Front-End Readout Electronics for the ALICE CPV and HMPID Particle Detectors

Clive Seguna

Department of Microelectronics and Nanoelectronics

University of Malta

February 2022

Submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy (Microelectronics and Nanoelectronics)

TESS

TERTIARY EDUCATION
SCHOLARSHIPS SCHEME

The research work disclosed in this publication is partially funded by the Tertiary Education Scholarships Scheme (TESS) Malta 2015, financed by the Ministry for Education and Employment in Malta. TESS seeks to contribute towards the Maltese labour market and economy.

Dedicated to my family

COPYRIGHT NOTICE

UNIVERSITY OF MALTA

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

SUBMISSION OF DISSERTATION FOR EXAMINATION

Student I.D. **75379M**

Student Name **Clive Seguna**

Course **Doctor of Philosophy**

Title of work submitted

Front-End Readout Electronics for the ALICE CPV and HMPID Particle Detectors

I am hereby submitting my dissertation for examination by the Board of Examiners.

________________________ February 2022

Signature of Student *Date*

Submission noted.

Prof. Ing. Edward Gatt ________________________

Principal Supervisor *Signature*

February 2022

Date

UNIVERSITY OF MALTA

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

DECLARATIONS BY POSTGRADUATE STUDENTS

Student I.D. **75379M**

Student Name **Clive Seguna**

Course **Doctor of Philosophy**

Title of work submitted

Front-End Readout Electronics for the ALICE CPV and HMPID Particle Detectors

(a) Authenticity of Dissertation

I hereby declare that I am the legitimate author of this thesis and that it is my original work.

No portion of this work has been submitted in support of an application for another degree or qualification of this or any other university or institution of higher education,

I hold the University of Malta harmless against any third party claims with regard to copyright violation, breach of confidentiality, defamation and any other third party right infringement.

(b) Research Code of Practice

As a Ph.D. student, as per Regulation 49 of the Doctor of Philosophy Regulations, I accept that my thesis be made publicly available on the University of Malta Institutional Repository.

__________________________ February 2022

Signature of Student *Date*

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

DECLARATION

Plagiarism is defined as "the unacknowledged use, as one's own work, of work of another person, whether or not such work has been published" (Regulations Governing Conduct at Examinations, 1997, Regulation 1 (viii), University of Malta).

I, the undersigned, declare that this thesis submitted is my work, except where acknowledged and referenced.

I understand that the penalties for making a false declaration may include, but are not limited to, loss of marks; cancellation of examination results; enforced suspension of studies; or expulsion from the degree program.

Work submitted without this signed declaration will not be corrected and will be given zero marks.

Clive Seguna ______________________

Student Name *Signature*

Front-End Readout Electronics for the ALICE CPV and HMPID Particle Detectors

Title of work submitted

February 2022

Date

ABSTRACT

This work, carried out in collaboration with the European Council for Nuclear Research (CERN) and the University of Malta, presents the development of a new electronic front-end readout system for the High momentum particle identification (HMPID) and Charged Particle Veto (CPV) detectors. The upgrade strategy of the A Large Ion Collider Experiment (ALICE) is based on the collection of more than 10 nb^{-1} Pb-Pb collisions at a luminosity of 6x1027 $cm^{-2}s^{-1}$, corresponding to a collision rate of 50 kHz for Pb-Pb and 200 kHz for pp and p-Pb. The requirements for such a high beam luminosity cannot be met with the existing CPV electronics, which had a low readout rate of 5 kHz. The development of such a system is a challenging task. Therefore, different technologies and architectural topologies were considered and investigated for the optimization of the front-end readout electronics. This work contributed to the development of a new custom front-end readout electronics system architecture for the CPV detector module in the PHOton spectrometer (PHOS). This newly developed electronics were commissioned and accepted by the Russian Institute of High Energy Physics and the ALICE collaboration for installation in November 2020. Compared to previous systems, the proposed new architecture allows parallel readout and processing of all 480 silicon photomultiplier pads connected to digital signal processing boards. Optimization strategies include the use of 28 nm FPGA technology with high pin count and low power consumption for simultaneous readout of digital signal processors, referred to as 5 DIL boards, and the use of high-speed 3.125 Gbps transceiver interconnects. In addition, the newly developed FPGA firmware architecture has helped increase the event readout rate and data throughput by a factor of ten. This work enables both the CPV and HMPID detectors to achieve an interaction rate of at least 50 kHz. The system design consists of three modules, each containing two segment cards, two readout common boards (RCBs), and 20 digital signal processors. Five processors are grouped on an electronic board called 5- DIL board. This report presents the architectural layout and preliminary results of performance measurements for the proposed new design. In addition, this work has contributed to the development of a new ASIC chip that integrates four digital signal processors, error correction and detection circuitry, and four serial transmitters with a bandwidth of at least 0.5 Gbps. The ASIC chip implementation uses XFAB-180-nm technology and is capable of processing at least 192 analogue channels simultaneously. The developed ASIC device can be easily integrated into current CPV and similar electronic readout circuits for physics particle detectors, which helps reduce the required number of electronic components and PCB manufacturing costs. This work concludes with recommendations for further planned updates to the hardware scheme.

ACKNOWLEDGEMENTS

First, I would like to thank my thesis advisor, Prof. Ing. Edward Gatt. Without the valuable technical guidance and support from my thesis advisor, this work would not have been possible. I would also like to thank Dr. Giacinto De Cataldo and Dr. Yuri Kharlov, who were very supportive and helpful in assigning me this project at CERN. My special thanks to my family, especially my wife, for their encouragement, support, and patience throughout this journey. These six years would not have been possible without the encouragement and support of all these people. This work was made possible through a collaboration between the Department of Microelectronics and Nanoelectronics at the University of Malta and ALICE CPV of the Conseil Europeen pour la Recherche Nucleaire (CERN).

LIST OF PUBLICATIONS

Sections of the work presented in this thesis were published in four peer reviewed conference proceedings papers and one journal paper:

Journal Publication

1. C.Seguna, E. Gatt, et.al, “Development of Front-End Readout Electronics System for the ALICE HMPID and Charged-Particle Veto Detector”, International Journal on Advances in Systems and Measurements, vol 13, pp. 1 -10, 2020.

Conference Publications

2. C.Seguna, E. Gatt, Giacinto De Cataldo, I. Grech, O. Casha, “Proposal for a new ALICE CPV-HMPID front-end electronics topology”, 2017 13th Conference on Ph.D. Research in Microelectronics and Electronics (PRIME), pp. 176-176, 2017

3. C.Seguna, E. Gatt, Giacinto De Cataldo, I. Grech, O. Casha, “A New FPGA-Based Controller Card for the Optimisation of the Front-End Readout Electronics of Charged-Particle Veto Detector at ALICE”, 2018 New Generation of CAS (NGCAS), pp. 45-48, 2018.

4. C.Seguna, E.Gatt, Giacinto De Cataldo, I.Grech, O. Casha, “A New Front-End Readout Electronics for the ALICE Charged-Particle Veto Detector”, 2018 11th International Conference on Advances in Circuits, Electronics and Micro-electronics (CENICS 2018), pp.11-18,2018.

5. C. Seguna, E. Gatt, I. Grech, O. Casha and G. De Cataldo, "Development of a New ASIC based, Multi-channel Data Acquisition and Real-Time Processing System," 2021 22nd IEEE International Conference on Industrial Technology (ICIT), 2021, pp. 779-782, doi: 10.1109/ICIT46573.2021.9453615.

Conference Publications – Under Review – Submitted February 2022

6. C.Seguna, E. Gatt, Giacinto De Cataldo, I. Grech, O. Casha, "An ASIC Data Readout Processor for the ALICE HMPID and Charged Particle Veto Detectors,”, 2022 Conference on Ph.D. Research in Microelectronics and Electronics (PRIME), 2022.

7. C.Seguna, E. Gatt, Giacinto De Cataldo, I. Grech, O. Casha, “Testbench development for the New ALICE Charged Particle Veto Detector Readout System”, 2022 New Generation of CAS (NGCAS), 2022.

8. C.Seguna, E. Gatt, Giacinto De Cataldo, I. Grech, O. Casha, “Design of a Gigabit Multi-Channel ASIC Serializer for High-Speed Data Transmission”, 2022 New Generation of CAS (NGCAS), 2022.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

ADC Analogue-to-Digital Converter

ALICE A Large Ion Collider Experiment

ASIC Application Specific Integrated Circuit BCID Bunch Crossing Identification

CERN Conseil Européen pour la Recherche Nucléaire

CPV Charge Particle Veto

CTP Central Trigger Processor

DAQ Data Acquisition

DDL Detector Data Link

FEE Front End Electronics

GBTX Giga-bit Transceiver

HMPID High Momentum Particle Identification Detector

L0 Level-Zero Trigger

L1 Level-One Trigger

LHC Large Hadron Collider

LM Level-Minus

LTU Local Trigger Unit

LUT Look-up Table

LVDS Low-Voltage Differential Signalling

MWPC Multiwire Proportional Chambers

MDART Multi-channel Data Acquisition and Real-Time Processing System

PCB Printed Circuit Board

RCB Readout Control Board

RDH Raw Data Header

SIU Source Interface Unit

TTC Timing, Trigger, and Control

TTCRx Timing, Trigger, and Control

CHAPTER 1 - Introduction

1.0 ALICE Experiment and LHC

The ALICE experiment at the Large Hadron Collider (LHC) on the French-Swiss border in Geneva is designed to study the behaviour of matter and to collect comparative data on heavy-ion and proton-proton collisions in heavy-ion physics at high energy levels. The LHC has a circumference of about 27 km and is located 100 m below ground. The LHC consists of two tubes surrounded by superconducting electromagnets that carry two beams of particles travelling in opposite directions at nearly the speed of light. These two particle beams travelling in opposite directions can be brought together at various interaction points, one of which is the ALICE experiment.

The grouped bundles of particle beams are spaced about 25 ns apart. Therefore, a 40 MHz clock is distributed and used as the main frequency for the operation of the various electronic systems in the LHC. The current system leaves open physics questions that include hadronization, nuclei, long-range capability correlations, and the small X-proton structure [1]. The ALICE detector was built to study these types of heavy ion collisions [2][3].

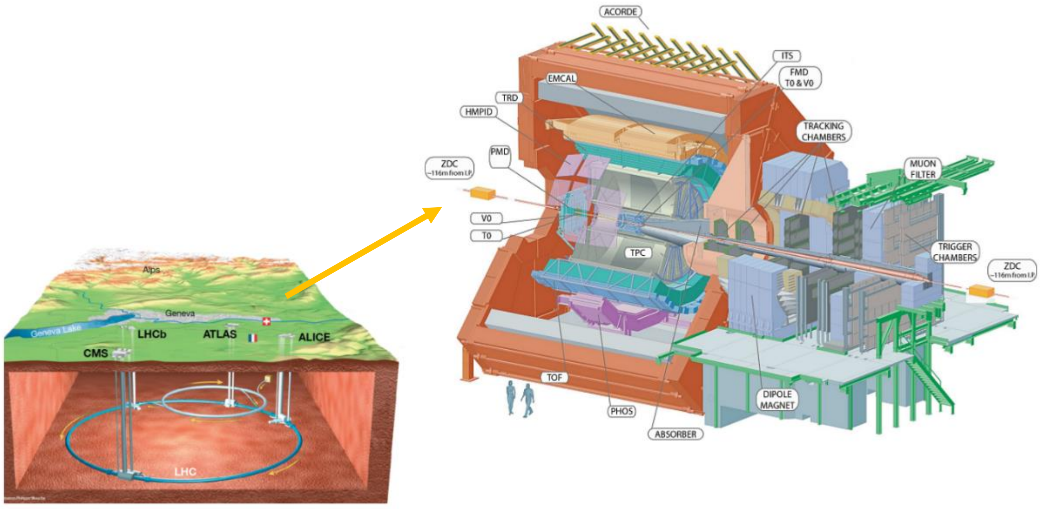


Figure 1: The LHC ALICE underground location and layout illustration of the various ALICE sub-detectors [1].

As part of the ALICE upgrade physics programme and to further investigate the theory behind quantum chromoDynamics (QCD), including at high temperatures, the detector performance of

ALICE needs to be improved to enable the detailed study of new phenomena and interactions of matter at high temperatures. To achieve such goals and to understand the various changing aspects of the QCD phases described above, highly statistical, and precise measurements are required. Given such complex measurements, there is a need for specific triggering methods. Therefore, to improve particle tracking, low-momentum vertexing, and data acquisition at much higher data rates, the current detector must be significantly upgraded. The ALICE experiment consists of 14 different subdetectors. This research is mainly focused on upgrading the readout Front-End Electronics (FEE) of the ALICE Charged Particle Veto Detector (CPV) to handle a tenfold increase in Pb-Pb data collision rates using either a continuous or triggered readout mode and will therefore be replaced by this newly developed system.

1.1 Charged Particle Veto Detector

The Charged Particle Veto detector is designed to suppress the detection of charged particles impinging on the front surface of the PHOton spectrometer (PHOS), which consists of 17,920 lead tungstate crystal-based analogue detection channels. The CPV consists of three separate modules, each with an area of 200 x 230 cm^2, placed on top of a PHOS module.

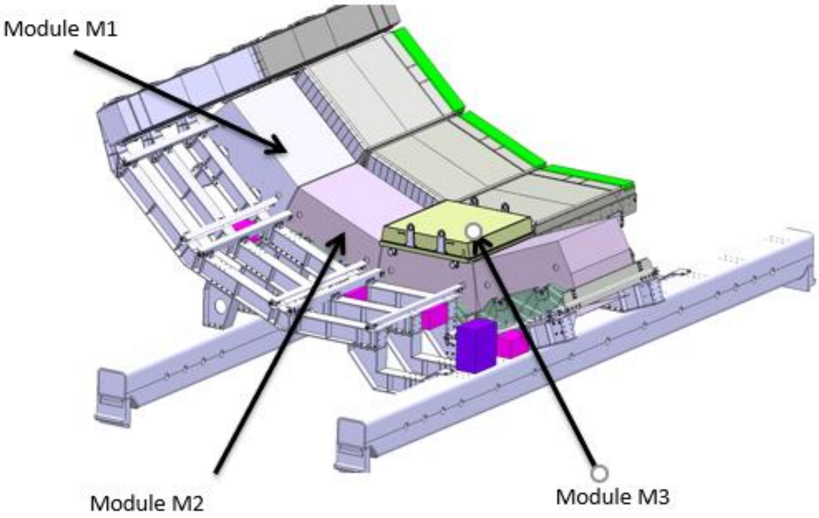


Figure 2: Three CPV Modules located in PHOS.

Data acquisition is digitised with 12-bit Analogue-to-Digital Converters (ADCs). The front-end electronic readout system consists of shapers, ADCs, preamplifiers with PIN photodiode, multiplexers, memory buffers, and Field-Programmable Gate Arrays (FPGAs).

The preamplifier is integrated with the PIN photodiode, which is optimised for measuring low energies of about up to 10GeV. PHOS plans to upgrade the readout firmware up to 30 - 40 kHz, CPV should be no slower than PHOS. A typical event size for the ALICE Charged Particle Veto Detector consists of 1.3 Kbytes for Pb-Pb particles. The maximum event readout rate that the CPV detector can currently achieve is 10 kHz at 1% occupancy [5]. Due to this technical limitation, a new FEE readout system is being developed that can detect more than 10 nb^{-1} Pb-Pb collisions at luminosities up to 6 x 10^{27} $\text{cm}^{-2}\text{s}^{-1}$. In addition, this new development for the front-end electronics is expected to help reduce the power dissipation and footprint of the current CPV detector FEE by more than 50%. Another goal of this new electronic architecture optimised for the CPV is to improve photon identification in the PHOton Spectrometer electromagnetic calorimeter (PHOS) by suppressing charged clusters, and the events recorded by PHOS and the CPV should be synchronised.

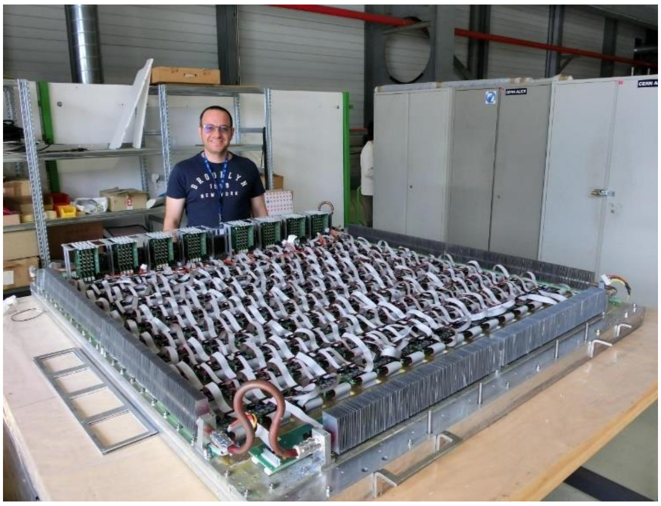


Figure 3: Illustration of one CPV module consisting of 7680 analogue channels, having an area of 200 x 230 cm^2.

In addition, the new CPV electronics is designed to correspond to the trigger-related electronic systems used in high-energy physics experiments (HEP), such as Timing Trigger and Control (TTC). The developed system can be easily used in other particle detectors such as the ALICE High Momentum Particle Identification Detector (HMPID). The 2019-2020 shutdown period provided an opportunity to upgrade CPV electronics to handle the new readout rates required for future runs. The newly developed FEE system will be installed in the Charged Particle Veto detector of the ALICE experiment. Run 3 started in 2021, after the planned 2019-2020 shutdown of the ALICE collaboration, to upgrade the current detector and further improve data acquisition rates to handle higher collision rates of 50 kHz for Pb-PB collisions with an expected peak luminosity of 6×10^{27} cm^{-2} s^{-1}. The proposed plan is to modify the ALICE detector to evaluate all interactions. This means that the CPV detector will need to be upgraded and the readout electronics modified to meet the stated requirements.

1.2 CPV Previous Electronics

The present ALICE CPV data acquisition is based on a trigger where fast subdetectors send a signal to slow detectors such as HMPID and CPV to start reading, controlling and monitoring data acquisition using multiple FEE and readout devices. The data collected by the readout modules is then transmitted to an online system to perform the necessary reconstruction and identify appropriate data that can later be recorded and processed offline.

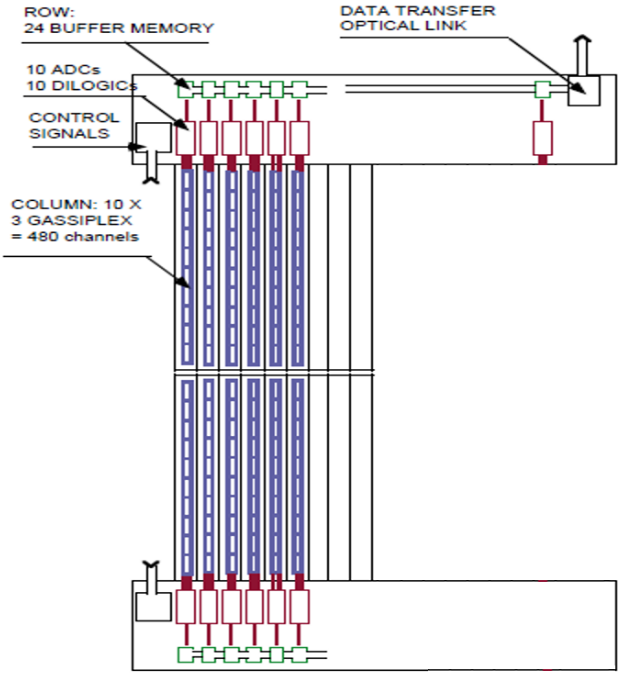


Figure 4: Schematics of implementation of one FE and RO electronics for one CPV module [2].

