the compressed baryonic matter experiment at fair," *The European Physical Journal A*, vol. 53, no. 3, p. 60, Mar. 2017. [Online]. Available: https://doi.org/10.1140/epja/i2017-12248-y

[60] T. Galatyuk, "Future facilities for high μ physics," *Nuclear Physics A*, vol. 982, pp. 163–169, Feb. 2019. [Online]. Available: http://dx.doi.org/10.1016/j.nuclphysa.2018.11.025

[61] W. F. J. f. t. C. C. Müller, "The cbm experiment @ fair–new challenges for front-end electronics, data acquisition and trigger systems," *Journal of Physics: Conference Series*, vol. 50, pp. 371–376, Nov. 2006. [Online]. Available: http://dx.doi.org/10.1088/1742-6596/50/1/050

[62] J. Lehnert, A. P. Byszuk, D. Emschermann, K. Kasinski, and W. F. J. Müller, "Gbt based readout in the cbm experiment," *Journal of Instrumentation*, vol. 12, no. 02, pp. C02 061–C02 061, Feb. 2017. [Online]. Available: http://dx.doi.org/10.1088/1748-0221/12/02/C02061

[63] J. de Cuveland and V. Lindenstruth, "A first-level event selector for the cbm experiment at fair," *Journal of Physics: Conference Series*, vol. 331, no. 2, p. 022006, Dec. 2011. [Online]. Available: http://dx.doi.org/10.1088/1742-6596/331/2/022006

[64] J. Serrano, "The white rabbit project," in *Proceedings of the 3rd International Beam Instrumentation Conference (IBIC 2013)*, 2013, presentation paper;

provides technical introduction and system overview. [Online]. Available: https://accelconf.web.cern.ch/ibic2013/papers/thbl2.pdf

[65] P. Moreira, J. Serrano, T. Włostowski, P. Loschmidt, and G. Gaderer, "White rabbit: Sub-nanosecond timing distribution over ethernet," in *2009 International Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS)*. IEEE, Oct. 2009. [Online]. Available: http://dx.doi.org/10.1109/ISPCS.2009.5340196

[66] "ITU-T Recommendation G.8261/Y.1361 (2019): Timing and synchronization aspects in packet networks," International Telecommunication Union (ITU), Tech. Rep. G.8261/Y.1361 (08/2019), Aug. 2019, in force; defines frequency synchronization, jitter and wander limits in packet networks. [Online]. Available: https://handle.itu.int/11.1002/1000/14010

[67] "IEEE Standard 1588-2019: Precision Clock Synchronization Protocol for Networked Measurement and Control Systems," IEEE, Tech. Rep. IEEE Std 1588-2019, 2020, precision Time Protocol (PTP) standard, defining sub-microsecond synchronization for control and measurement systems. [Online]. Available: http://dx.doi.org/10.1109/IEEESTD.2020.9120376

[68] T. Włostowski, "Precise time and frequency transfer in a white rabbit network," Master's thesis, Warsaw University of Technology, Warsaw, Poland, May 2011, master's thesis.

[69] M. Rizzi, M. Lipiński, T. Włostowski, J. Serrano, G. Daniluk, P. Ferrari, and S. Rinaldi, "White rabbit clock characteristics," in *2016 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*. IEEE, Sep. 2016. [Online]. Available: http://dx.doi.org/10.1109/ISPCS.2016.7579514

[70] M. Lipinski, E. van der Bij, J. Serrano, T. Wlostowski, G. Daniluk, A. Wujek, and M. Rizzi, "White rabbit applications and enhancements," in *2018 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*. IEEE, Sep. 2018. [Online]. Available: http://dx.doi.org/10.1109/ISPCS.2018.8543072

[71] G. Daniluk, "White Rabbit Calibration Procedure (Version 1.1)," CERN, BE-CO-HT, Tech. Rep., nov 9 2015, technical note describing fiber latency, device delay, and asymmetry measurement for White Rabbit synchronization. [Online]. Available: https://white-rabbit.web.cern.ch/documents/WR_Calibration-v1.1-20151109.pdf

[72] H. Peek and P. Jansweijer, "White rabbit absolute calibration," in *2018 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*. IEEE, Sep. 2018, pp. 1–5. [Online]. Available: http://dx.doi.org/10.1109/ISPCS.2018.8543067

[73] J. Vojtech, O. Havlis, M. Slapak, V. Smotlacha, T. Horvath *et al.*, "Long haul single fiber reamplified-reshaped white rabbit transmission," in *2021 Joint Conference of the European Frequency and Time Forum and IEEE International Frequency Control Symposium (EFTF/IFCS)*. IEEE, Jul. 2021, pp. 1–3. [Online]. Available: http://dx.doi.org/10.1109/EFTF/IFCS52194.2021.9604245