Each CPV module consists of 16 columns per module, with each column consisting of ten GASSIPLEX07 ASIC devices capable of handling 48 cathode readout lead tungsten pads. Each Gassiplex07 output is then processed by one and ADC and a DILOGIC signal processor with memory buffers and then remotely transmitted via an optical link as shown in Figure 4.

1.3 MDART ASIC

This research work also presents the development of a new custom cell-based Application Specific Integrated Circuit (ASIC) using XFAB-180nm technology to replace the functionality of the current FPGA-based electronic hardware suitable for the new experimental data acquisition requirements ALICE. This is because the required functions of the commercially available ASICs are not suitable for the CPV detector. The newly developed ASIC device called Multichannel Data Acquisition and Real-Time Processing System (MDART) with four integrated digital signal processor cores is designed to simultaneously process 192 12-bit digital channels, perform zero suppression, apply triple modular redundancy strategies with integrated cyclic redundancy checks (CRC), and transmit data over four low-voltage differential links (LVDS) transmission links at a minimum of 0.5 Gbit/s to the data acquisition system called Readout Common Board (RCB). Another task of this research is to determine the requirements, design, test, and verify the functionality of the MDART digital processing unit to replace the existing electronics in the current detector, reduce power consumption, reduce the number of electronic components for front-end readout by at least 50%, and meet the various data acquisition requirements of the CPV and HMPID particle detectors. The MDART's interfaces are designed to be easily integrated with various high-speed data acquisition systems.

1.3 Motivation and Contributions for the CPV FEEs Upgrade

The current trigger source for the CPV electronics is triggered directly by the Central Triggering Processor (CTP) and a Local Triggering Unit (LTU) via a Trigger and Timing Distribution Network (TTS). The LTU uses a fast serial trigger link protocol to transmit trigger and other timing information, while the CTP provides three trigger signals LM, L0, L1, and L2 that contribute to different latencies, the latter being a delayed copy of LM. The busy time is the period when the busy signal goes high, from the arrival of the L0 trigger to the end of the transmission of an event, which contributes to a waiting time of about 108 µs for the L2 trigger and the transmission time of the event data depending on the number of words. This long L2 delay contributes to the current trigger rate of approximately 4 kHz being ten times lower than the newly established target requirements. Thus, the current Front-end CPV and HMPID readout electronics are not suitable to achieve the required detector performance for Run 3. Therefore, a complete redesign of the FEE CPV readout electronics is required.

In addition, Chapters 3 and 4 present the design, implementation, and complete development of a new effective high-speed readout FEE system architecture for the CPV detector to meet the 50 kHz data acquisition requirements. The characterisation and verification of the MDART ASIC is then explained in Chapter 5. All development of the CPV FEE electronics was done in collaboration with the Russian Institute of High Energy Physics and the Department of Physics at the University of Bari in Italy. The Department of Microelectronics and Nanoelectronics at the University of Malta contributed to the development of the CPV and HMPID FEE readout electronics and the MDART ASIC device. This research was also partially funded by the Tertiary Education Scholarships Scheme (TESS) Malta 2015, funded by the Ministry of Education and Employment in Malta.

CHAPTER 2 – Literature Review

Particle detectors such as CPV and HMPID are needed to find and possibly identify all particles emerging from a scattering event. The development of such detectors for colliding-beam experiments presents several challenges. These include low power, minimising material consumption due to limited space and power, potentially high data rates, and reliability at certain radiation tolerance levels [7]. The current CPV and HMPID front-end readout electronics have been installed since 2014. However, the use of GASSIPLEX and DILOGIC crisps started even before that. These two crisps are the basic building blocks of the front-end and readout electronics, while a series of FPGAs generate the necessary control signals to interface with the experiment's data acquisition (DAQ) and trigger systems [8]. Since then, new low-power, more efficient, faster, and more accurate technologies have emerged. This section provides an overview of the existing system and presents the design requirements for the new CPV FEE readout electronics.

2.1 CPV Present System Architecture

The photon spectrometer PHOS is a lead tungsten calorimeter used to detect, identify, and measure the 4-momenta of photons. The CPV is a charged particle veto detector for photon identification located at PHOS and consists of a multi-wire proportional chamber (MWPC) with cathode readout. The CPV consists of three large-area modules of approximately 200 x 230 cm^2. Each CPV module consists of 16 columns per module,

- 10 cards per column, where each card consists of three ASIC charge amplifier signal condition chips named GASSIPLEX,
- 48 cathode readout pads per Gassiplex07-3 (3-GAS) card,
- 7680 channels of amplitude analysis per module using DILOGIC processors.

The 700 nm GASSIPLEX Application Specific Integrated Circuit (ASIC) [9][10] is tasked with performing analogue signal processing and conditioning. The GASSIPLEX board is the heart of the front-end electronics, which can be connected to either wire chambers or silicon strip detectors. It uses a switchable filter stage that can be configured to adapt to different input current shapes depending on the detector type. Each GASSIPLEX chip has sixteen analogue input channels, with each channel consisting of different function blocks. The first stage consists of a low-noise Charge Sensitive Amplifier (CSA) with a long delay to ensure complete charge capture from the detector's MWPC signal, followed by a deconvolution filter to compensate for the logarithmic shape of the charge signal and provide a quasi-stepping function through a long integration time. The deconvolution filter is designed to extract most of the charge delivered to the detector in only a few nanoseconds and to convert the long integration time of the charge signal into a quasi-step function. Low-pass and band-pass filters then follow the shaping circuitry to convert the quasi-step function into a half-Gaussian shape, taking one-third of the time to reach the peak and two-thirds to reach the baseline. Such a shaping procedure simplifies the measurement of the peak voltage and improves the signal-to-noise ratio (SNR). The shaper is then followed by a track-and-hold (T/H) circuit to store the analogue information at peak time.

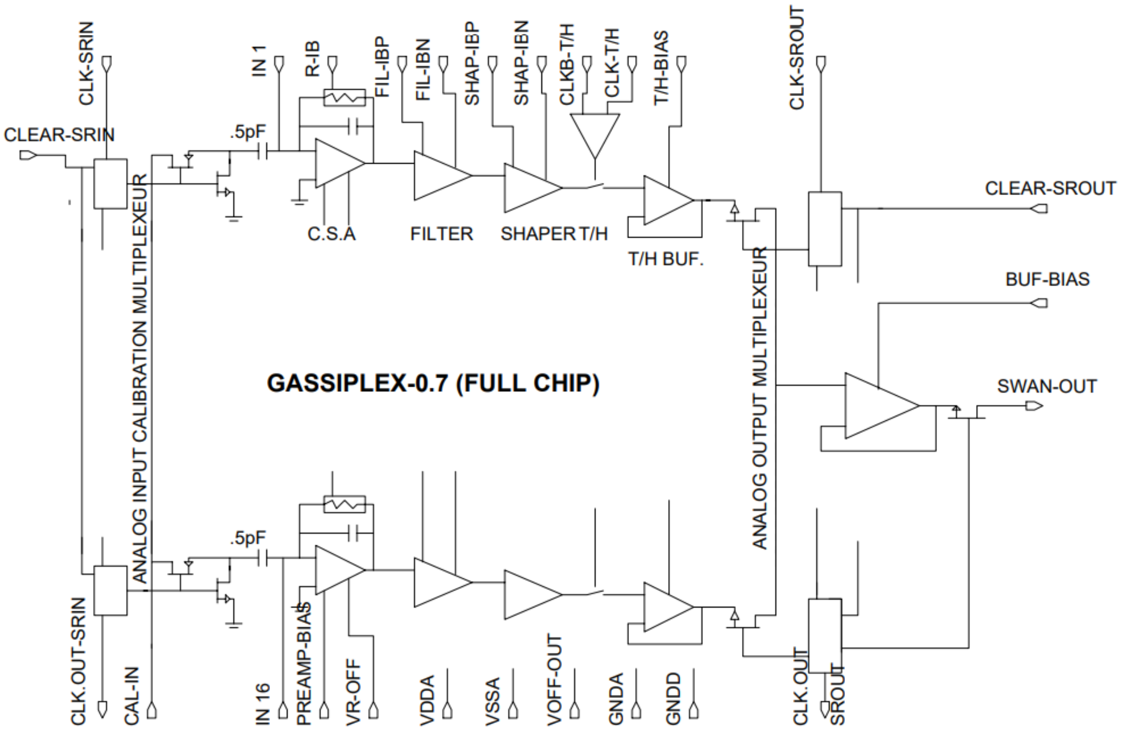


Figure 5: Gassiplex07 ASIC – Internal Functional blocks for one Analogue channel.

Depending on the application, the GASSIPLEX instrument can be operated in different power modes. For the CPV detector, the GASSIPLEX chip is operated in the gas detector mode, whose electrical characteristics are shown in Table 1.

Technology Silicon Area	ALCATEL-MIETEC 700nm CMOS 14.5 mm^2
Gain	2.2 mV/fC
Dynamic range (+)	560 fC (0 to 2 V)
Dynamic range (+)	300 fC (0 to -1.1 V)
Nonlinearity	+/- 2 fC
Noise at 0 pF	530 e^- rms
Noise slope	11.2 e^- rms/pF
Power consumption	8mW/channel at 10MHz
Peaking Time	1.2 µs
Analog readout speed	10 MHz (50 pF load)

Table 1. Electrical Characteristics for GASSIPLEX 700 nm ASIC device when operating in a Gaseous detector Mode.

Each 3-GAS card is connected directly to the back of the MWPC cathode. The sensitivity of each GASSIPLEX device is 3.6 mV/fC, with a dynamic range of more than 560 fC. The relatively long peak time (~1.2 µs) provides the delay required to make the level 0 (L0) trigger decision and thus store the captured analogue information in an internal capacitor. The sixteen capacitors are then multiplexed onto a common output line. Only three signals are required to operate the chip: the track-and-hold signal to shop the analogue signal, the clock sequence (one per channel) to multiplex onto the output line, and the clear signal to reset the chip. These are generated by the Column Controller (CC) card and distributed in parallel (one cable per ten 3- GAS cards) using flat cables. On a 3-GAS card, 3 chips (48 pads) are multiplexed onto the same output line, optimising the number of connectors, ADCs and DILOGIC chips required for signal processing. The 16 multiplexed analogue GASSIPLEX output channels are sampled by these 12-bit ADCs and later processed by the DILOGIC digital signal processor to eliminate empty channels and store the digitised data in memory buffers, which are later transmitted over an optical cable. Each DILOGIC can process up to 64 analogue input channels. A custom electronic board called 5-DIL consists of five analogue input channels processed by 12-bit analogue-to-digital converter modules (ADC) and five DILOGIC processors to process the digitised information. Each column contains 480 pads connected to two 5- DIL cards. A group of Field-Programmable Gate Arrays (FPGAs), called column or segment controllers, are used to process the signals from a column and provide the necessary interfaces with both the DAQ (Data Acquisition) and the Central Trigger Processor (CTP). A CPV module, as shown in Figure 6, consists of sixteen columns and 7680 channels for amplitude analysis.

The CPV Front-End (FE) and read-out (RO) electronics [6] measure and process the analogue signal induced at the cathode level of a MWPC divided into pads, allowing measurement of the localization of Cherenkov photons. The recorded pattern is transmitted to the experiment's data acquisition system for event generation and data logging after an online trigger selection. The interface to the DAQ is through the Readout and Control Board (RCB), which also contains the Timing, Trigger and Control Receiver (TTCRx) chip for trigger synchronisation. The RCB reads all fragments of the event stored on the Column Controller, creates a complete event with the DAQ format for ALICE and sends it to the DAQ via the Standard Interface Unit (SIU).

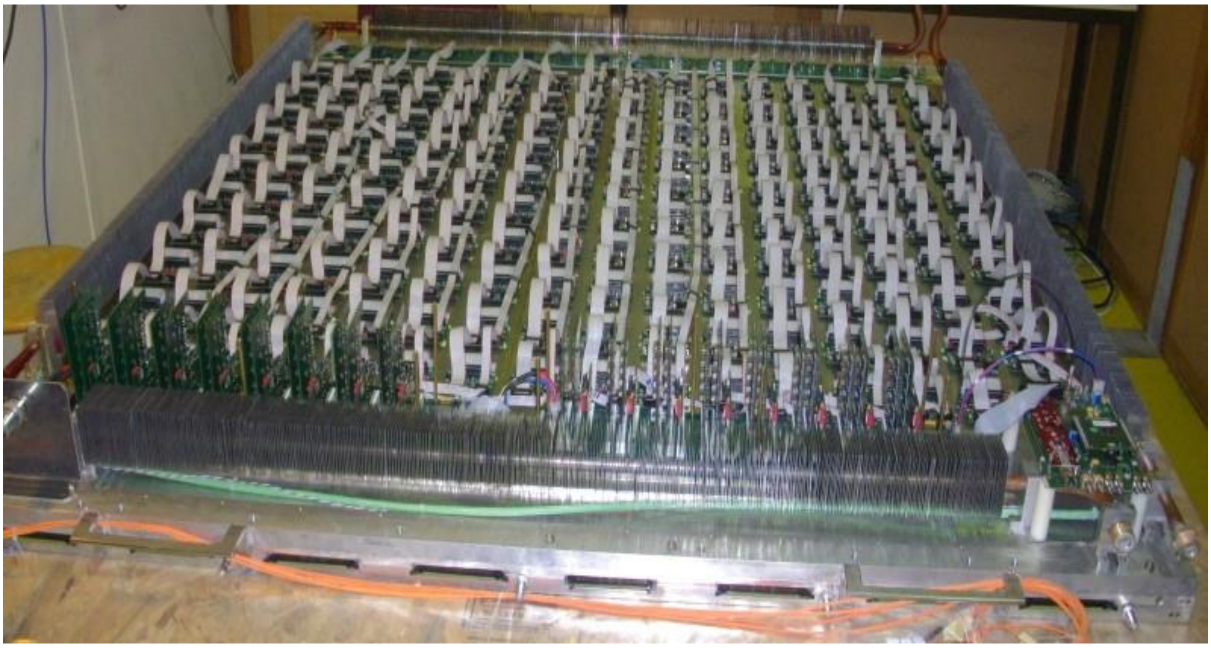


Figure 6: One CPV Module.

The number of channels per DILOGIC (in multiples of 16) and the multiplexing frequency (maximum 10 MHz, which corresponds to a readout time of 5 μs after which a new L0 can be accepted) are programmable parameters. The readout is based on the DILOGIC chip for data pre-processing and on the DDL link for the DAQ interface. The DDL is also used for configuration and control of the whole electronic chain. Each DIL-5 card input processes 48 3-GAS channels, which are then digitized by the 12-bit ADCs, while the DIL-5 card performs zero suppression, socket subtraction, data format and local storage of the selected data.

The DIL-5 processors of two cards (10 integrated circuits) are daisy-chained on the same 18-bit output bus, with the first 12 bits used to store the pulse height and the last 6 the channel address. The ten DIL -5 processors are sequentially emptied into a common bidirectional (first-in-first-out) FIFO located in the Field Programmable Gate Array (FPGA) of the CC card. This FPGA also supplies all the necessary control signals for reading out the DILOGIC and GASSIPLEX chips.

2.2 Mode of Operation

At power-up, an order of magnitude of 1000 events is collected through the FEE shown in Figure 7, with zero suppression turned off to measure pedestal levels. For each channel, the thresholds THR for each channel j are calculated as follows: THR(j) = Ped(j) + N(j)*Sig(j), where Ped(j) is the pedestal average for channel j, Sig(j) is the corresponding RMS value, and N(j) is a parameter usually set to 3. The pedestal and threshold table are downloaded to the DIL5 First-In-First-Out memory (FIFO) and finally zero suppression is turned on. The system is then ready for normal DAQ operation.

The L0 trigger is used to store the captured analogue information in the GASSIPLEX chip and start the multiplexing sequence. Then, the pulse height information for the channels above the threshold is stored in the internal DIL-5 FIFO (512 x 18-bit words) along with the corresponding address after pedestal subtraction. The data from two DIL-5 cards are collected in the CC FIFO. When the L2 signal arrives, the data is finally transferred to the experiment DAQ via the Detector Data Link (DDL).

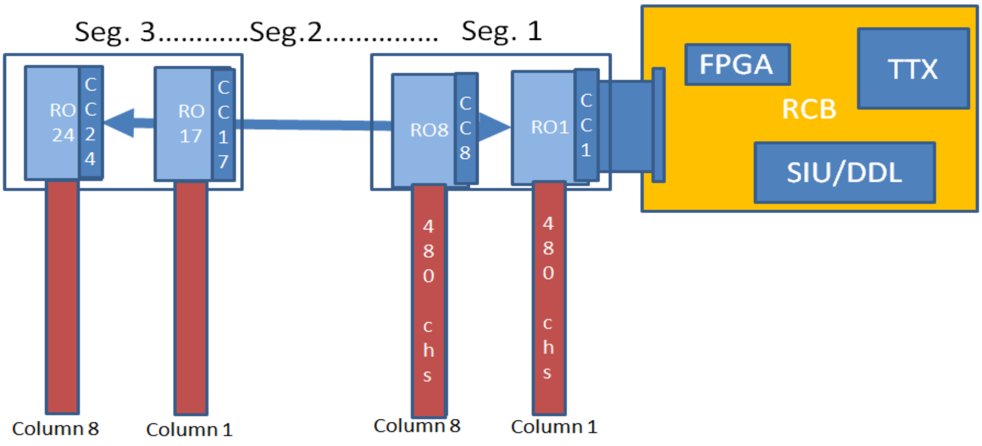


Figure 7: CPV RO present RO scheme.

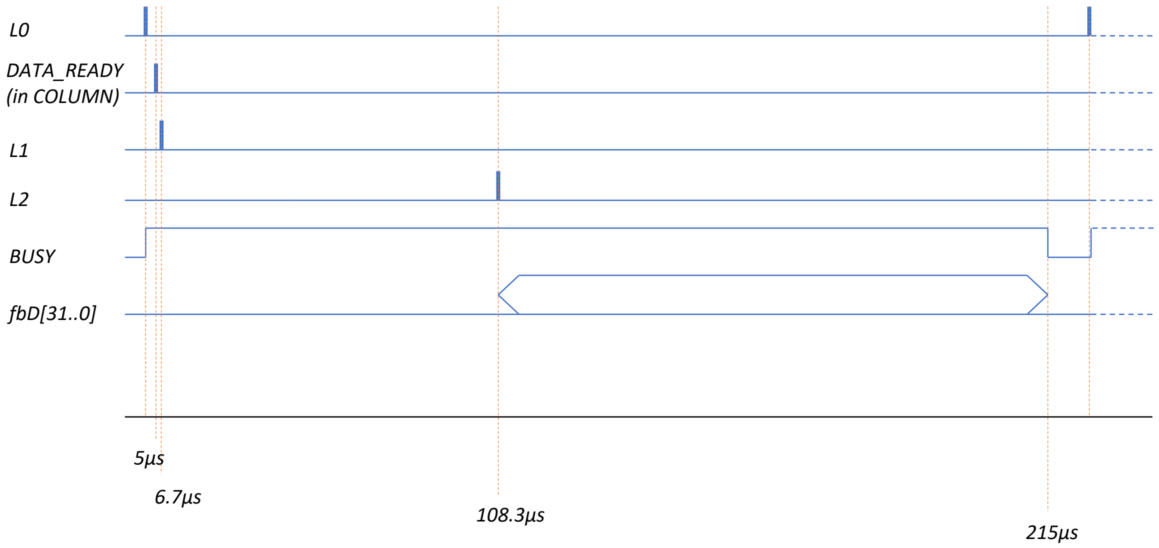


Figure 8: Timing Diagram for event Acquisition.