[74] E. F. Dierikx, A. E. Wallin, T. Fordell, J. Myyry, and P. Koponen, "White rabbit precision time protocol on long-distance fiber links," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 63, no. 7, pp. 945–952, Jul. 2016. [Online]. Available: http://dx.doi.org/10.1109/TUFFC.2016.2518122

[75] N. Kaur, F. Frank, J. Pinto, P. Tuckey, and P.-E. Pottie, "A 500-km cascaded white rabbit link for high-performance frequency dissemination," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 69, no. 2, pp. 892–901, Feb. 2022. [Online]. Available: http://dx.doi.org/10.1109/TUFFC.2021.3134163

[76] L. Śliwczyński, P. Krehlik, A. Czubla, L. Buczek, and M. Lipiński, "Dissemination of time and rf frequency via a stabilized fibre optic link over a distance of 420 km," *Metrologia*, vol. 50, no. 2, pp. 133–145, Feb. 2013. [Online]. Available: http://dx.doi.org/10.1088/0026-1394/50/2/133

[77] M. Jimenez-Lopez, F. Torres-Gonzalez, J. L. Gutierrez-Rivas, M. Rodriguez-Alvarez, and J. Diaz, "A fully programmable white-rabbit node for the ska telescope pps distribution system," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 2, pp. 632–641, Feb. 2019. [Online]. Available: http://dx.doi.org/10.1109/TIM.2018.2851658

[78] J. Sanchez-Garrido, A. Jurado, M. Jimenez-Lopez, A. Balzer, and H. Prokoph, "A white rabbit-synchronized accurate time-stamping solution for the small-sized cameras of the cherenkov telescope array," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–14, 2021. [Online]. Available: http://dx.doi.org/10.1109/TIM.2020.3013343

[79] T. Gerrits, I. A. Burenkov, Y. S. Li-Baboud, A. Rahmouni, D. M. Anand *et al.*, "White rabbit-assisted quantum network node synchronization with quantum channel coexistence," p. FM1C.2, 2022. [Online]. Available: http://dx.doi.org/10.1364/cleo_qels.2022.fm1c.2

[80] M. Alshowkan, P. G. Evans, B. P. Williams, N. S. V. Rao, C. E. Marvinney *et al.*, "Synchronizing a quantum local area network with white rabbit," p. FM1C.4, 2022. [Online]. Available: http://dx.doi.org/10.1364/cleo_qels.2022.fm1c.4

[81] F. Girela-Lopez, E. Ros, and J. Diaz, "Precise network time monitoring: Picosecond-level packet timestamping for fintech networks," *IEEE Access*, vol. 9, pp. 40 274–40 285, 2021. [Online]. Available: http://dx.doi.org/10.1109/ACCESS.2021.3064987

[82] J. Lopez-Jimenez, J. L. Gutierrez-Rivas, E. Marin-Lopez, M. Rodriguez-Alvarez, and J. Diaz, "Time as a service based on white rabbit for finance applications," *IEEE Communications Magazine*, vol. 58, no. 4, pp. 60–66, Apr. 2020. [Online]. Available: http://dx.doi.org/10.1109/MCOM.001.1900602

[83] J. Lopez-Jimenez, M. Jimenez-Lopez, J. Diaz, and J. L. Gutierrez-Rivas, "White-rabbit-enabled data acquisition system," pp. 410–416, July 2017. [Online]. Available: http://dx.doi.org/10.1109/FCS.2017.8088907

[84] W. Hennig and S. Hoover, "White rabbit time synchronization for radiation detector readout electronics," *IEEE Transactions on Nuclear Science*, vol. 68, no. 8, pp. 2059–2065, August 2021. [Online]. Available: http://dx.doi.org/10.1109/TNS.2021.3085242

[85] W. Pan, Q. Du, and G. Gong, "Development of a white rabbit interface for synchronous data acquisition and timing control," pp. 1–3, June 2012. [Online]. Available: http://dx.doi.org/10.1109/RTC.2012.6418218

[86] S. Vitturi, C. Zunino, and T. Sauter, "Industrial communication systems and their future challenges: Next-generation ethernet, iiot, and 5g," *Proceedings of the IEEE*, vol. 107, no. 6, pp. 944–961, June 2019. [Online]. Available: http://dx.doi.org/10.1109/JPROC.2019.2913443

[87] L. Lo Bello, "Novel trends in automotive networks: A perspective on ethernet and the ieee audio video bridging," September 2014. [Online]. Available: http://dx.doi.org/10.1109/ETFA.2014.7005251

[88] T. Zhang, G. Wang, C. Xue, J. Wang, and M. Nixon, "Time-sensitive networking (tsn) for industrial automation: Current advances and future directions," *ACM Computing Surveys*, vol. 57, no. 2, pp. 1–38, October 2024. [Online]. Available: http://dx.doi.org/10.1145/3695248

[89] J. Pfrommer, A. Ebner, S. Ravikumar, and B. Karunakaran, "Open source opc ua pubsub over tsn for realtime industrial communication," in *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, September 2018. [Online]. Available: http://dx.doi.org/10.1109/ETFA.2018.8502479

[90] "IEEE Standard for Local and Metropolitan Area Networks—Timing and Synchronization for Time-Sensitive Applications (IEEE 802.1AS-2020)," IEEE, Tech. Rep. 802.1AS-2020, 2020, defines timing and synchronization mechanisms for real-time Ethernet-based networks. [Online]. Available: http://dx.doi.org/10.1109/IEEESTD.2020.9121845

[91] S. Rodrigues and J. Lv, "Synchronization in time-sensitive networking: An introduction to ieee std 802.1as," *IEEE Communications Standards Magazine*, vol. 6, no. 4, pp. 14–20, Dec. 2022. [Online]. Available: http://dx.doi.org/10.1109/MCOMSTD.0001.2200038

[92] P. Liu, R. Yang, S. Liu, and Q. He, "Design and implementation of best master clock selection algorithm based on fpga," in *2022 International Conference on Industrial IoT, Big Data and Supply Chain (IIoTBDSC)*. IEEE, September 2022, pp. 1–5. [Online]. Available: http://dx.doi.org/10.1109/IIoTBDSC57192.2022.00011

[93] P. Meyer, T. Steinbach, F. Korf, and T. C. Schmidt, "Extending ieee 802.1 avb with time-triggered scheduling: A simulation study of the coexistence of synchronous and asynchronous traffic," pp. 47–54, Dec. 2013. [Online]. Available: http://dx.doi.org/10.1109/VNC.2013.6737589

[94] D. Thiele, R. Ernst, and J. Diemer, "Formal worst-case timing analysis of ethernet tsn's time-aware and peristaltic shapers," pp. 251–258, Dec. 2015. [Online]. Available: http://dx.doi.org/10.1109/VNC.2015.7385584

[95] M. Máté, C. Simon, and M. Maliosz, "Asynchronous time-aware shaper for time-sensitive networking," *Journal of Network and Systems Management*, vol. 30, no. 4, Sep. 2022. [Online]. Available: http://dx.doi.org/10.1007/s10922-022-09688-y

[96] F. Song, L. Li, I. You, and H. Zhang, "Enabling heterogeneous deterministic networks with smart collaborative theory," *IEEE Network*, vol. 35, no. 3, pp. 64–71, May 2021. [Online]. Available: http://dx.doi.org/10.1109/MNET.011.2000613

[97] S. Bhattacharjee, K. Katsalis, O. Arouk, R. Schmidt, and T. Wang, "Network slicing for tsn-based transport networks," *IEEE Access*, vol. 9, pp. 62 788–62 809, 2021. [Online]. Available: http://dx.doi.org/10.1109/ACCESS.2021.3074802

[98] P. Kehl, J. Ansari, M. H. Jafari, P. Becker, and J. Sachs, "Prototype of 5g integrated with tsn for edge-controlled mobile robotics," *Electronics*, vol. 11, no. 11, p. 1666, may 2022. [Online]. Available: http://dx.doi.org/10.3390/electronics11111666

[99] N. Berger, "The mu3e experiment," *Nuclear Physics B - Proceedings Supplements*, vol. 248–250, p. 35–40, mar 2014. [Online]. Available: http://dx.doi.org/10.1016/j.nuclphysbps.2014.02.007

[100] S. Bachmann, N. Berger, A. Blondel, S. Bravar, and A. Buniatyan, "The proposed trigger-less tbit/s readout for the mu3e experiment," *Journal of Instrumentation*, vol. 9, no. 01, pp. C01 011–C01 011, jan 2014. [Online]. Available: http://dx.doi.org/10.1088/1748-0221/9/01/C01011

[101] P. Durante, N. Neufeld, R. Schwemmer, U. Marconi, and G. Balbi, "100 gbps pci-express readout for the lhcb upgrade," *IEEE Transactions on Nuclear Science*, vol. 62, no. 4, pp. 1752–1757, aug 2015. [Online]. Available: http://dx.doi.org/10.1109/TNS.2015.2441633