The busy time is the period during which the busy signal goes high, from the arrival of the L0 trigger to the end of the transmission of an event. Thus, it includes the waiting time of an L2 trigger (about 108 µs) and the transmission time that depends on the number of words. The data block that the electronics sends from RO to the DAQ contains a fixed and a variable part. The fixed part consists of headers and markers:

- CDH (10 words, 40 bytes),
- HMPID/CPV header (5 words, 20 bytes),
- Column headers (24 words, 96 bytes),
- Dilogic markers (240 words, 960 bytes),
- Segment markers (3 words, 12 bytes),

leading to a total data size of 1128 bytes to be transferred via DDL.

The variable part depends on the number of activated channels, where the read-out (RO) electronics spends two periods in the local state machine for transferring a word from a column to the SIU with additional 200 ns. The busy time conditions the event rate, which decreases with the number of words transferred, as shown in Figure 9. As described in [11], the time required to read out an event for a 15% occupancy of the central mode Pb + Pb in It can be observed that the current total readout time of 169 µs is far from the required target of 20 µs or less.

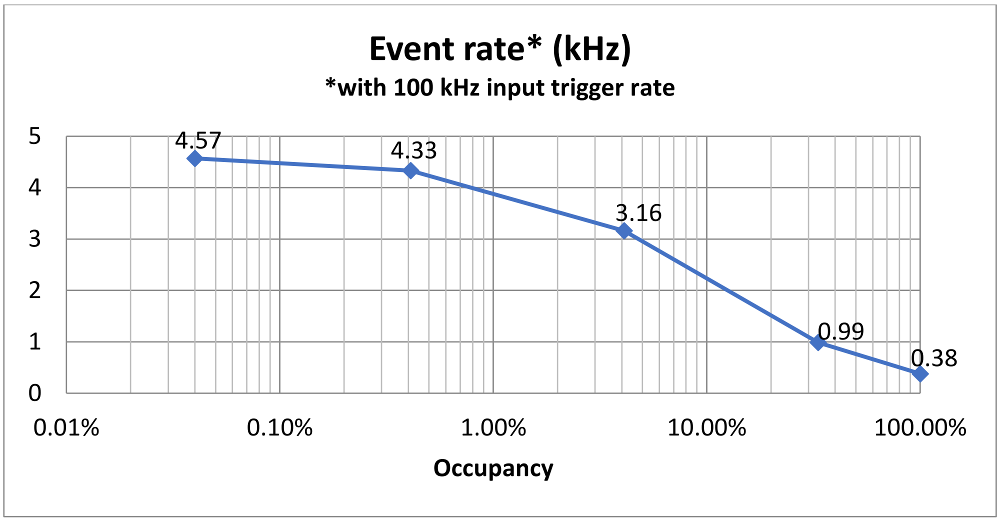


Figure 9: Busy Time versus Occupancy [3].

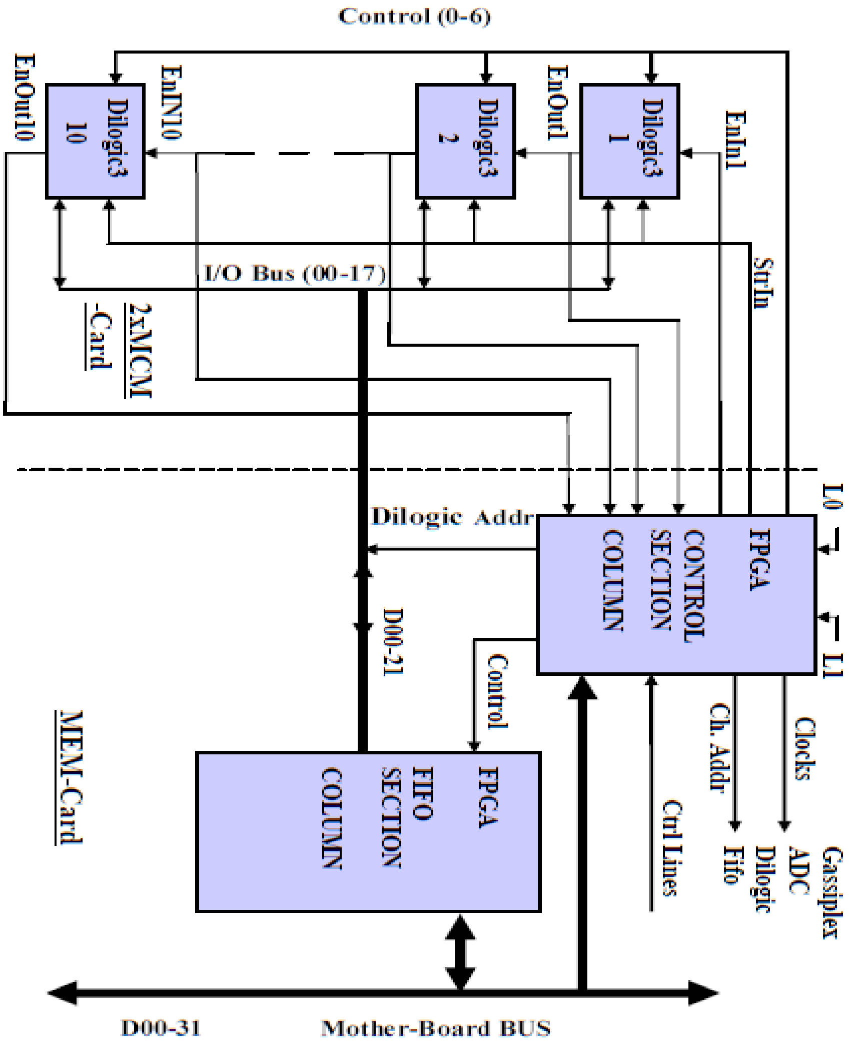


Figure 10: Block Diagram of Column Controller [2].

A complete block diagram for a 5-DIL card, illustrating the main components of the Front-end readout electronics is shown in Figure 10. An FPGA controller directs the DILOGIC processor to perform a specific operation via the 4-bit FCODE<3:0> interface. The following table illustrates the various functions that can be executed for each 4-bit code.

FCODE <3:0>	**Description**
0000	Front-end test mode
0001	Load "almost full" preset
0010 … 0111	No operation
1000	Pattern readout
1001	Pattern delete
1010	Analog readout
1011	Analog delete
1100	Reset fifo Pointers
1101	Reset daisy chain
1110	Configuration write
1111	Configuration read

Table 2. FCODE <3:0> Operation for DILOGIC processor [4].

The address counters of the data and bitmap FIFO memories can be reset with function code "1100" after a StrIn_N cycle has been executed, which clears the memory contents. Function code "0001" can be used to load a preset 9-bit register that can be used to compare the read and write address counters. When the write counter reaches the read counter minus the preset 9-bit value, the AlmostFull_N flag is set, indicating that the trigger flow should be terminated to prevent overwriting of the FIFO data. Function code "0000" can be used to enable the test mode where the amplitude and channel address data ChAddr < 5:0 > can be loaded from the INOUT < 17:12 > data pins, while the amplitudes can be loaded from the INOUT < 11:00 > pins.

The analogue signal is sampled by the GASSIPLEX chip when it receives the L0 trigger. L1 signal initiates a multiplexing sequence, so to store the channel pulse height above pedestal value in the

5-DIL card FIFO (512 x 18-bit words) memory along with the corresponding address after threshold subtraction. In the end, when the L2 signal arrives, the data is transmitted to the experiment DAQ via the DDL link. An End Event Word (EEW) generated internally by the 5-DIL card identifies the data associated with various event formation triggers. The EEW also contains the number of event words and the event number.

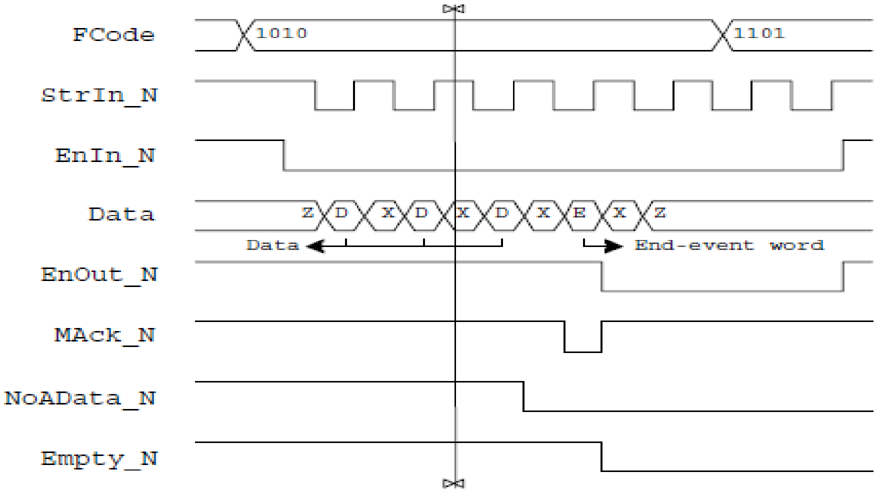


Figure 11: Timing Diagram for Column Controller.

Each DILOGIC chip is set to analogue readout mode when the function code is set to "1010" and EnIn_N is low. Successive StrIn_N cycles cause all modules in the chain to place their digitised data on the data bus in sequence, starting with the first module in the chain. An enable signal is passed from the EnOut_N pin to the EnIn_N pin of the next chip when the module has placed its analogue data of an event on the data bus. The Mack_N pin indicates the occurrence of the end event on the data bus and marks the end of the analogue readout on this 5-DILOGIC card. At this moment, the EnOut_N pin is pulled low to start reading out the next chip, and Mack_N goes high again. The analogue readout process is finished when the EnOut_N of the last chip goes low. All 5-DILOGIC card ICs must be reset by applying the daisy chain reset code "1101" and an additional Strin_N strobe. This will reset the EnOut_N pins of all crisps in the chain, otherwise they will remain in analogue readout mode. The analogue data FIFO has two flags, the Empty_N flag indicates that the FIFO is completely empty and the NoAData_N indicates that there is no analogue information on the FIFO, only end event words. This status occurs when the 5-DILOGIC has received triggers with empty events. The function code allows the events in the FIFO, starting with the first event to be read, to be deleted.

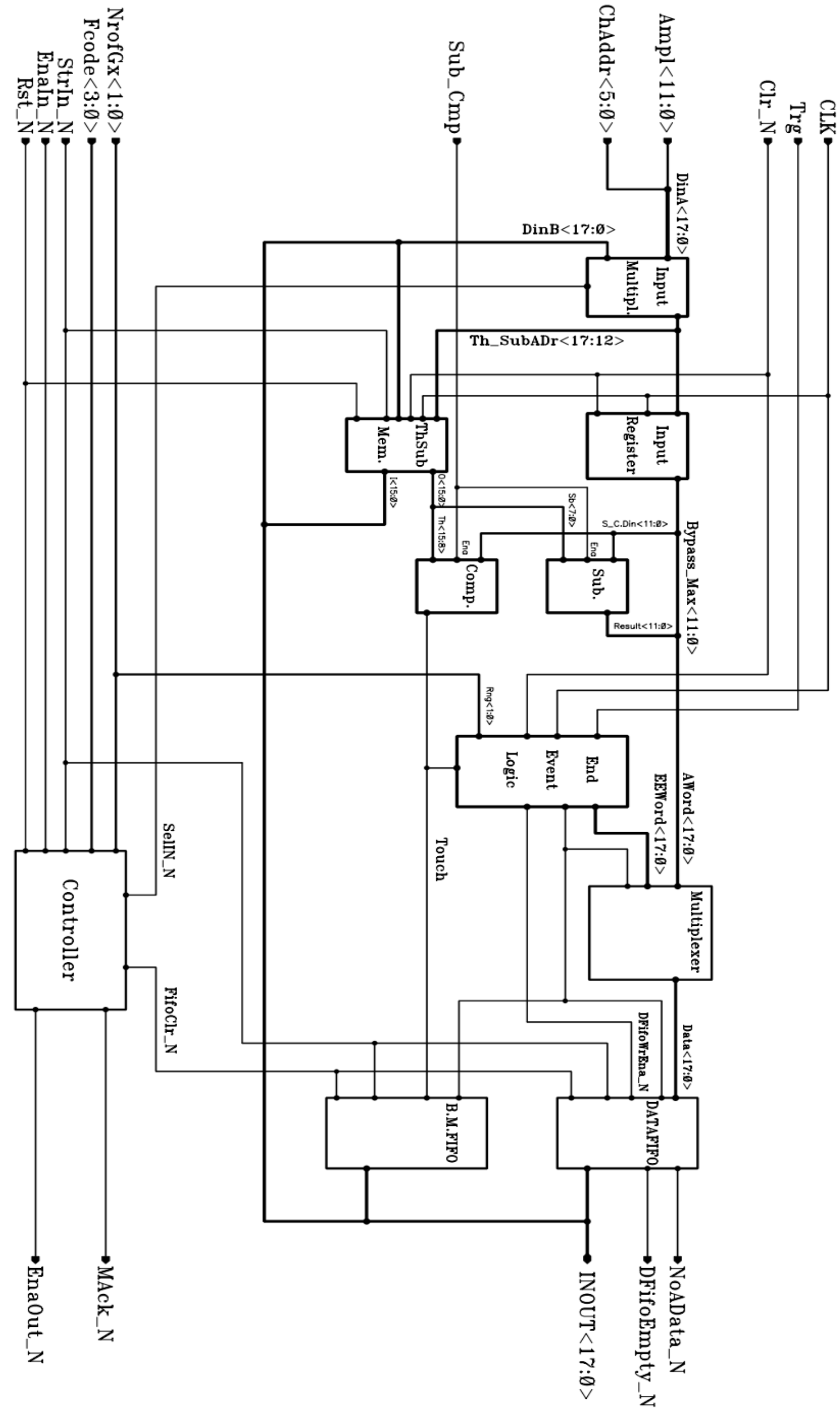


Figure 12: DILOGIC Block Diagram [4].

2.3 Present FEE-SIU Interface

In the present system, the FEEs implement the full-duplex Serial Interface Unit (SIU) interface shown in Figure 13. The main task of the SIU is to transmit event data, receive commands from the RORC (Read-out Receiver Card) - (Destination Interface Unit) DIU interface related to the control and test of the FEE. The DDL has two interfaces: the interface FEE-SIU and the interface RORC - DIU DDL.

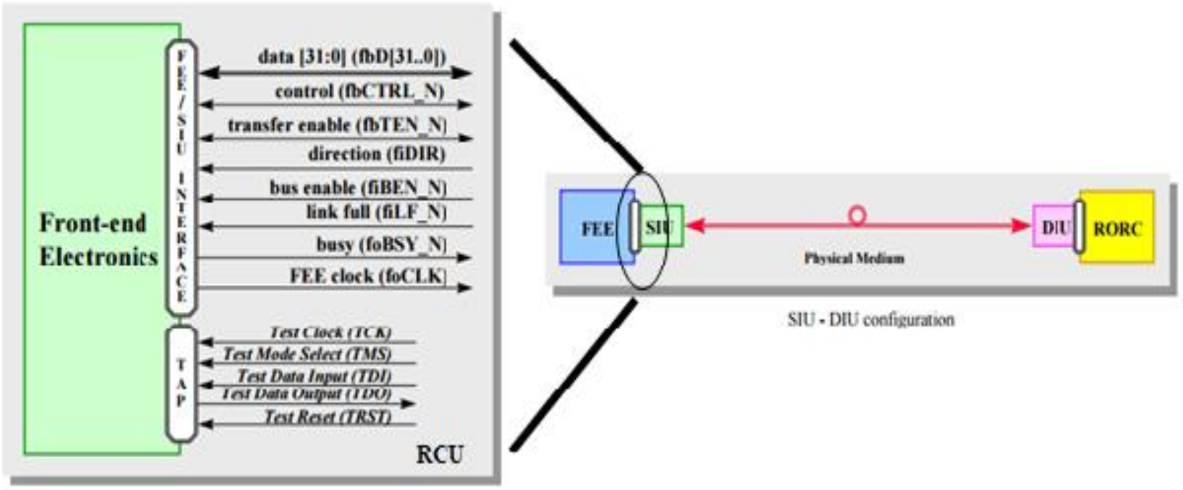


Figure 13: SIU-FEE interface signals and configuration [5].

2.3.1 RCU-SIU Interface

DDL signal interfaces are composed of four components: the interface identifier, the direction identifier, identifier name, and the polarity. All signals of the interface RO-SIU are synchronized with the clock signal FEE (foCLK). The fbD [31..0] is a 32-bit wide, three-level data bus. When the level of the fiDIR line is high, data on these bus lines is transferred from RO to the SIU on a rising edge of the foCLK clock, when fbTEN_N is set to low. The fbD [31..0] contains status words when fbCTRL_N is low and normal data words when high.

When the level of the fiDIR line is low, the data on these bus lines is transferred from the SIU to the RCU during a low-to-high transition of the foCLK when fbTEN_N is active. fbD [31..0] contains commands when fbCTRL_N is active, otherwise it contains normal data words. fbCTRL_N is a bidirectional tri-state management control line for the data bus with active level. The active level of this line indicates that the data word to be transmitted between RO and the SIU is a command word or a status word, depending on the direction of information transmission. It is a status word if the information is being transmitted from RO to the SIU, otherwise it is a command.

The fbTEN_N N is a bidirectional tri-state management control line for the data bus with active level. The active level of this line enables the transfer of data between the RCU and the SIU during the transition from low to high of the foCLK. The direction of information transfer depends on the level of the fiDIR line.

The fiDIR is an input for the management line for the data bus. The high level of this line indicates that the information is transferred from the RCU to the SIU on the fbD [31..0] lines, while the low level indicates the opposite direction of information transfer. When the level of this line is high, all bidirectional lines are driven by the RCU, otherwise they are driven by the SIU. fiBEN is an active low input line for the data bus. The inactive level of this line places the tri-state drivers of all bidirectional lines of the RO-SIU interface in the SIU and in the RCU in a high impedance state.

The fiLF_N is an active low-flow control line. The active level of this line indicates that the RCU should stop data block transmission and transmission of the status word FESTW (EOB = 1) to the SIU because the SIU and/or the DIU and/or the RORC are busy. After this line has become active, only one more data word or status word may be transmitted from the RO to the SIU.

The foBSY_N is an active low-flow control line. The active level of this line indicates that the RCU is not able to receive a data block from the DDL. After this line becomes active, only one more data word can be transmitted from the SIU to the RCU.

The foCLK is an output clock line. This free-running clock is generated by the RCU for synchronizing the information transmission between the RCU and the SIU. To improve the signal quality, the foCLK signal is terminated by a resistive load on the SIU board.

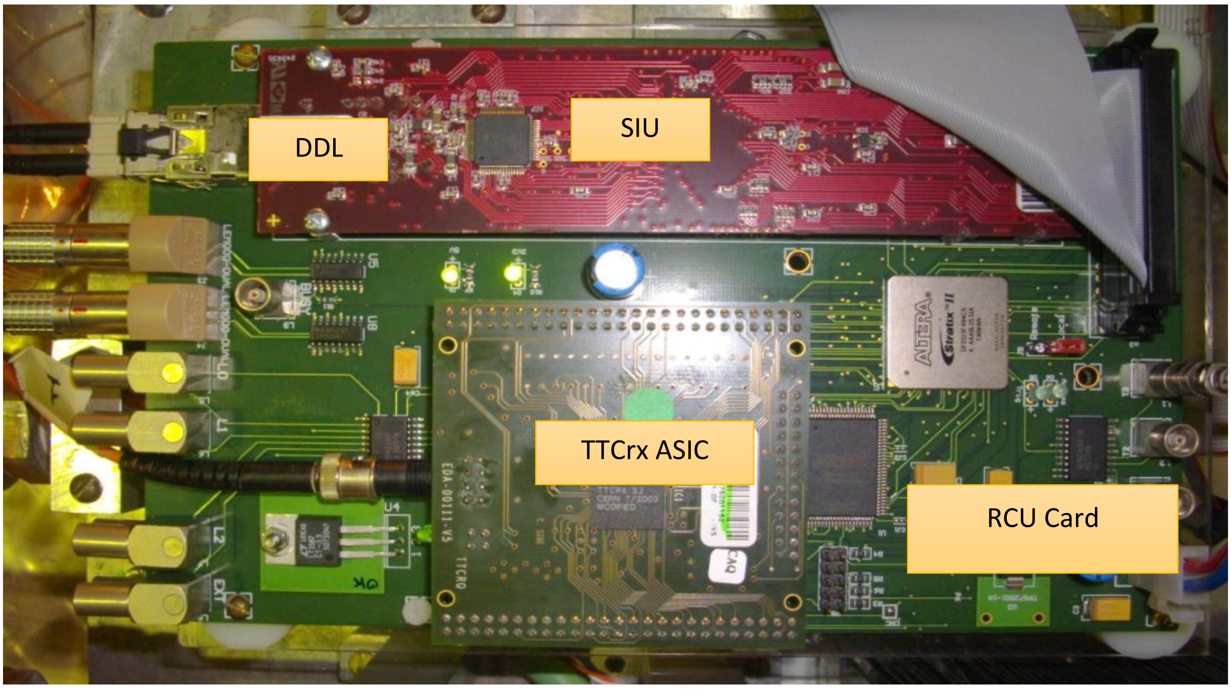


Figure 14: Illustration of RCU, FEE Readout and DDL link.

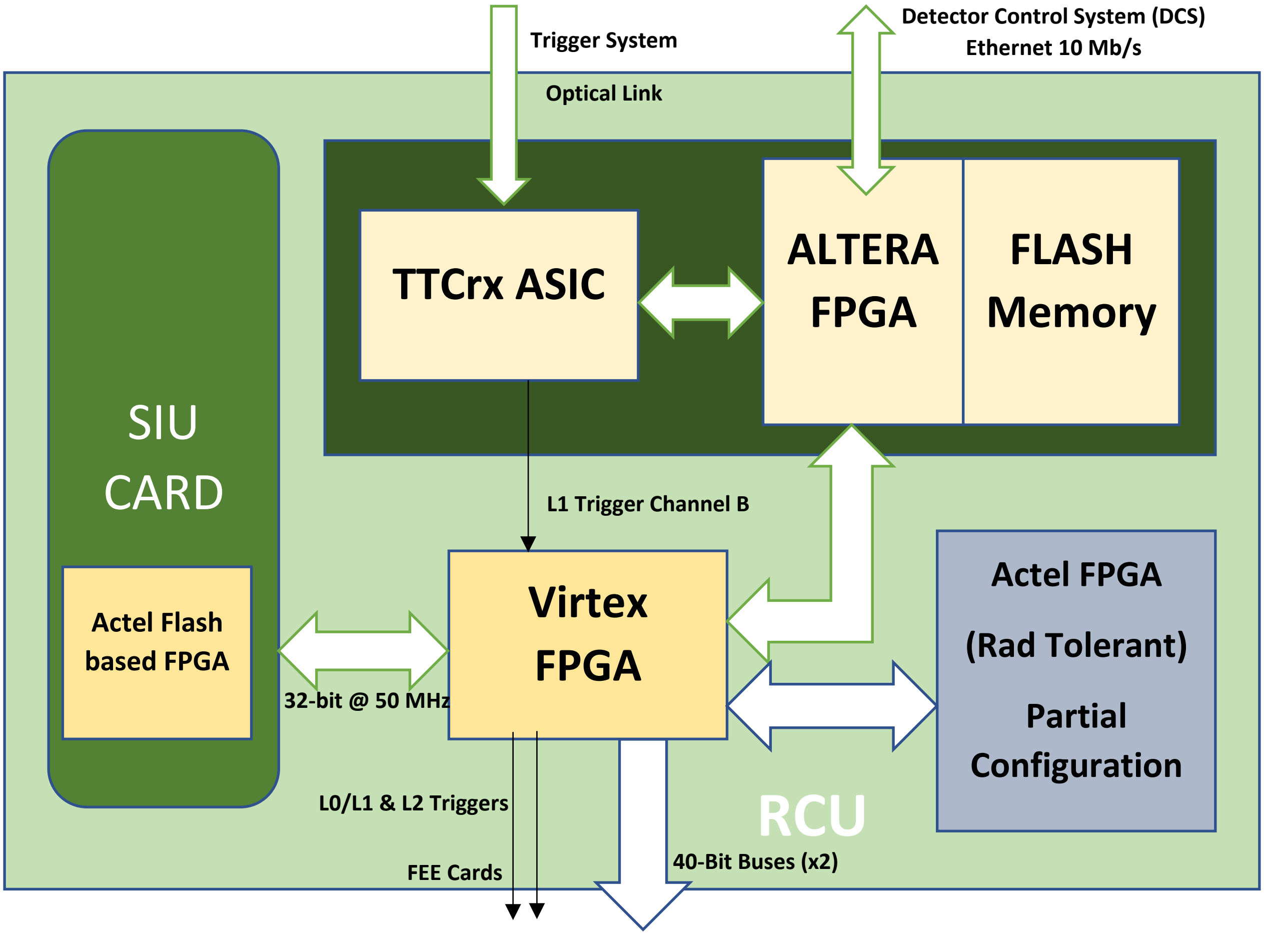


Figure 15: Block diagram for RCU card [6].

The RCU is an electronic card on the motherboard that controls the reading of data from FEE. It consists of various modules such as the Detector Control System (DCS) and the SIU, explained in the previous section, which are connected to the FEE cards via two 40-bit digital buses. The DCS's job is to configure and control the FEE and runs a Linux operating system. Communication with the DCS is via Ethernet and connects the local Trigger, Timing and Control (TTC) board to the FEE boards via a chip called the TTC receiver (TTCrx) [15]. The TTCrx system is connected to both the Altera FPGA and the DCS card.

The L0-L1-L2 trigger sequence is controlled by the RCU firmware and decoded into the L1 and L2 trigger messages from TTCRx. A Common Data Header (CDH) is constructed and prepended to each data event. The CDH contains information on bunch crossing, orbit number, region of interest, trigger classes, and other information. The detector electronics RO appends the CDH to all data sent via the transmitter of the ALICE DDLs. The CDH also contains information required to identify and tag the data in the downstream systems (DAQ, HLT and Offline). CDH is mandatory and should be generated for all data blocks sent via the DDL. Any invalid data blocks not associated with a specific event may be removed. Each CDH should have a unique identification number within the same run. All events related to the same software or physics trigger must be sent through the DDL and should have a CDH associated with them. The L0 trigger is sent as an LVDS signal with a latency of 1.2 µs. The TTC sends the L1 signal over channel A, while channel B is used to send the trigger and system data associated with L1. The L2 trigger is sent as a message to the TTC system after a delay. In the electronics of HMPID RO, the three trigger signals are delayed as shown in Figure 16, the L1 is received after 6.7 µs from L0 (latency), L2 after 108.3 µs from L0.

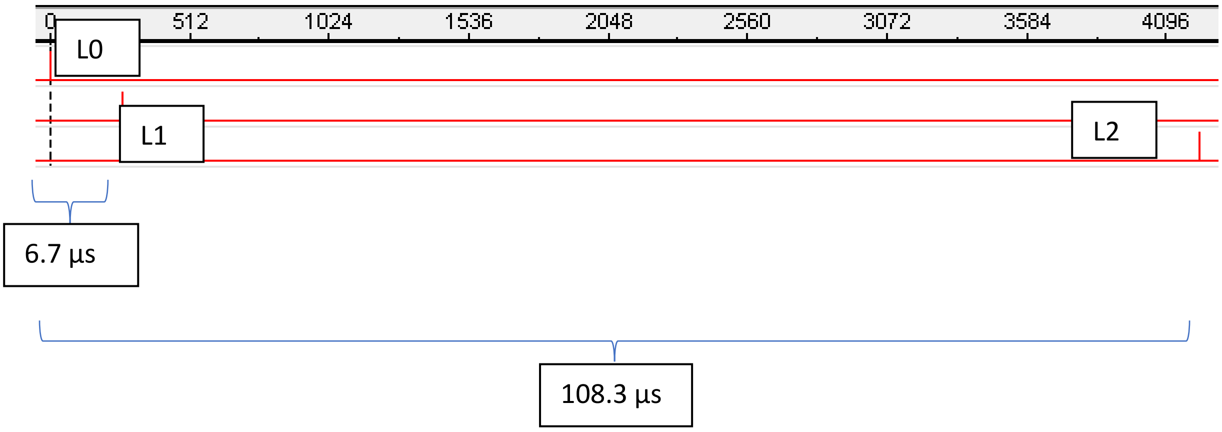


Figure 16: L0-L1-L2 trigger sequence timing.

The received triggers are distributed by CTP to the various ALICE detectors including PHOS, which are hierarchically divided into three levels: Level 0 (L0), Level 1 (L1a), and Level 2 (L2). The fastest trigger is L0 with about 1.2 µs from the time of event interaction, followed by L1a with a latency of 6.5 µs. The latency of the L2 trigger is about 88 µs to avoid data congestion before and after the collision. The central trigger processor transmits the calibration, software, and physics hardware triggers, which are distributed to various detectors such as the PHOS spectrometer. Physical triggers are generated using information from the PHOS detector FEE cards. The TTCrx system on the DCS board contains various interfaces to the Altera FPGA and is used to transmit the trigger information to FEE via optical fibre. The RCU firmware performs two different tasks related to data readout and functional control, which are further divided into a readout node and a control node. The task of the control node is to monitor various parameters such as the temperature and voltage levels of the FEC. The readout sequence is initialised by the triggers that put the TTCrx module into readout mode. A Level 2 Accept (L2a) trigger is used to acquire and store all channel event data in the data buffer. The data is framed with eight header words and a single trailer word as specified in the ALICE DDL Common Data Header (CDH) format.

Off-the-shelf SRAM-based flash memories are used, and radiation could cause functional or latch-up errors [16]. Active Partial Reconfiguration is a feature of the Xilinx device [17] that can be used to correct single-upset events (SUE) in FPGA memory. The readout of the data is controlled. In case of an SEU in the DCS FPGA memory, a restart is performed to reload the configuration memory and clear the error.

The pedestal values in the CPV are measured by a pedestal run of approximately 5000 events without using a physical trigger at a trigger frequency of 100 Hz, assuming negligible cosmic ray contamination. The FEE is reset with the CPVreset TTC command issued through the DAQ computer terminal. Zero suppression must be turned off in both segments of each module. This is done with the command CPVZSoff. A local trigger unit (LTU) is then used to run an emulation programme to initialise the TTC module and start data acquisition. The emulation process is stopped when about 5000 trigger events are reached. Since CPV consists of 16 column controllers, 16 pedestal files (1 per CC) are generated containing pedestals and over-pedestal thresholds for each channel in an 18-bit hex number. The first 9 bits of the 18-bit hex number contain the pedestal value, which means that the pedestal value must not exceed 29 or 512 ADC counts. The remaining 9 bits are used to define the threshold for exceeding the pedestal. This threshold is used to reduce the noise gain and usually corresponds to 3 Pedestal RMSs. Once the pedestal files are available, they are written or read to the CPV FEE memory by the DDL card using special commands contained in a script. All commands for writing and reading the DDL card are executed by a Linux script programme.

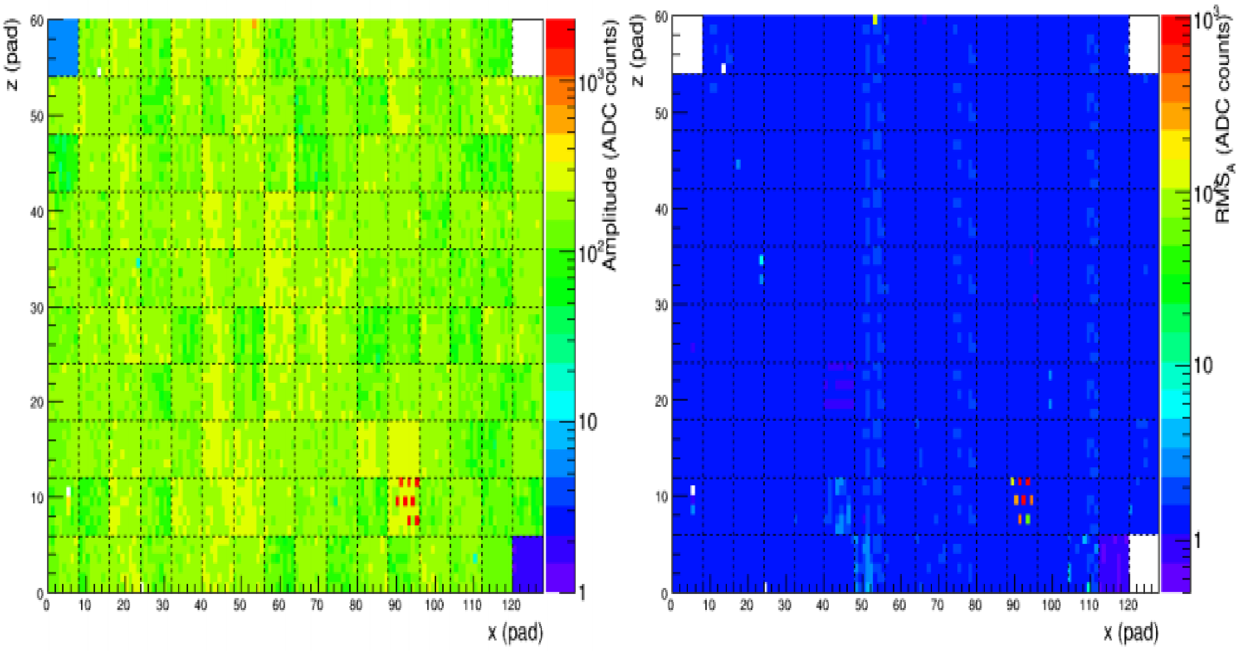


Figure 17: Illustration of Mean Amplitude and RMS amplitude Maps.

Once the pedestal files have been generated then it's time to write it into CPV FEE memory. Read/write procedures to FEE could be performed by DDL card with special commands show in Table 3.

Command	**Command Description**
write_<command 0x800	reset BUSY
write_command 0x4FF	reset TTCrx
Segment Selection	
write_command 0x5a4a8	select segment selection #1
write_command 0x6a4a8	select segment selection #2
Switch off zero suppression	
write_command 0x314a8	zero suppression off in Column #1
write_command 0x324a8	zero suppression off in Column #2
write_command 0x334a8	zero suppression off in Column #3
write_command 0x344a8	zero suppression off in Column #4
write_command 0x354a8	zero suppression off in Column #5
write_command 0x364a8	zero suppression off in Column #6
write_command 0x374a8	zero suppression off in Column #7
write_command 0x384a8	zero suppression off in Column #8
Switch on zero suppression	
write_command 0x315a8	zero suppression on in Column #1
write_command 0x325a8	zero suppression on in Column #2
write_command 0x335a8	zero suppression on in Column #3
write_command 0x345a8	zero suppression on in Column #4
write_command 0x355a8	zero suppression on in Column #5
write_command 0x365a8	zero suppression on in Column #6
write_command 0x375a8	zero suppression on in Column #7
write_command 0x385a8	zero suppression on in Column #8
Block Write	
write_block 0x61408	write block to Column #1
write_block 0x62408	write block to Column #2
write_block 0x63408	write block to Column #3
write_block 0x64408	write block to Column #4
write_block 0x65408	write block to Column #5
write_block 0x66408	write block to Column #6
write_block 0x67408	write block to Column #7
write_block 0x68408	write block to Column #8

Block Read	
read_and_print 0x61608	read block from Column #1
read_and_print 0x62608	read block from Column #2
read_and_print 0x63608	read block from Column #3
read_and_print 0x64608	read block from Column #4
read_and_print 0x65608	read block from Column #5
read_and_print 0x66608	read block from Column #6
read_and_print 0x67608	read block from Column #7
read_and_print 0x68608	read block from Column #8
Trigger configuration	
write_command 0x0	enable L2
write_command 0x20C8C	LTU L1 delay 0x103 (to be adjusted with cable length)
write_command 0x1800	L0 delay

Table 3. Configuration of CPV Front-end electronics using various special commands.

2.3.2 Front-End Commands

The various front-end commands (FECMDs) listed in Table 4 are part of the RORC and RCU message protocol used to monitor and control the FEEs.

D31	D30 D12	D11 D8	D7	D6	D5	D4	D3	D2	D1	D0	FECMD
	PARAMETER FIELD	ID FIELD	CODE FIELD				DESTINATION FIELD				
							R	RO	SIU	DIU	
x	is not used	ID	0	0	0	1	0	1	0	0	RDYRX
x	is not used	ID	1	0	1	1	0	1	0	0	EOBTR
x	address	ID	1	1	0	1	0	1	0	0	STBWR
x	address	ID	0	1	0	1	0	1	0	0	STBRD
x	address	ID	1	1	0	0	0	1	0	0	FECTRL
x	address	ID	0	1	0	0	0	1	0	0	FESTRD

Table 4. Front-end commands

The two mandatory standard FECMDs that must be implemented and used by the RCU of each sub-detector are as follows:

- Ready to Receive (RDYRX),

- End of Block Transfer (EOBTR).

A transaction to transfer event data may be opened with the RDYRX command. The RORC may send this command to the RCU via the DDL only when it is ready to receive event data. When the RCU receives this command, it transfers an event data block to the SIU after receiving an

L2 acceptance signal from the trigger system. Open block transfers (e.g., event data transfer, user-defined block write/read) are closed by the RCU when it receives an EOBTR command.

Implementation and application of the following four user-defined FECMDs is optional:

- Start of Block Write (STBWR),
- Start of Block Read (STBRD),
- Front-End Control (FECTRL),
- Front-End Status Read-out (FESTRD).

A block write transaction can be opened with the STBWR command. This transaction downloads a block of data from the RORC to the RCU. A block read transaction can be initiated with the STBRD command. This transaction reads a block of data from the RORC to the RCU. The RCU can be controlled using FECTRL instructions. Each address can represent an independent control function. FESTRD commands can be used to read status words from the RCU. Each address can represent an independent function for reading out status words.

There are two different types of the status words (FESTW):

1. The RORC read-out of a FESTW (EOB = 0) status word is done by sending a FESTRD command. The parameter field is reserved for RCU specific status information, while the error bit can be used for the indication of any predefined error states inside the RCU.
2. A FESTW (EOB = 1) status word is automatically generated by the RCU, when the transfer of an event or data block from the RCU to the SIU is terminated. The error bit shall be set to ‘1’ if any errors have been detected inside the RCU during the data block transfer.

2.4 The ALICE Detector Data Link

The ALICE Detector Data Link was developed to meet all data transmission requirements between detectors and the ALICE data acquisition system [6][18][19]. The first version of the protocol, called DDL1, is a 2.125 Gb/s full duplex multipurpose fibre link. It allows bidirectional transmission of data between the FEE and the data acquisition system. During data acquisition, information flows from the FEE to the DAQ, while each time a new run is started, it is possible to send configuration control messages to the detector via the opposite communication channel. The DDL1 consists of three main components, shown in Figure 18 below.