[102] K. Arndt, H. Augustin, P. Baesso, N. Berger, F. Berg *et al.*, "Technical design of the phase i mu3e experiment," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 1014, p. 165679, Oct. 2021. [Online]. Available: http://dx.doi.org/10.1016/j.nima.2021.165679

[103] CERN, Geneva, LHC Experiments Committee, "Lhcb trigger and online upgrade technical design report," CERN Document Server, Tech. Rep., 2014, includes details of the trigger-less readout system and full software trigger capabilities. [Online]. Available: http://cds.cern.ch/record/1701361

[104] M. Feo, F. Alessio, P. Durante, L. Granado Cardoso, and G. Vouters, "The real-time system for distribution of clock, control and monitoring commands with fixed latency of the lhcb experiment at cern," *IEEE Transactions on Nuclear Science*, vol. 70, no. 6, pp. 985–992, Jun. 2023. [Online]. Available: http://dx.doi.org/10.1109/TNS.2023.3273086

[105] E. Mendes, S. Baron, C. Soos, and F. Vasey, "A timing, trigger, and control system with picosecond precision based on 10 gbit/s passive optical networks for high-energy physics," *IEEE Transactions on Nuclear Science*, vol. 68, no. 4, pp. 447–457, Apr. 2021. [Online]. Available: http://dx.doi.org/10.1109/TNS.2021.3065128

[106] J. Mitra, E. David, E. Mendez, S. A. Khan, and T. Kiss, "Trigger and timing distributions using the ttc-pon and gbt bridge connection in alice for the lhc run 3 upgrade," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 922, pp. 119–133, Apr. 2019. [Online]. Available: http://dx.doi.org/10.1016/j.nima.2018.12.076

[107] J. Kvapil, A. Bhasin, M. Bombara, D. Evans, and A. Jusko, "Alice central trigger system for lhc run 3," *EPJ Web of Conferences*, vol. 251, p. 04022, 2021. [Online]. Available: http://dx.doi.org/10.1051/epjconf/202125104022

[108] B. Abi, R. Acciarri, M. A. Acero *et al.*, "Volume iv. the dune far detector single-phase technology," *Journal of Instrumentation*, vol. 15, no. 08, pp. T08 010–T08 010, Aug. 2020. [Online]. Available: http://dx.doi.org/10.1088/1748-0221/15/08/T08010

[109] A. Barcock, D. Cussans, D. Lindebaum, D. Newbold, S. Paramesvaran, and S. Trilov, "Timing and synchronization of the dune neutrino detector," *Journal of Instrumentation*, vol. 18, no. 01, p. C01067, Jan. 2023. [Online]. Available: http://dx.doi.org/10.1088/1748-0221/18/01/C01067

[110] S. M. Trilov, "Timing and synchronisation of the dune far detector," p. 077, Apr. 2025. [Online]. Available: http://dx.doi.org/10.22323/1.468.0077

[111] A. Rost *et al.*, "Performance of the cvd diamond based beam quality monitoring system in the hades experiment at gsi*," in *Proc. 10th International Particle Accelerator Conference (IPAC'19), Melbourne, Australia, 19-24 May 2019*, ser. International Particle Accelerator Conference, no. 10. Geneva, Switzerland: JACoW Publishing, Jun.

2019, paper WEPGW019, pp. 2507–2509, https://doi.org/10.18429/JACoW-IPAC2019-WEPGW019. [Online]. Available: http://jacow.org/ipac2019/papers/wepgw019.pdf

[112] K. Erciyes, *Distributed Real-Time Systems*. Cham: Springer International Publishing, 2019, pp. 41–62. [Online]. Available: https://doi.org/10.1007/978-3-030-22570-4_3

[113] B. A. Forouzan, *Data Communications and Networking*, 5th ed. New York, NY: McGraw-Hill Professional, feb 2012.

[114] N. Suhaimy, N. A. M. Radzi, W. S. H. M. W. Ahmad, K. H. M. Azmi, and M. A. Hannan, "Current and future communication solutions for smart grids: A review," *IEEE Access*, vol. 10, pp. 43 639–43 668, 2022.

[115] I. Serban, S. Céspedes, C. Marinescu, C. A. Azurdia-Meza, and J. S. Gómez, "Communication requirements in microgrids: A practical survey," *IEEE Access*, vol. 8, pp. 47 694–47 712, 2020.