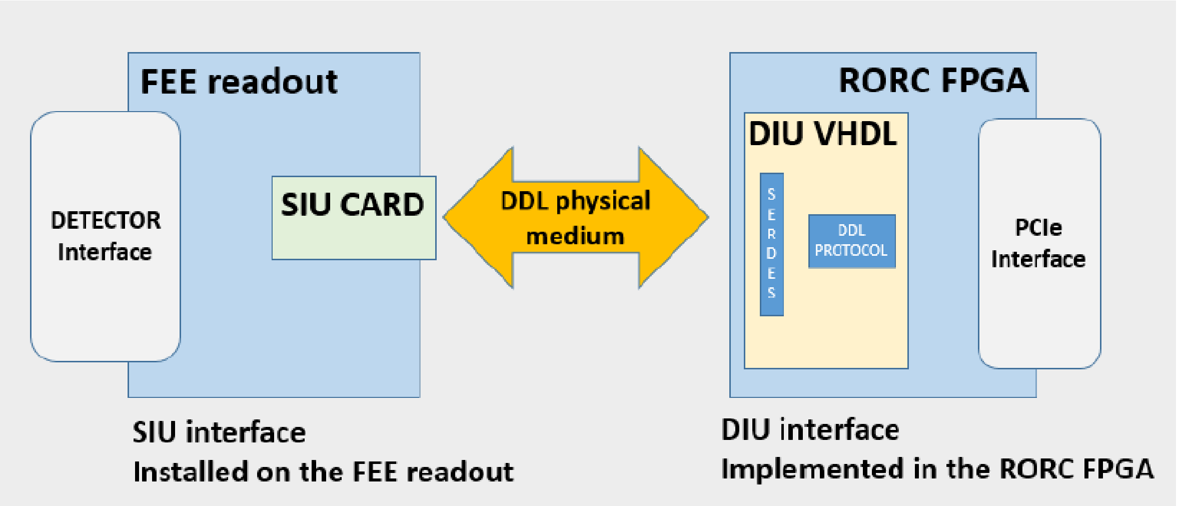


Figure 18: DDL1 components in data taking configuration between FEE and DAQ [5].

The physical medium is a multimode optical fibre. The detectors are connected to the DAQ farm via optical fibres of varying lengths up to 200 m. The DDL1 is a point-to-point protocol, i.e., each SIU is directly connected to a channel of the RORC via the optical fibre.

2.4.1 DDL Firmware

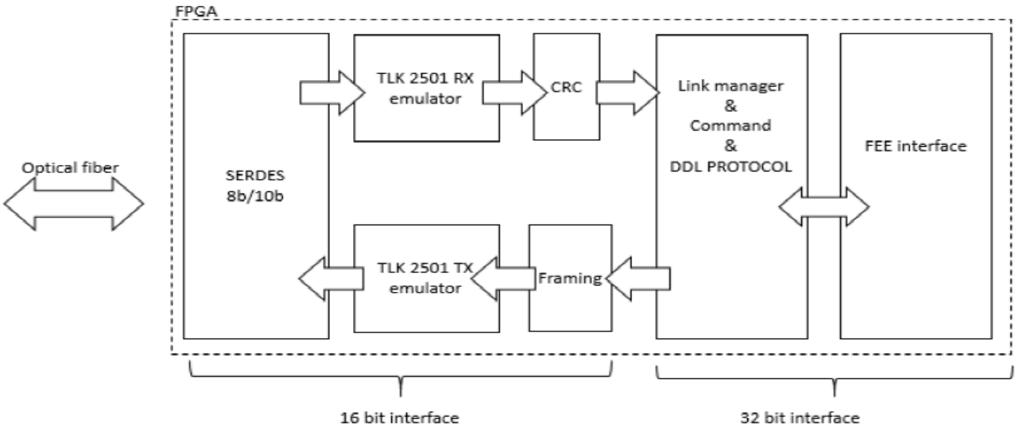


Figure 19: Main DDL firmware components and their communication paths [7].

The DDL provides a common data transfer protocol for detectors connected to the ALICE DAQ system. It encapsulates the data collected by FEE in a DDL frame and serializes it over the optical fiber. Its main components are:

SERDES: The Serializer - Deserializer transmits the serialized data over high-speed differential pairs, where it then recovers and aligns the data at the receiving end.

TLK2501 RX/TX State Machines: The TLK2501 transceiver chip from Texas Instruments was used in the old version of the RORC cards. This chip uses state machines to restore the connection and align the data. In the latest version of RORC, the Texas Instruments transceiver has been replaced by the SERDES available in the FPGA.

CRC: a Cyclic Redundancy Check (CRC) algorithm used to check data to detect possible errors during data transmission.

Framing: SERDES uses a 16-bit data bus interface, so the 32-bit DDL word is split into two 16-bit words.

DDL protocol and command interface: it encapsulates the data coming from the detector via DDL data blocks. When commands are sent from the RORC to the FEE, the control signals are incremented so that the FEE can respond to incoming commands or configuration data.

2.4.2 DDL2 Protocol

ALICE forecasts to increase the luminosity of the accelerator by 2018. Because of higher data throughput generated by sub-detectors, the DDL protocol has been maintained to achieve a faster speed of up to 6 Gb/s [7]. Therefore because of DDL1 bandwidth limitation of 2.125 Gbit/s, DDL2 which is at least twice faster, 4.25 Gb/s will replace DDL1. For DDL2 [8]the SIU card has been replaced by a VHDL core implemented in FPGA firmware of the readout electronics.

The main task of the FEE is to read the needed number of channels and transmit data to one DDL transmitter. The DDL1 bandwidth currently used in CPV and HMPID is 2.125 Gbit/s. A Typical event size from one CPV module is 1.3 to 5 Kbyte in Pb-Pb collisions. Further to achieve an event readout rate 50 kHz in Pb-Pb collisions, with an event size 5 Kbyte, the data throughput will be 2 Gbit/s which is already at the limit of DDL1 bandwidth. To further reduce the sub-event size by a factor of two, one can replace DDL1 by DDL2 which runs at a bandwidth of 4.2 Gbit/s.

2.5 DAQ Interface

The RCU can be remotely controlled via the RORC by sending user-defined FECTRL commands. The FECTRL commands are transmitted from the RORC to the RCU via the DDL. All FECTRL commands are automatically acknowledged by the SIU by sending a CTSTW back to the DIU. The CTSTW indicates whether an error occurred during the transmission of a FECTRL command. The FECTRL command will not be transmitted from the SIU to the RCU if a transmission error has occurred. The CTSTW is transmitted from the DIU to the RORC to terminate the transaction and report errors. Figure 20 illustrates the front-end control transaction. RORC can read FESTW from the RCU by sending a user-defined FESTRD command via the DDL. Upon receiving a FESTRD command, the RO transmits a FESTW to the SIU and then it is transmitted to RORC via the DDL. At the end of the transaction, the SIU acknowledges the FESTRD command with a CTSTW. The CTSTW indicates whether an error occurred during the transmission of a FESTRD command. The FESTRD command will not be transmitted from the SIU to the RCU if a transmission error has occurred. The CTSTW is transmitted from the DIU to the RORC to terminate the transaction and report errors. Figure 21 shows the front-end status word readout transaction.

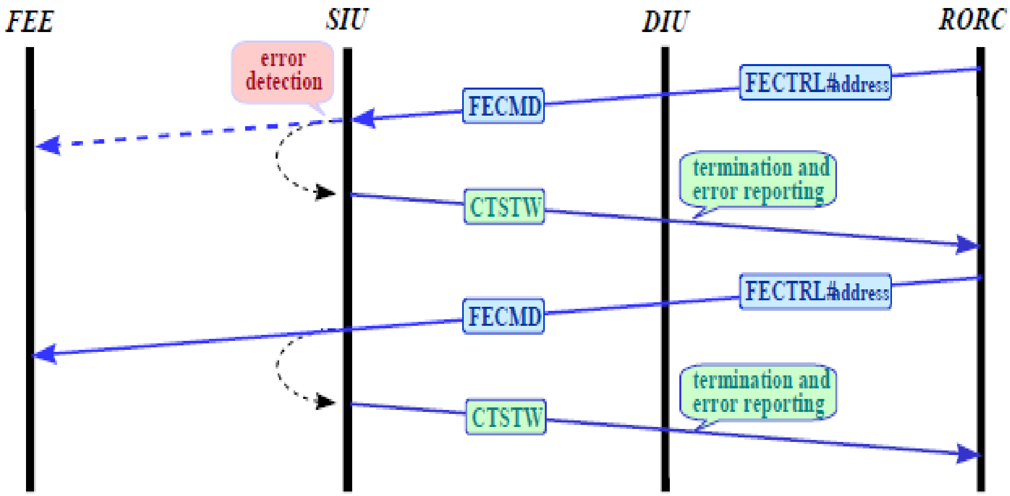


Figure 20: The front-end control transaction [9].

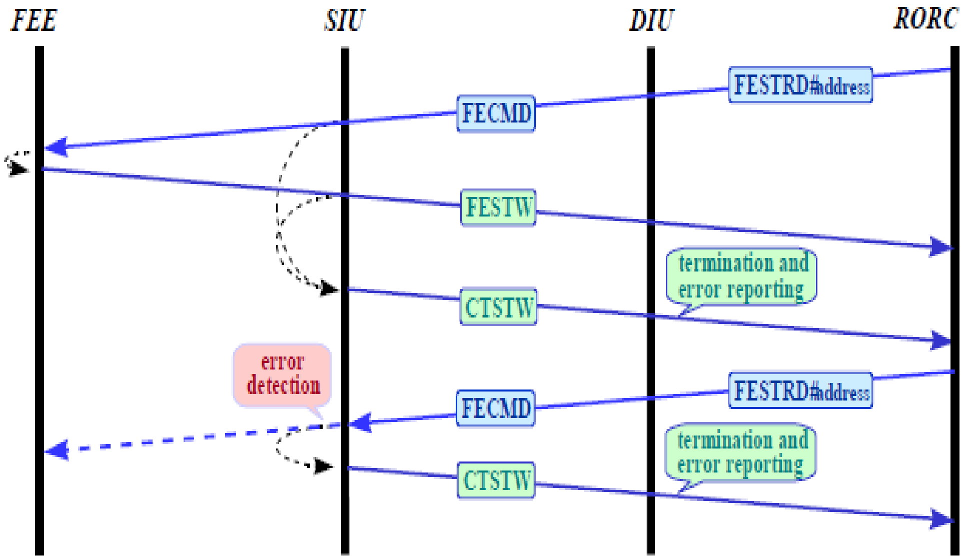


Figure 21: The front-end status word read-out transaction [5].

Event data can be transmitted from the RCU to the RORC via the forward channel. The RORC first sends an RDYRX command to the RCU to initiate the transmission of event data. The transmission of this command is acknowledged by the SIU. After each trigger system readout command, blocks of event data are transmitted from the RCU to the RORC, possibly with flow control. The end of a data block is indicated by the RCU transmitting a FESTW to the SIU. When a FESTW is received, a DTSTW is generated by the SIU and transmitted to the RORC. Data blocks may be continuously transmitted from the RCU to the RORC until the data transmission is terminated by the RORC sending an EOBTR command to the RCU. The transmission of this command is acknowledged by the SIU sending back a CTSTW to the DIU. This status word is transmitted from the DIU to the RORC to terminate the transaction and report errors. Figure 22 illustrates the event data transmission transaction.

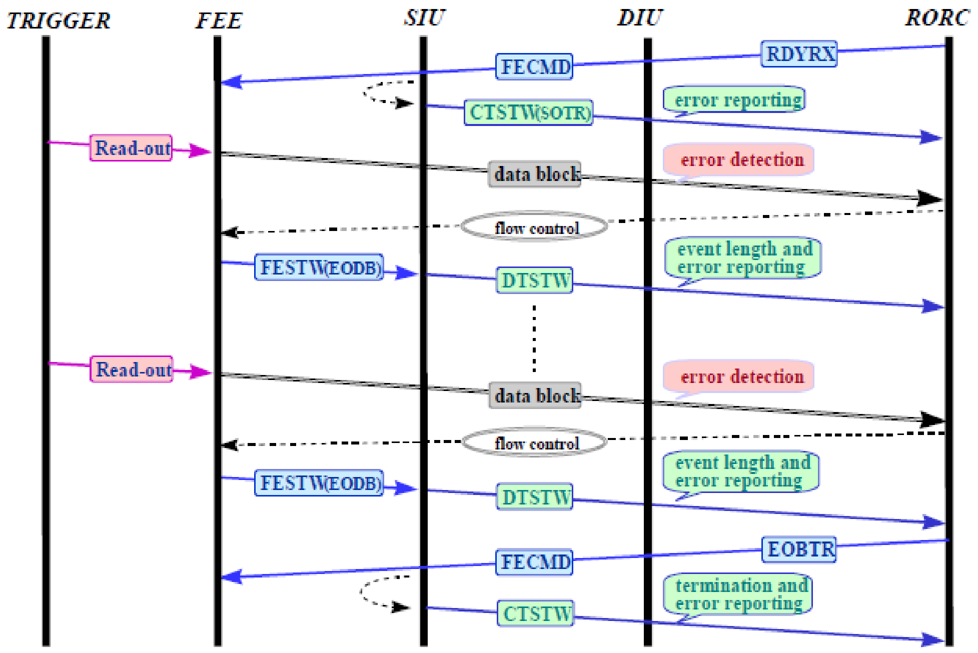


Figure 22: The event data transmission transaction.

Detector parameter data and event data can be downloaded from the RORC to the RCU memories via the reverse channel. First, an STBRW command is sent from the RORC to the RCU to select the target memory bank for the write operation. The transmission of this command is acknowledged by the SIU. A data block is transferred from the RORC to the selected memory bank of the RCU, possibly with flow control, followed by the DTSTW generated by the RORC. The end of a data block is indicated by the RORC sending an EOBTR command to the RCU. Upon receipt of this command, the target memory bank in the RCU is first deselected and the command transmission is acknowledged by the SIU by sending a CTSTW back to the DIU. This status word is transmitted from the DIU to the RORC to terminate the transaction and report errors. Figure 23 shows the transaction for downloading data blocks. Data blocks can be read back from the RCU's memories by the RORC using the reverse channel. First, an STBRD command is sent from the RORC to the RCU to select the target memory bank for the read transaction. The transmission of this command is acknowledged by the SIU. A data block is transferred from the selected memory bank of the RCU to the RORC, possibly with flow control. The end of a data block is indicated by the RCU transmitting a FESTW (with EODB = 1) to the SIU. When this FESTW is received, a DTSTW is generated by the SIU and transmitted to the RORC. The readback operation is completed by the RORC sending an EOBTR command to the RO. Upon receipt of this command, the target memory bank is first deselected in the RO and then the command transmission is acknowledged by the SIU, which returns a CTSTW to the DIU. This status word is transmitted from the DIU to the RORC to terminate the transaction and report errors. Figure 24 shows the data block read transaction.

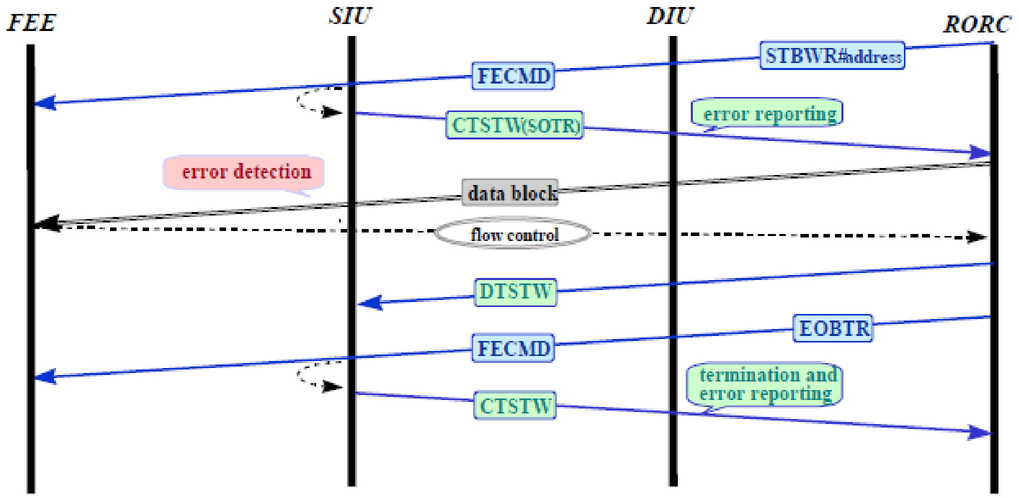


Figure 23: The data block downloading transaction.

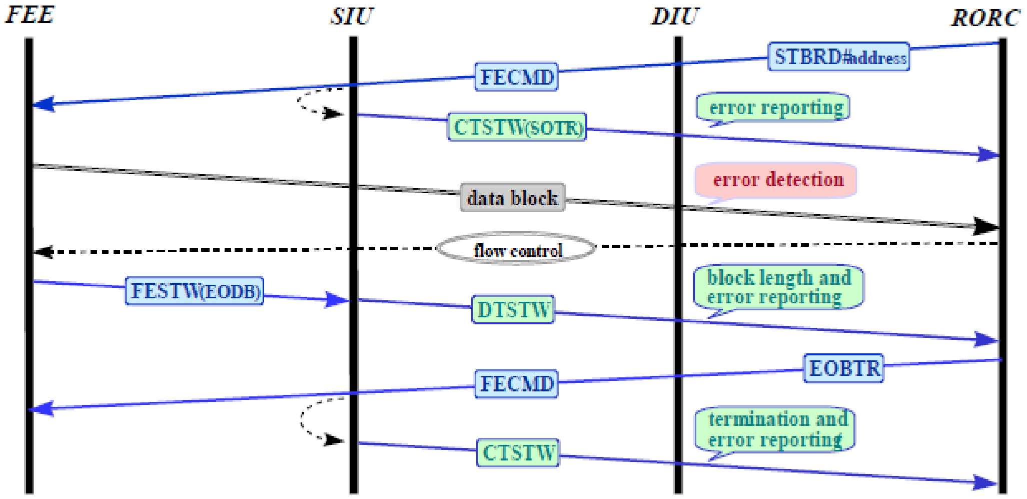


Figure 24: The data block read transaction.

2.5.1 SIU-RCB protocol

The following tables describe the various protocol commands used for SIU and RCB transactions. The focus is on the detector-defined fields in the 32-bit word, for bits from 30 to 12. The DES field (fbD [30...28]) contains information about the destination of the data: RCB, segment, column.

<table>
<tr><td></td><td colspan="11"><b>RCB ← SIU</b></td></tr>
<tr><td><b>CMD</b></td><td colspan="11"><b>fbD[31..0]</b></td></tr>
<tr><td rowspan="3"><b>FECTRL
FESTRD
STBWR
STBRD</b></td><td rowspan="2"><b>E</b></td><td rowspan="2"><b>DES</b></td><td rowspan="2"><b>NUM</b></td><td rowspan="2"><b>CTL</b></td><td colspan="4"><b>DATA</b></td><td rowspan="2"><b>ID</b></td><td rowspan="2"><b>CMD</b></td><td rowspan="2"><b>0x4</b></td></tr>
<tr><td><b>ZS</b></td><td><b>DILO CMD</b></td><td><b>CS</b></td><td><b>FS</b></td></tr>
<tr><td>[31]</td><td>[30..28]</td><td>[27..24]</td><td>[23..21]</td><td>[20]</td><td>[19..16]</td><td>[15..14]</td><td>[13..12]</td><td>[11..8]</td><td>[7..4]</td><td>[3..0]</td></tr>
<tr><td></td><td></td><td colspan="7"><b>Detector defined</b></td><td colspan="3"><b>SIU defined</b></td></tr>
<tr><td></td><td colspan="11"><b>RCB → SIU</b></td></tr>
<tr><td><b>CMD</b></td><td colspan="11"><b>fbD[31..0]</b></td></tr>
<tr><td rowspan="2"><b>FESTW</b>
(return of a FESTRD CMD)</td><td><b>E</b></td><td><b>0x4</b></td><td colspan="4"><b>000000000000</b></td><td colspan="2"><b>NUM</b></td><td><b>ID</b></td><td><b>CMD</b></td><td><b>0x4</b></td></tr>
<tr><td>[31]</td><td>[30..28]</td><td colspan="4">[27..16]</td><td colspan="2">[15..12]</td><td>[11..8]</td><td>[7..4]</td><td>[3..0]</td></tr>
<tr><td></td><td></td><td colspan="7"><b>Detector defined</b></td><td colspan="3"><b>SIU defined</b></td></tr>
</table>

Table 5. Command words in the SIU-RCB protocol [5].

DES	**DESTINATION**
000	RCB
1nn	SEGMENT (*nn* is the segment number)
011	COLUMN

Table 6. DES field [5].

The NUM field (fbD [27...24]) contains the column number if the target is a column; the hexadecimal number 'A' if the target is a segment. In addition, it contains commands for the RCB in conjunction with the CTL field (fbD [23...21]). The DATA field is for columns only. It consists of four subfields: ZS, DILO CMD, CS, FS. The ZS field indicates whether the null suppression is on or off. DILO CMD filed contains the command to configure the DILOGICs [2]. CS and FS fields indicate respectively the number of channels per DIL-5 card (in multiple of 16) and the multiplexing frequency. Both parameters are configurable.

DESTINATION	NUM	CTL	DECODING
COLUMN	0nnn		*nnn* is the column number
SEGMENT	1010		segment ID
RCB	0000	00x	RCB command: main state machine resetting
RCB	0000	01x	RCB command: TTCRx resetting
RCB	0000	10x	RCB command: busy signal clearing
RCB	0000	11x	RCB command: L1 latency setting
RCB	0001	00x	RCB command: RCB status word request
RCB	0001	10x	RCB command: L0 delay setting

Table 7. NUM and CTL fields [5].

ZS	DESCRIPTION
0	Zero suppression off
1	Zero suppression on

Table 8. ZS field [5].

DILO CMD	DESCRIPTION
0000	FE test mode
0001	Load "almost full preset"
0010..0111	No operation
1000	Pattern readout
1001	Pattern delete
1010	Analog readout
1011	Analog delete
1100	Reset FIFO pointers
1101	Reset daisy chain
1110	Configuration write
1111	Configuration read

Table 9. DILO CMD field [5].

BIT	NUMBER OF CHANNELS	MULTIPLEXING FREQUENCY
00	16	10 MHz
01	32	8 MHz
10	48	5 MHz
11	64	1.25 MHz

Table 10. CS and FS fields [5].

2.6 ALICE FEE Read-out Units

Other existing ALICE FEE readout architectures and integrated circuits have been considered in this review, such as the Scalable Readout Unit [20] and the Time Projection Chamber [21]. In this section, such electronic architectures for particle detectors are described in more detail. Also, a description of the Gigabit Transceiver Transmitter (GBTX) and Receiver (GBRX) ASIC (Application Specific Integrated Circuit) used for data transmission is given.

2.6.1 Gigabit Transceiver ASIC

Because of the high luminosity of the beams, the experiments require high data rate connections and electronic devices that can withstand high doses of radiation. The Application-Specified Integrated Circuit (ASIC) Gigabit Transceiver (GBTX) and Optical Transceiver Versatile Link Transceiver (VTRX) (VTTX) have been developed and extensively tested. The basic idea is to use the GBTX ASIC for data transmission, trigger distribution and clock forwarding to the front-end electronics. The GBT (GigaBit Transceiver) e-link is a 4.8 Gb/s bidirectional fibre optic link between the counting room and the physical experiments [20][21][22][23]. The GBTX interface contains an advanced digital interface for interfacing with FEE systems. The GBTX-SERDES interface can be operated in either parallel mode or e-link interface mode. The 40, 20 or 10 bidirectional serial links have bit rates of 80 Mb/s, 160 Mb/s and 320 Mb/s respectively. Each port transmits and receives serial data and the clock using a scalable low voltage signalling standard. The GBTX link provides the clock and can be configured over the link but reading its status must be done over a secondary link. Slow control links are implemented via an integrated Ethernet link of an advanced Reduced Instruction Set Machine (RISC) processor and a 1.25 Gbps optical fibre using Microsemi FPGA's Igloo2 internal serializer/deserializer [24].

The on-detector element will be a bidirectional optoelectronic module known as a Versatile Transceiver. It is based on a commercially available module type that has been minimally adapted to meet the mass, volume, power consumption, operating temperature, and radiation environment constraints of the Super LHC (SLHC) on-detector environment. The Versatile Link Module is intended to provide a multi-gigabit-per-second optical physical data link layer for readout and control of SLHC experiments. It is envisioned that the systems and components currently being evaluated and developed will support both bidirectional point-to-point (P2P) and point-to-multipoint (PON) architectures. The P2P implementation and its relationship to the GBT project [25] are shown in Figure 25 below.

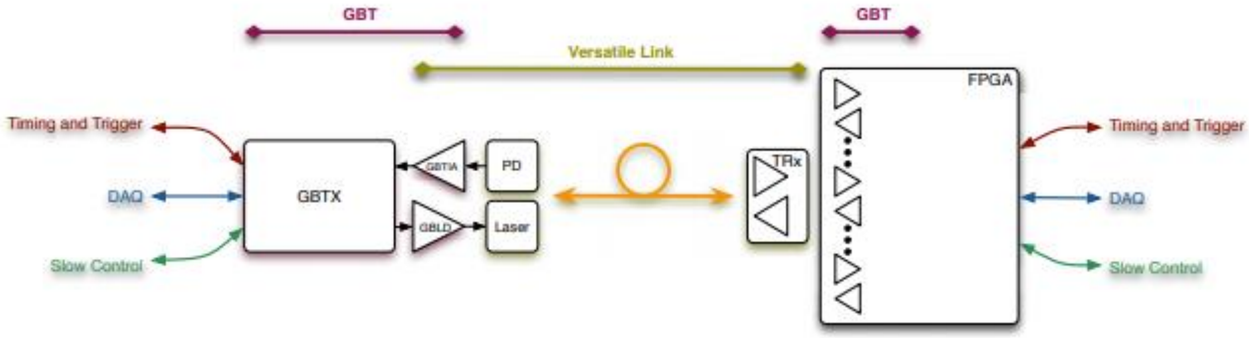


Figure 25: P2P radiation hard optical link for SLHC [10].

2.6.2 ALICE - Scalable Readout Unit

A Scalable Readout Unit (SRU) was developed as part of the CERN RD51 research and development collaboration [26]. SRU makes use of point-to-point data links between the FEE and the readout electronics. Each SRU has 40 point-to-point connections for the 40 FEE boards. Therefore, readout time is reduced by reading the boards from FEE simultaneously rather than sequentially over a bus. Event data and triggers, system clock, and commands are transmitted over a DTC (Data, Trigger, Clock, and Control) link between the SRU and each FEE card with a maximum bandwidth of 2 Gbps. This DTC link bandwidth is limited to 20 Mb/s due to the limited hardware capabilities of the legacy FEE FPGA. The SRU is connected to each FEE board via a custom DTC daughterboard, which consists mainly of an RJ45 port, an LVDS driver, and a power circuit designed specifically for the FEE board. The FPGA firmware uses a pipelining strategy in both the FEE and SRU firmware. Pipeline strategy allows better data readout throughput.

The transmission time over existing DDL links is denoted by t_{ddl} , while t_{dtc} denotes the time to transmit FEE data over the DTC link. As shown in Figure 26, the transmission time over the existing DDL links limits the readout rate for events larger than 3.6 KB. Therefore, if needed, this limitation can be mitigated by firmware updates in the SRU and using 10 Gb elink or using the DDL2 link speed of up to 5 Gbps. Assuming a data size of 1080 bytes per column, the SRU readout rate can vary from 18k events/s to 1.8 million events/s at a DTC link speed of 20 Mbps to 2 Gbps. At a DDL data transfer rate of 5.3125 Gbit/s, the event rate can vary from 70k events/s for 8 columns/DDL to 35k events for 16 columns/DDL.

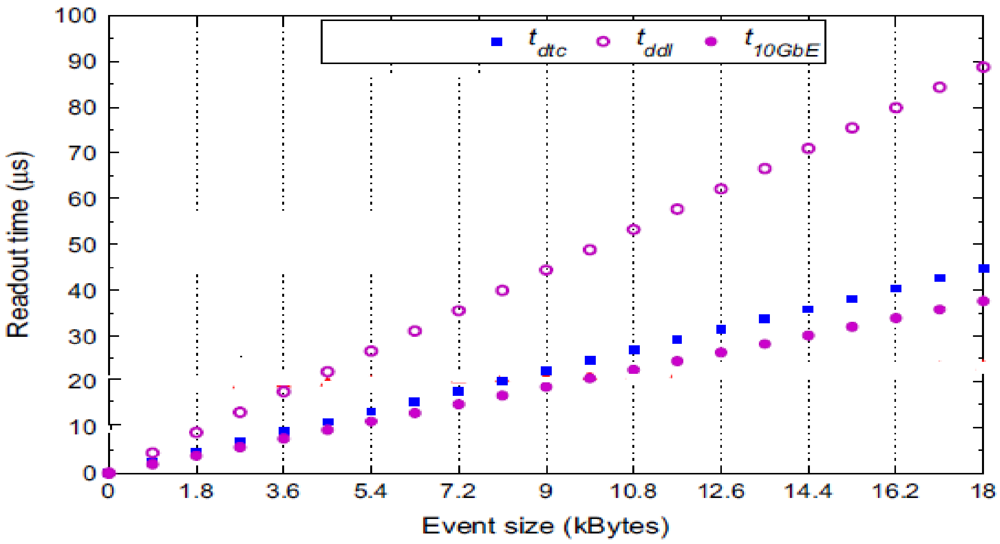


Figure 26: The correlation between the readout time and the event size for the various readout steps.

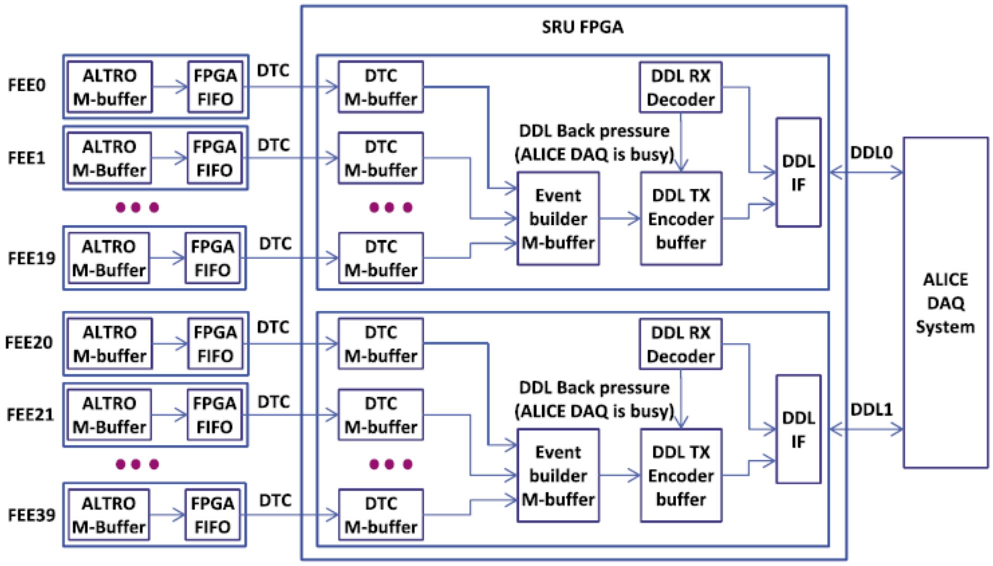


Figure 27: Block Diagram of SRU [11].

2.6.3 ALICE – Time Projection Chamber Particle Detector

The Time Projection Chamber particle detector (TPC) [27][28] is the main tracking detector of the ALICE experiment. It consists of a gas-filled cylinder with MWPC readout at both end plates with 557, 568 readout pads, and the signals from these pads are processed by 4,356 FEE cards. Data from the FEE cards is transmitted to 216 readout control units (RCUs), which are then connected to the rest of the ALICE DAQ system. The maximum readout electronic rate from TPC that has been measured is about 320 Hz. A 130-nm ASIC device called the SAMPA chip combines the functions of the earlier PASA and ALTRO ASIC [29] crisps into one. It also doubles the number of channels from 16 to 32, supports bipolar input signals, and provides the ability to operate in a continuous readout mode. Data acquired by the SAMPA is transmitted at 1.2 Gbit/s over four 320 Mbit/s serial links to a Gigabit Transceiver (GBTx) ASIC [30][31], which multiplexes input from various SAMPA crisps into a 4.8 Gbit/s versatile optical link component to a Common Readout Unit (CRU). A sketch of the system design for Run 3, planned for 2020, is shown in Figure 27. The SAMPA implements the GBT Slow Control ASIC (GBT-SCA), which has a dedicated data link on the GBT interface and acts as a board controller for configuring all SAMPA crisps on the same front-end board.

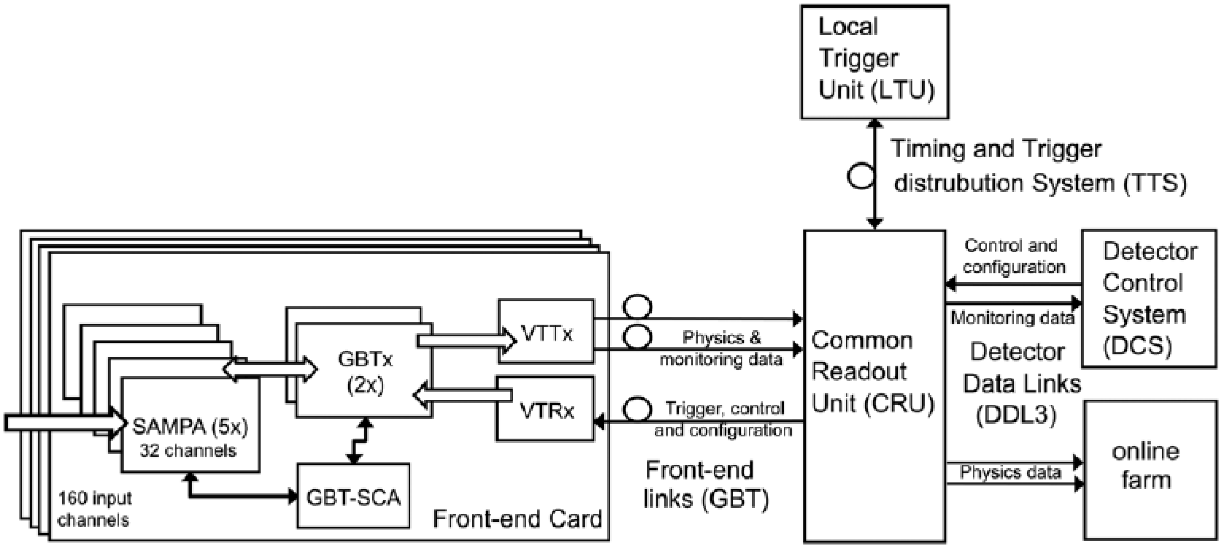

Local Trigger Unit (LTU)
Timing and Trigger distrubution System (TTS)
Control and configuration
Monitoring data
Detector Control System (DCS)
Common Readout Unit (CRU)
Detector Data Links (DDL3)
Physics data
online farm
Physics & monitoring data
Trigger, control and configuration
Front-end links (GBT)
VTTx
VTRx
GBTx (2x)
SAMPA (5x)
32 channels
GBT-SCA
160 input channels
Front-end Card

Figure 28: Block Diagram of TPC RO Architecture [12].

The FPGA-based CRU [32] acts as an interface between the front-end electronics, the online and offline computer system, and the central trigger processor. CRU manages and controls the readout, configuration and monitoring of the FEE cards.

2.6.4 ASICs versus FPGAs

The work presented in [33] compares and quantifies the difference between Field Programmable Gate Arrays (FPGAs) and Application Specific Integrated Circuits (ASICs) or fully custom design. The benefits associated with using FPGAs for design and implementation include reduced non-recurring development costs (NRE) and shorter time-to-market at the expense of increased silicon area, lower performance, and higher power consumption. This gap is more accurately measured using an empirical method that incorporates the results of many benchmark designs. Each benchmark design is implemented in an FPGA and standard cells. The silicon area, maximum operating frequency, and power consumption of the two implementations are compared to quantify the gap in area, delay, and power consumption between FPGAs and 90 nm ASICs. Both implementations compared, i.e., the standard ASIC STMicroelectronics and Altera Stratix II FPGA -based cells, use 90-nm CMOS technology. The benchmarks with less than 5% register difference written in similarly synthesised VHDL or Verilog are from different sources, while the designs were developed for projects at the University of Toronto. The gap given in [33] is the factor by which the area, delay or speed steps, and static and dynamic power dissipation are larger for FPGAs than for ASIC implementations. To enable programmable interconnects between these LUTs and flip-flops, FPGAs require an area 35 times larger than that of standard cell ASICs. This area difference can be reduced by 18 times by using heterogeneous hard blocks, such as hard memories. ASIC density can be compromised by adding whitespace and larger buffers to maintain speed and signal integrity. FPGAs are already designed in such a way that this area overhead is no longer an issue. Hierarchical floor planning techniques, where the design is divided into smaller blocks that are placed and routed individually, may become necessary for ASICs. Although the white space and buffer factors are important, the work presented in [33] focuses on comparing the core area gap, for both FPGA and ASIC devices, as it can have a significant impact on the cost difference between FPGAs and ASICs. The core area gap can be reduced by considering the package cost. As reported in [33] for circuits using pure logic, the average FPGA circuit is 3.4 times slower than the ASIC implementation, while for circuits using the hard DSP multiplier accumulator blocks, the average circuit in the FPGA was 3.5 to 4.8 times slower than in an ASIC; thus, the use of the hard block slowed down the design. The additional routing that may be required to accommodate the fixed positions of the hard multiplier, memory, or DSP blocks may negate the speed advantage that such hard blocks provide. For the difference in dynamic performance, FPGAs are found to consume on average 14 times more dynamic power than ASICs when the circuits contain only logic. Evaluating the gap in static performance is difficult due to temperature scaling, maturity, and subthreshold leakage for current technologies, which are highly process dependent, making it difficult to make a fair comparison. Despite this difficulty, it can be seen that the static power gap and area gap are reasonably correlated with a value of

0.81 for the typical and worst-case measurements. This is due to transistor width, which is generally proportional to static power consumption. Therefore, the area gap partially reflects the difference in total transistor width between an FPGA and an ASIC. The use of hard blocks reduces both the area gap and static power consumption. All modern ASIC designs require design-for-testability techniques to simplify post-manufacturing testing. Implementing scan chains simplifies testing [34] and requires replacing all sequential cells in the design with their scan-equivalent implementations.