[116] M. Rico, J.-P. Aubry, C. Botteron, and P.-A. Farine, "Ns-level time transfer over a microwave link using the ptp-wr protocol," in *2015 Joint Conference of the IEEE International Frequency Control Symposium & the European Frequency and Time Forum*, 2015, pp. 690–695.

[117] J. E. Gilligan, E. M. Konitzer, E. Siman-Tov, J. W. Zobel, and E. J. Adles, "White rabbit time and frequency transfer over wireless millimeter-wave carriers," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 67, no. 9, pp. 1946–1952, 2020.

[118] M. Lipiński, T. Włostowski, J. Serrano, and P. Alvarez, "White rabbit: a ptp application for robust sub-nanosecond synchronization," in *2011 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, 2011, pp. 25–30.

[119] L. D. Meder, "Timing synchronization and fast-control for fpga-based large-scale readout and processing systems," Ph.D. dissertation, Karlsruher Institut für Technologie (KIT), 2017.

[120] P. Moreira, P. Alvarez, J. Serrano, I. Darwezeh, and T. Wlostowski, "Digital dual mixer time difference for sub-nanosecond time synchronization in ethernet," in *2010 IEEE International Frequency Control Symposium*, 2010, pp. 449–453.

[121] P.-A. Loizeau, D. Emscherman, J. Lehnert, W. F. J. Müller, and J. Yang, "The prototype readout chain for cbm using the afck board and its software components," in *Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments 2015*, R. S. Romaniuk, Ed., vol. 9662, International Society for Optics and Photonics. SPIE, 2015, p. 96622X. [Online]. Available: https://doi.org/10.1117/12.2205820

[122] C. G. Larrea, K. Harder, D. Newbold, D. Sankey, A. Rose *et al.*, "Ipbus: a flexible ethernet-based control system for xtca hardware," *Journal of Instrumentation*, vol. 10, no. 02, p. C02019, feb 2015. [Online]. Available: https://dx.doi.org/10.1088/1748-0221/10/02/C02019

[123] D. Taylor, M. Klein, and V. Vendramini, "All Digital VCXO Replacement for Gigabit Transceiver Applications (7 Series/Zynq-7000) (XAPP589)," AMD (formerly Xilinx), San Jose, CA, Application Note XAPP589 (v2.3.1), oct 2021. [Online]. Available: https://docs.amd.com/v/u/en-US/xapp589-VCXO

[124] K. Chen, H. Chen, J. Huang, F. Lanni, and S. Tang, "A generic high bandwidth data acquisition card for physics experiments," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 7, pp. 4569–4577, 2020.

[125] Xilinx, "Stacked silicon interconnect technology delivers breakthrough fpga capacity, bandwidth, and power efficiency," Xilinx Inc., San Jose, CA, White Paper WP380, dec 2012, document ID WP380; revision 1.2. [Online]. Available: https://www.xilinx.com/support/documentation/white_papers/wp380_Stacked_Silicon_Interconnect_Technology.pdf

[126] W. M. Zabołotny, M. Gumiński, M. Kruszewski, and W. F. J. Müller, "Control and diagnostics system generator for complex fpga-based

measurement systems," *Sensors*, vol. 21, no. 21, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/21/7378

[127] J. Hoffmann, *PEXARIA5 General Description.*, https://www.gsi.de/fileadmin/EE/Module/EXPLODER/pexaria5_14.pdf, 2013, gSI internal technical manual.

[128] P. Moreira *et al.*, "The gbt project," in *Topical Workshop on Electronics for Particle Physics*. CERN, 2009.

[129] P. Moreira, S. Kulis, S. Baron, S. Biereigel, and E. B. D. S. Mendes, "lpgbt: Low-power radiation-hard multipurpose high-speed transceiver asic for high-energy physics experiments," *IEEE Transactions on Nuclear Science*, vol. 72, no. 1, pp. 24–37, 2025.

[130] M. B. Marin, S. Baron, S. S. Feger, P. Leitao, E. S. Lupu *et al.*, "The gbt-fpga core: features and challenges," *Journal of Instrumentation*, vol. 10, no. 03, p. C03021, mar 2015. [Online]. Available: https://dx.doi.org/10.1088/1748-0221/10/03/C03021

[131] J. M. Mendez, S. Baron, S. Kulis, and J. Fonseca, "New lpgbt-fpga ip: Simulation model and first implementation," in *Proceedings of Topical Workshop on Electronics for Particle Physics — PoS(TWEPP2018)*, vol. 343, 2019, p. 059.

[132] P. Moreira, J. Christiansen, K. Wyllie, P. Moreira, S. Baron *et al.*, *GBTX manual*, 2021. [Online]. Available: https://cds.cern.ch/record/2809057