	FPGA	**ASIC**
Size	Large (x35)	Small
Speed	Slower	Faster (x 3.0 to 4.6)
Dynamic Power	High (x14)	Low
Time to Market	Fast	Long
Flexibility	Fully Programmable	Not Possible
Cost	Low	High
Custom IP	No	Yes

Table 11. Comparison between ASIC and FPGA [13].

Table 11 compares the various gap factors between FPGAs and ASIC devices. As can be observed the drawback of using ASIC devices is concerned with costs and time to market. Such disadvantage can be contended using newer “structured ASIC” platforms [14]. The 90 nm Hardcopy II devices offer up to 90% cost reduction and up to 50% reduction in static and dynamic power consumption when compared to Stratix II FPGA prototypes.

2.6.5 ASICs for Particle Detectors

ASIC technology makes it possible to combine on one chip several functions that traditionally required separate components or modules. Due to limited space and specific thermal requirements for detector operation, signal processing electronics must consume very little power and occupy a compact space. In addition, the unavailability of commercial components that integrate a large number of analogue-to-digital (A/D) converters and the need to combine A/D converters channels in a custom data processor have resulted in the need to use custom ASIC devices. The large amounts of data generated by recent high-energy physics experiments require high-speed data links between the on-detector and off-detector systems.

The FEEs for ALICE EMCal, CPV, HMPID, and TOF particle detectors include the use of such ASIC integrated circuits. A non-exhaustive list includes NINO ASIC, which is used in ALICE TOF as a front-end amplifier/discriminator; ALTRO (ALICE TPC Read Out) chip, a mixed-signal integrated circuit designed as one of the building blocks of readout electronics for gas detectors; and 130 nm SAMPA ASIC, which is intended to replace ALTRO IC for upgrading the front-end readout electronics of ALICE TPC and Moun chambers. The POM (Processor Of Muon decays) ASIC was developed as a prototype digitizer for the straw tracker of the Mu2e experiment. The following section provides a description of the internal architecture of this, and other similar ASIC chips used in high energy physics (HEP) experiments. In addition, compact, radiation-tolerant optical data links with high bandwidth (in the range of 40 to 100 Gbps) are required. Finally, a comparison of the different characteristics of such ASICs is presented.

2.6.5.1 NINO ASIC Discriminator

NINO ASIC was fabricated in IBM 0.25µm CMOS technology [36]. NINO is an ultrafast preamplifier discriminator chip used in ALICE Time-Of-Flight detector. NINO IC has 8 channels, each channel equipped with an amplifier with a peak time of less than 1 ns and a discriminator with a minimum detection threshold of 10 fC and an output stage. Each channel consumes 27 mW, and the 8 channels fit into the 2x4 mm2 ASIC.

2.6.5.2 ALTRO ASIC

The ALTRO (ALICE TPC Read Out) ST Microelectronics HCMOS7 0.25µm technology chip integrates 16 channels, each of which consists of a 10-bit ADC, a pipeline data processor and a multi-acquisition data memory used for gas detector digitization and processing [22]. When the L1 trigger arrives, a predefined number of samples are processed and temporarily stored in a data memory. When the L2 trigger arrives, the last acquisition is recorded, otherwise it is overwritten by the next acquisition. Data is read at 60 MHz over a 40-bit wide bus, giving a total bandwidth of 300 Mbyte/s. The ADCs are placed in the centre of the chip to avoid noise caused by the memories. The main drawback of the ALTRO device is that it does not support continuous readout. Its design uses local data buffers and then transfers the data externally, but not simultaneously. The reason for this is that the sensitive ADC analogue input channels should not be flooded with noise.

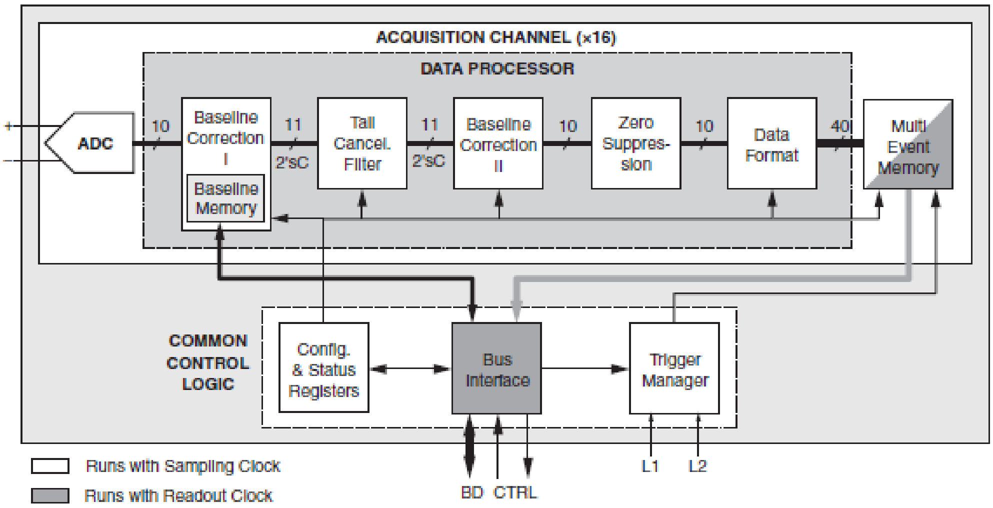


Figure 29: Block Diagram for ALTRO IC.

The area of the chip is 64 mm^2 and the power consumption is 320 mW when the 16 channels acquire at 10 MHz. The measurements show a resolution of more than 9.5 ENOB on all channels.

Parameter	**ALTRO specification**
Supply Voltage	2.5 V
Technology	ST 0.25 µm
Number of Channels	16
ADC resolution	10-bit
ADC maximum sampling frequency	25 MSps
Power consumption	20 mW
Die size	64 mm^2

Table 12. ALTRO ASIC specifications.

2.6.5.3 Processor Of Muon decays

POM is one of the first 65nm CMOS ASIC technology used for high energy physics [37]. The use of POM severely limited the chip area, requiring power supplies and regulators to be located outside the chip, which increased susceptibility to crosstalk. Key internal components of the

POM include a low-voltage differential signal (LVDS) interface, a near-end external pre-amplifier (NEXPA) input, a current-sensitive pre-amplifier (PRE), and a two-wire port. The far end current signal is transmitted to the far end preamplifier (FEPA), amplified, discriminated, and routed to an amplified discriminated far end output (FADO9) LVDS port. The output of FADO is routed to an input of the far end me-to-digital converter (FETDC) and used to stop the far end time-to-digital converter (TDC).

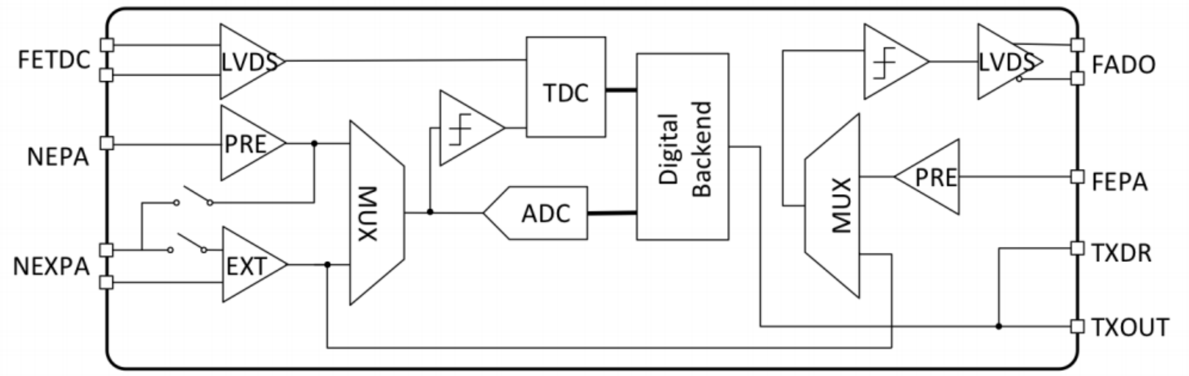


Figure 30: Top level diagram for two POM channels

The prototype POM could be useful for the development of future HEP detector readout electronics because it is low power, reliable, and radiation resistant. In addition, crosstalk in such a device is avoided by carefully isolating analogue and digital signals, as well as power, ground, and reference inputs in both the ASIC and its package.

Parameter	**POM specification**
Technology	65 nm
Number of Channels	4
ADC resolution, ENOB	7-bits minimum
ADC maximum sampling frequency	50MSps
Power consumption	50 mW/channel

Table 13. POM ASIC specifications.

POM was never used for Mu2e but it was then decided to implement the digitizer using commercial components, and no further development of the POM ASIC is foreseen.

2.6.5.4 Belle-II PXD

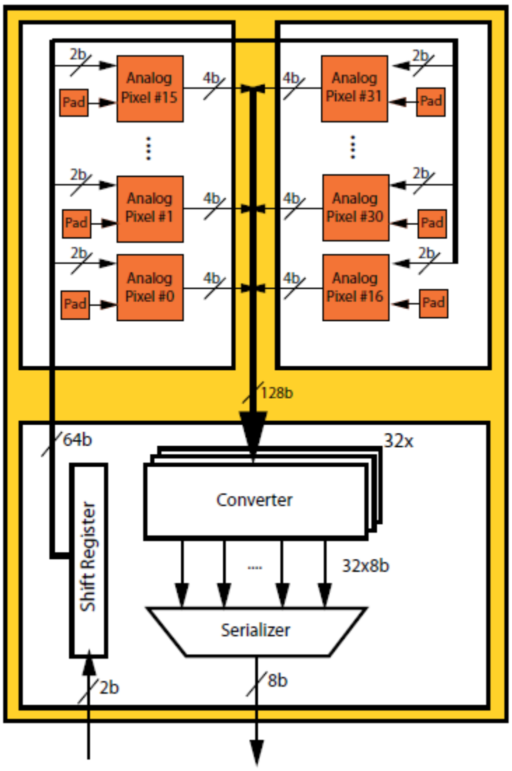


Figure 31: A simplified schematic diagram for one DCDB chip [15].

The silicon pixel vertex detector (PXD) is based on monolithic arrays of DEPFET (Depleted P-channel FET) transistors. It is located in the Belle- II physics experiment at the SuperKEKB particle accelerator in Japan [38]. Unlike the LHC, the world's highest energy proton accelerator, SuperKEKB/Belle II is designed to have the highest luminosity in the world. The DEPFET Current Digitizer ASIC (DCDB) chip is a very complex multichannel ASIC for the pixel detector of the Belle- II experiment, containing 256 8-bit channels with a sampling rate of 10 MHz, ADC, and a very low power dissipation of 4 mW per channel. The matrix elements are processed so that rows are read sequentially, and all columns are read in a cycle of less than 100 ns. As shown in Figure 31, a DCDB channel consists of eight identical blocks, each with 32 ADC channels and a single 8-bit wide output link. The ADC uses redundant sign cyclic conversion (RSD) and starts by comparing the input signal with two thresholds, one positive and one negative, and sets two new bits for each new conversion. In case the input signal is greater than the positive threshold, the pair of output code bits is set to 10, which means +1, and a reference current is subtracted. If the input signal is less than the negative threshold, the output code is set to 01 (-1) and the reference is added. If it is between the threshold values, the bits are set to 00 (0) and no arithmetic operation is performed. These two bits are used to extend the dynamic range of the ADC by dynamically compensating for pedestal fluctuations on the input signal. The translated conversion of all 256 ADC channels is serialized to eight 8-bit links, where each link needs to operate at a speed of 400 MHz, resulting in a data rate of 400 MB/s and hence a total rate of 3.2 GB/s for the entire chip.

2.6.5.4 IDE 3465

The IDE 3465 0.35μm CMOS ASIC was designed for reading silicon charged particle detectors [39]. IDE 3465 contains 20 charge-sensitive preamplifier (CSA) inputs, a total of 37 digital logic trigger outputs, and an analogue multiplexer output for pulse heights. The chip is optimised for reading and triggering charge from the p-side of silicon sensors. All 20 input channels are protected by diodes against overvoltage and electrostatic discharge (ESD). In addition, the chip is hardened by design and fabrication against SEU/SEL radiation to meet total ionisation dose requirements of 100 krad (TID). Guard rings have been used in the digital circuits to prevent one-time latching (SEL). Another programmable configuration register was developed to recover from a single event upset (SEU) through triple redundancy with self-correcting circuitry.

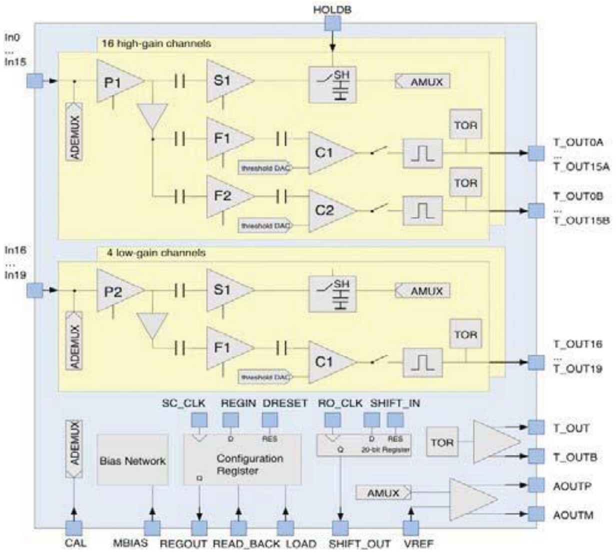


Figure 32: Block Diagram for IDE3465 Chip [16].

2.6.5.5 SAMPA

The SAMPA ASIC [40] consists of 32 identical channels consisting of a Charge Sensitive Amplifier (CSA), a pulse shaper, a ADC and a Digital Signal Processor (DSP) to digitise very small and fast 10 fC and 10 ns TPC pad signals.

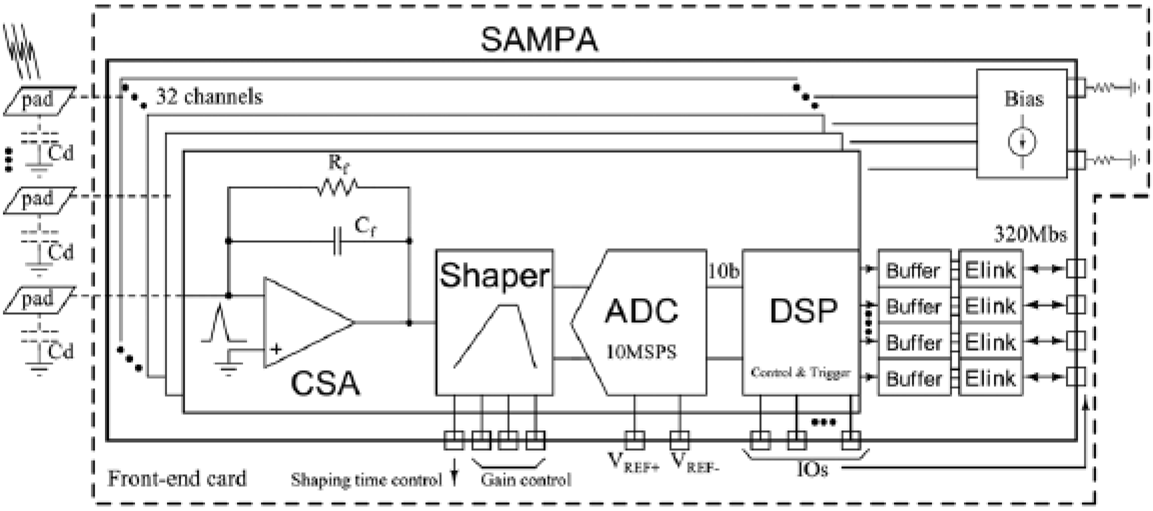


Figure 33: Block Diagram for SAMPA ASIC [17].

The CSA design is based on the previous PASA IC with several additional options to convert the incoming signal into a pulse with a long tail whose amplitude is proportional to the total charge. Additional features include selectable polarity, gain, and shaping time to match different detectors. SAMPA includes a 10-bit capacitive successive approximation processor (SAR) ADC, optimised to operate at 10 MSps, but optionally up to 20 MSps, to dramatically reduce power consumption compared to the ADC pipeline system used in the ALTRO chip. The digital signal processing chain (DSP) first performs baseline correction by subtraction and self-calibration, then activates the IIR cancelation filter to smooth the signal tail, and again performs a second baseline correction due to the crossover effect. Finally, it implements a zero-suppression method using a fixed threshold pulse detection scheme and then loads the compressed data into a FIFO memory buffer for later transfer to a Common Readout Unit (CRU) via a GBTx ASIC link. The DSP can operate in both triggered and continuous readout modes.

Ref	No of Channels	ENOB Resolution bits	Sampling Rate MHz	Readout bandwidth MB/sec	Power Consumption mW	Technology µm	Area mm^2	Supply Voltage (Volts)	Dimensions mm^2
[18]	16	9.8	10	300	320	0.25	64	2.5	7.70 × 8.35
[19]	8	/	10	/	216	0.25	8	2.5	2.0 x 4.0
[20]	4	6.8	50	/	126	0.065	/	/	/
[15]	256	8	10	3200	1024	0.18	48	1.8	3.2 x 5.0
[16]	20	/	/	0.0625	65	0.35	42.6	3.3	6 x7.1
[21] [17]	32	9.2	10	20	320	0.13	/	1.25	/

Table 14. Comparison between various ASIC devices used in HEP experiments.

A comparison of the various properties of the described ASIC devices used in particle detectors is shown in Table 14 above. The work presented in [38] contributes to the highest bandwidth due to the largest number of simultaneous readout channels, resulting in the best luminosity quantity of 2.1 × 1034 cm^{-2}s^{-1}, compared to the requirements of CERN ALICE of 6 × 1027 cm^{-2}s^{-1}. Thus, based on this evaluation, increasing the data transmission rate and parallel readout of silicon or MWPC pad channels should help improve the luminosity values. In current CPV or HMPID FEE systems, the channels are read out sequentially, resulting in a poor readout rate of no more than 10 kHz. In addition, signal conditioning, digital signal processing, and event data transmission are performed by three different devices, namely GASSIPLEX ASIC, the DIL-5 card, and a ALTERA FPGA, resulting in an approximate readout delay of 106µs. The CPV DIL-5 processor board is designed to process the data output from a 10-bit ADC so that the empty channels are eliminated, the baseline pedestals are subtracted, and the true digitized data is logically stored. Limitations that drastically affect CPV performance due to the Dilogic-5 card include the following:

- Maximum readout speed of 5 MHz -10 MHz,
- A daisy chain interface between DILOGIC chips on a 5-DIL card, leading to the sequential readout of all processors,
- Can handle the sequential readout of up-to 64 channels only,
- Old 700 nm technology leading to 60mW power consumption per DILOGIC chip and slow speed,
- Sequential processing of pedestals and noise levels,

- Only one external FPGA per segment used to sequentially manage all data transfer from each DILOGIC chip. Simultaneous data transfer not possible with the present architecture.

The above limitations contribute to the increased space requirements, power consumption, and cost of manufacturing and maintaining the readout electronics of the CPV detector. In addition, as described earlier, the use of an FPGA contributes to an increase in power dissipation, silicon area, and a decrease in speed, which led to the selection of an ASIC process technology of less than 180 nm for the implementation of a new custom FEE system that integrates all three functions to improve the performance of the CPV/HMPID system in terms of power dissipation, area, throughput, and speed.

2.7 Single Event Functional Interrupts (SEFI) in FPGAs

As specified [22] by the Joint Electron Device Engineering Council (JEDEC) JESD89A specification, SEFI definition of a SEFI is specified as:

"*A soft error that causes the component to reset, lock-up, or otherwise malfunction in a detectable way, but does not require power cycling of the device (off and back on) to restore operability, unlike single-event latch-up (SEL) or result in permanent damage as in single event burnout (SEB).*"

Additionally, as specified by JESD89A:

"Soft errors are non-destructive functional errors induced by energetic ion strikes. Soft errors are a subset of single event effects (SEE), and include SEUs, multiple-bit upsets (MBU), SEFI, single-event transients (SET) that, if latched, become SEU, and SEL where the formation of parasitic bipolar action in CMOS wells induce a low impedance path between power and ground, producing a high current condition (SEL can also cause latent and hard errors)."

Based on the above statements, SEFI is often associated with an upset of a bit in a register, and it is non-destructive. Design techniques such as Error Detection Cyclic Redundancy Check (EDCRC), and Triple Modular Redundancy Strategies (TMR) on data packets may greatly affect the SEFI ratio. The industry defines the SEFI ratio as the number of SEE divided by the number of functional interrupts. Therefore, having no functional interrupts under an unlimited number of SEE would be the target goal, but in practice this would be difficult to achieve. Based on this the system design requirements must specify an SEFI ratio for the specific application.

The injection of faults into a register allows a system designer to evaluate the overall system response to SEU.

Other terminology specified by JEDEC, and that can be used to estimate the failure in time rate for a specific device include the following:

- **Flux:** rate at which particles impinge upon a unit surface area, given in particles/cm^2/s,
- **Fluence:** total number of particles that impinge upon a unit surface area for a given time interval, given in particles/cm^2,
- **FIT**: Failure in time rate (in 1 billion hours):

  -1 FIT is one Failure in one billion hours of operation,
- **MTBF:** Mean Time Before Failure is the predicted elapsed time between Failures.

2.7.1 Estimation of Failure-In-Time for New CPV FEE

This section describes a fictitious example for estimating the FIT rate concerning an FPGA-based electronic system. FPGA part number details will be undisclosed. As specified by [23] the reference neutron flux altitude is 0 feet (sea-level) is 14 neutrons/cm^2 hours. Radiation conditions after LS2 are calculated: based on the ALICE public note ALICE-PUB-405 (https://alice-notes.web.cern.ch/node/405), still in preparation at the time of writing. Calculations are based on the expected EMCal fluence after LS2, 1.9 x 10^9 neutrons/cm^2. It is non–Ionizing Energy Loss (NIEL) in 1-MeV neutron equivalent hadron fluence. CPV is located at a distance 450 cm from the ALICE interaction point therefore its radiation conditions are the same as one for EMCal.

Assuming a 100% usage of the FPGA device,

Total FIT = 2490 errors are expected in 1 billion hours.

For EMCal the expected fluence after Long Shutdown 2 (LS2) is 1.9 x 10^{9}n/cm^2 [24],

Therefore, the number of errors after LS2 = 2490 x (1.9 x 10^9/ 14 x 10^9) = 33.

Considering the SEFI factor:

Number of errors after LS2 = 338 / 10 = 34

Taking into consideration 20% resource usage for the VHDL implementation of the SIU module in a selected FPGA device as shown in Figure 20 then,

Total FIT = 20% of 2490 = 498 errors are expected in 1 billion hours.

The Number of errors after LS2 = 498 x (1.9 x 10^9/ 14 x 10^9) = 68

Considering the SEFI factor:

Number of errors after LS2 = 68 / 10 = 7

So, with 24 FPGAs, that's 7*24 = 168 faults, which is ~1 SEU fault/week over three years or after LS2. Based on the results of these calculations, it can be shown that the FIT rate is minimal and therefore easily recoverable.

Logic Utilization (ALMs)	11%
Total registers	8554
Total block of memory bits	9%
Total HSSI RX PCSs	17%
Total HSSI TX PCs	17%
Total HSSI PMA TX Serializers	17%
Total PLLs	17%
Total Pins	4%

Table 15. An example of FPGA Resources for SIU Implementation.

2.8 Conclusion

This chapter of this dissertation reviews the various types of FEE readout architectures used in different types of particle detectors. It was found that each detector implements a custom ASIC or FPGA design to meet the system requirements. It was also found that ASIC devices provide better performance in terms of area, power dissipation, and speed. As described, the CPV detector FEE must be redesigned to achieve the required luminance of 6 x 10^{27} $cm^{-2}s^{-1}$. This luminance can be achieved either by developing a new electronic readout system architecture or by implementing a newly developed ASIC device that provides the capability to process many MWPC channels of the detector simultaneously.

In addition, parallel processing of channels is limited by the current CPV FEE. Therefore, a new FEE needs to be developed to achieve the event readout rate of 50 kHz required by the ALICE Collaboration. The following sections of this dissertation provide details of the chosen methodology, including architectural details of the newly developed FEE readout electronics, which has been installed, commissioned, and deployed since November 2020. Simulation and measurement results for both the developed ASIC device and the newly installed CPV readout system FEE are presented in the following chapters.

CHAPTER 3 - Implementation of the CPV Readout Electronic System

Scientific studies in experimental nuclear physics rely on electronics and are mostly used to detect radiation or particles. Although detectors seem to vary widely, the basic principles of readout apply to all. They consist of a signal current sensor whose output, after integration, gives a charge proportional to the amount of energy collected.

Each particle detector must have a tailored FEE system to detect specific beam types and particle characteristics, so the equipment must be modular and adaptable. In addition, the design criteria for a particle detector depend on the application, energy resolution, rate or timing requirements, and sensor positioning. Large-scale systems have additional requirements for power consumption, scalability, and ease of reliable data acquisition and monitoring while reducing maintenance costs. Today, high-resolution particle detector systems must improve their luminosity and resolving power, which can be achieved by using a large number of sensor channels. For large systems, power dissipation and size are critical, so systems are not necessarily designed for optimal noise, but for reasonable noise, and circuitry must be tailored to the specific requirements of the detector.

The prior CPV detector FEE limits the readout rate to 4 kHz, which is ten times lower than the targeted requirements and luminosities. This results in the need to implement a newly adapted FEE system architecture that meets a specific design criterion and constraints imposed by a team of physicists from the Russian Institute of High Energy Physics. Examples of such design constraints and limitations for the redesigned CPV electronics are related to the use of the current DIL-5 cards and GASSIPLEX chips. This is due to time, space and budget constraints, including reliability and acquired know-how related to these components. In addition, both ASIC devices were designed specifically for the CPV detector, whose purpose is to improve photon identification in the photon spectrometer PHOS. Thus, there are no similar components available on the market. The entire newly developed CPV electronics system must be completed before the Run 3 period, which is scheduled to begin in May 2022 and be extended to 2024. Therefore, commissioning and installation of the complete system should be completed by January 2022. This chapter provides a more detailed description of the new CPV FEE system implementation, focusing mainly on the hardware electronics and digital implementation.

3.1 Target Design Requirements

The ALICE upgrade strategy for Run 3 is based on the collection of more than 10 nb^{-1} Pb-Pb collisions at luminosities up to 6x10^{27} cm^{-2}s^{-1}, corresponding to a collision rate of 50 kHz, with each collision sent remotely to the online and offline computer system (O^{2}), following either a self-triggered or a continuous readout procedure. Other additional requirements include the absence of the L2 trigger, meaning many detectors will transmit data to the new DAQ system in continuous mode, using only L0 and L1 trigger signals. The CPV performs no action when an L1 trigger arrives, while L0 is necessary for the operation of track and hold of the photon-generated charge-sense amplifiers located in the front-end electronic readout cards filled with GASSIPLEX devices. Also, during Run 3, the detector does not wait for an L2 trigger to begin event transmission to the DAQ. When the data is available in the COLUMN memories of the GASSIPLEX chips, the transmission can be opened, and event data frames can begin to be sent immediately. The time required to get the data from the memories of COLUMN, since the start of the "track and hold" signal for the GASSIPLEX devices, is estimated to be 5 µs. Therefore, the CPV detector occupancy time will be about 110 µs shorter compared to Run 2.

Since the online system can process both continuous readouts from the detectors, the event construction is based on the compilation of data recorded in a common frame for all detectors. To avoid spreading the data over two consecutive time frames, a frame duration of at least 100 ms is targeted, compared to the TPC drift time associated with the start of the ionization event to the end of the MWPC. Such time frame limits are communicated by FEE through the transmission of non-physical heartbeat triggers. This allows the data frames to be divided into segments for event formation. Such heartbeat triggers are scheduled by the CTP with the highest possible priority and at a fixed interval. The FEE readout detector hardware then implements copies of the orbit, trigger, and bunch counters, which are also simultaneously aligned with the LHC counters, and which are then fully transmitted only during a heartbeat event. The counters are rebalanced in the event of a discrepancy between these counters and the detected error is then transmitted to the O^{2} system where they are used for data segmentation, troubleshooting, and other recovery issues. Therefore, the readout electronics of the CPV detector must also be modified to handle this combination of heartbeat and physics triggers. A data block must be preceded by a header for each trigger, heartbeat, or physics.

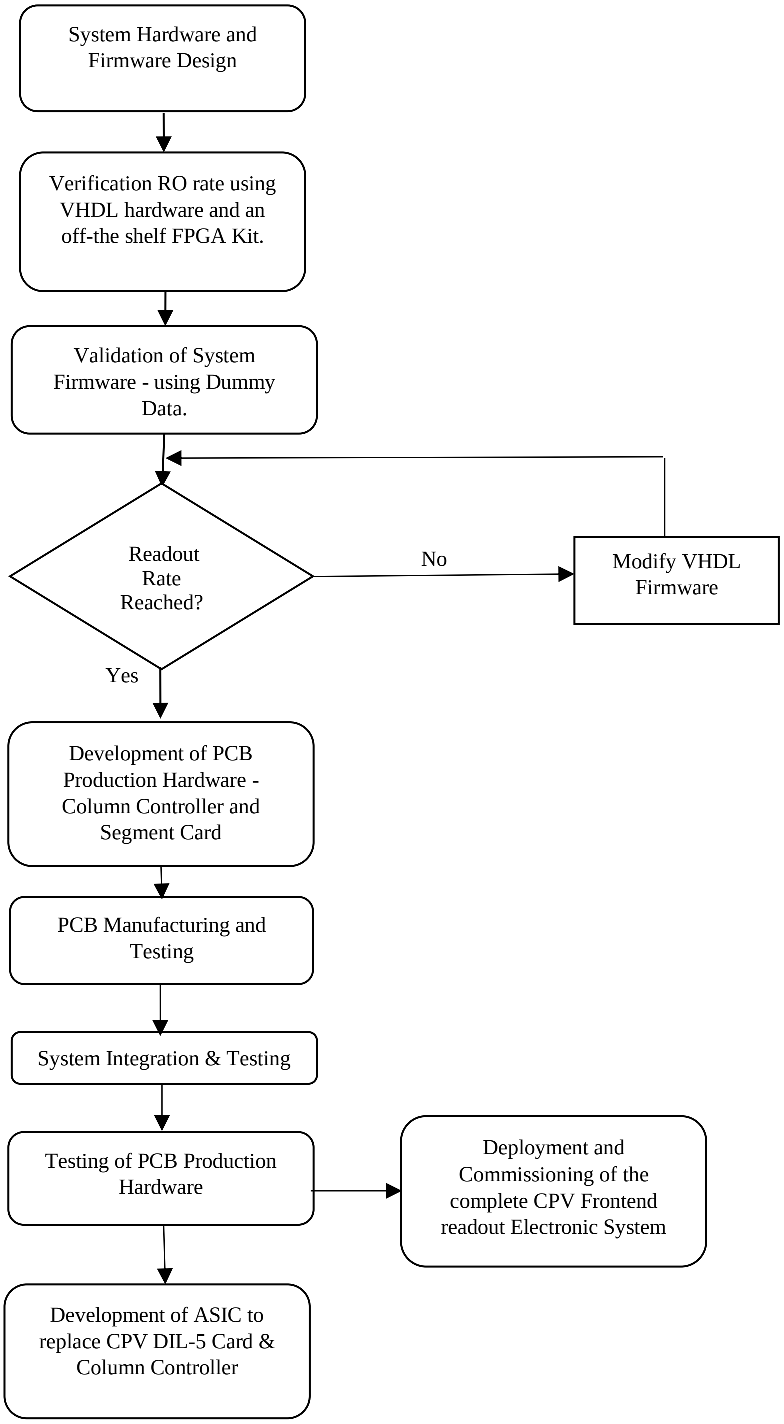

System Hardware and Firmware Design
Verification RO rate using VHDL hardware and an off-the shelf FPGA Kit.
Validation of System Firmware - using Dummy Data.
Readout Rate Reached?
No
Modify VHDL Firmware
Yes
Development of PCB Production Hardware - Column Controller and Segment Card
PCB Manufacturing and Testing
System Integration & Testing
Testing of PCB Production Hardware
Deployment and Commissioning of the complete CPV Frontend readout Electronic System
Development of ASIC to replace CPV DIL-5 Card & Column Controller

Figure 34: Various Stages planned for the Implementation of the new CPV Frontend Readout Electronic System.

Considering the elimination of L2 latency (about 110 µs) and the current time required to transfer event data to the COLUMN FIFO memories, which is less than ~22 µs, the new design of the FEE readout architecture should meet the target event readout rate of 50 kHz, considering such timing constraints. Figure 34 illustrates the adopted system-level design procedure and phases for the new CPV FEE readout system.

3.2.1 FEE Prototype Schema Verification using Off-the-Shelf FPGA Kits.

The upgrade of the new CPV front-end readout electronics detector system requires the need and development of a new scheme consisting of different partitions. As requested by the ALICE collaboration, this new scheme should use both the GASSIPLEX and DILOGIC chips, which includes the use of a certain number of segments and column controller FPGA boards. Therefore, before actual production and fabrication of all CPV electronic circuit boards began, an initial custom electronic prototype was developed for evaluation purposes, as shown in Figure 35 below. This first prototype consists of the following electronic boards:

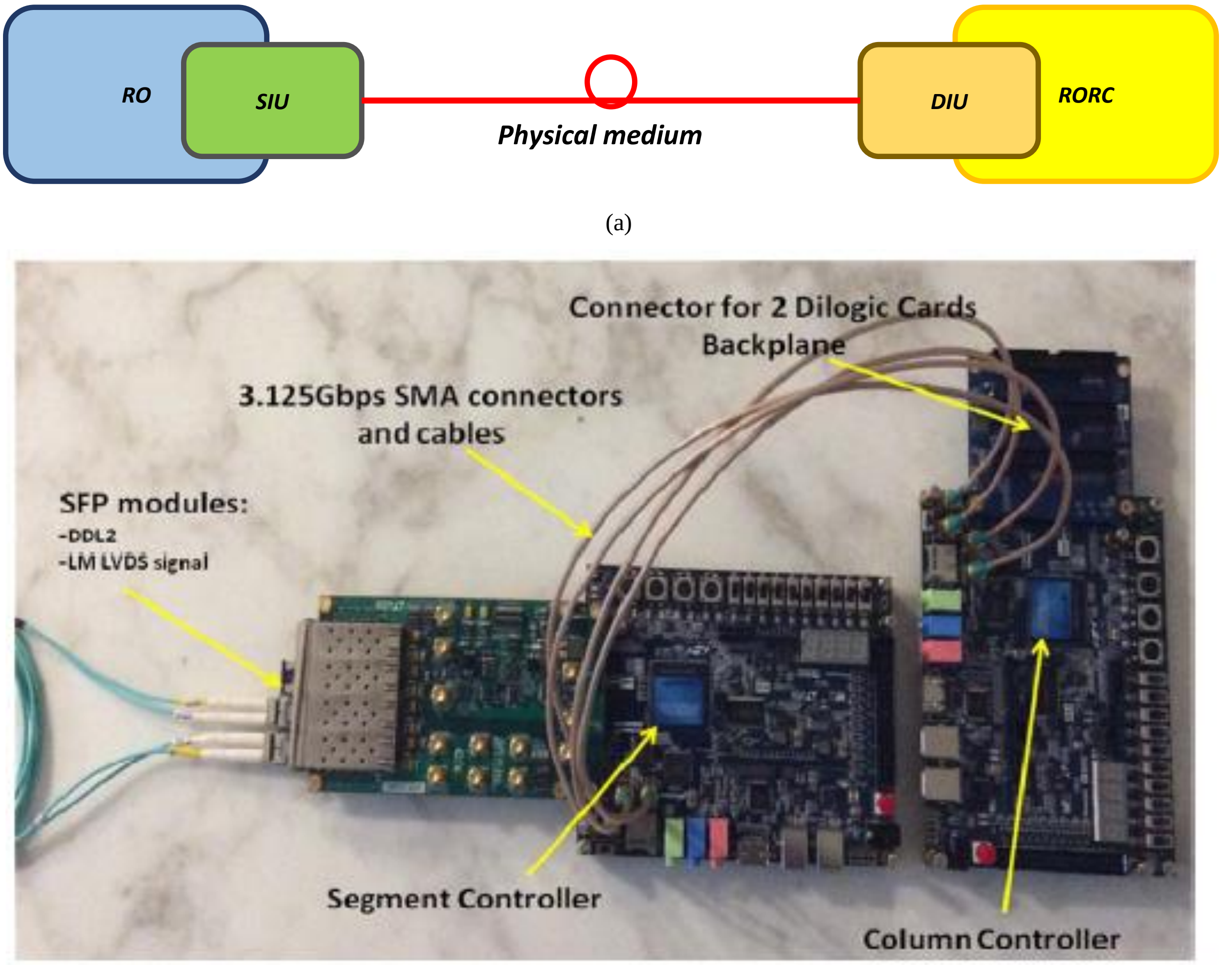
- One segment card,
- One column controller card,
- One SFP (Small Form-Factor Pluggable) module, and,
- One backplane for interfacing two DILOGIC cards.

Further, the use of TLK2501 RX/TX chips, for the implementation of DDL1 protocol found in the previous CPV and HMPID electronics, was replaced with the use of new VHDL modules emulating the operation of TLK2501 RX/TX integrated circuits, so to implement the SIU and DDL2 communication protocol on a Cyclone V GX FPGA device.

A standard detector readout chain consisting of DDL, RORC, or CRU, is connected to DAQ via the ALICE subdetectors. The purpose of the DDL2 is to transfer event data from RO FEE to the RORC. In addition, the DDL2 can be used to remotely control and test the readout electronics from DAQ by using specific commands and status words. Unlike status commands, data blocks can be transmitted in both directions. The DDL2 consists of three main components:

- Source Interface Unit (SIU), connected to the FEE RO.
- Destination Interface Unit (DIU), connected to RORC.
- Physical medium (two fibre optic cables).

Event fragments are transmitted from the experimental pit to the computer room via DDL and then stored in the RORC input buffer. The Local Data Collector (LDC) then processes such event fragments from the RORC input buffer and assembles them as sub-events. The central event building switch then assembles the received sub-events. The SIU and DIU are connected to the FEE readout and the RORC, respectively. Both the SIU and DIU communicate only through the physical DDL2 connection and require no other cable. The DDL2 control software controls these various subsystems from the LDC.



(a)

(b)

Figure 35: (a) SIU - DDL structure (b) Prototype implementation of CPV detector front-end read-out using off-the-shelf FPGA kits.

The block diagram shown in Figure 36 illustrates the various VHDL entities developed to emulate the DDL2 Source Interface Unit (SIU) for data transfer between a column and segment controller board. Two internal 100 MHz clocks from FPGA PLL drive the FPGA logic, while a 156.25 MHz reference clock is required to clock the 3.125 Gb/s full-duplex serial transceivers using Altera's physical transceiver IP (Intellectual Property) core. The FPGA logic sends and receives parallel 32-bit data words to and from both transceivers.

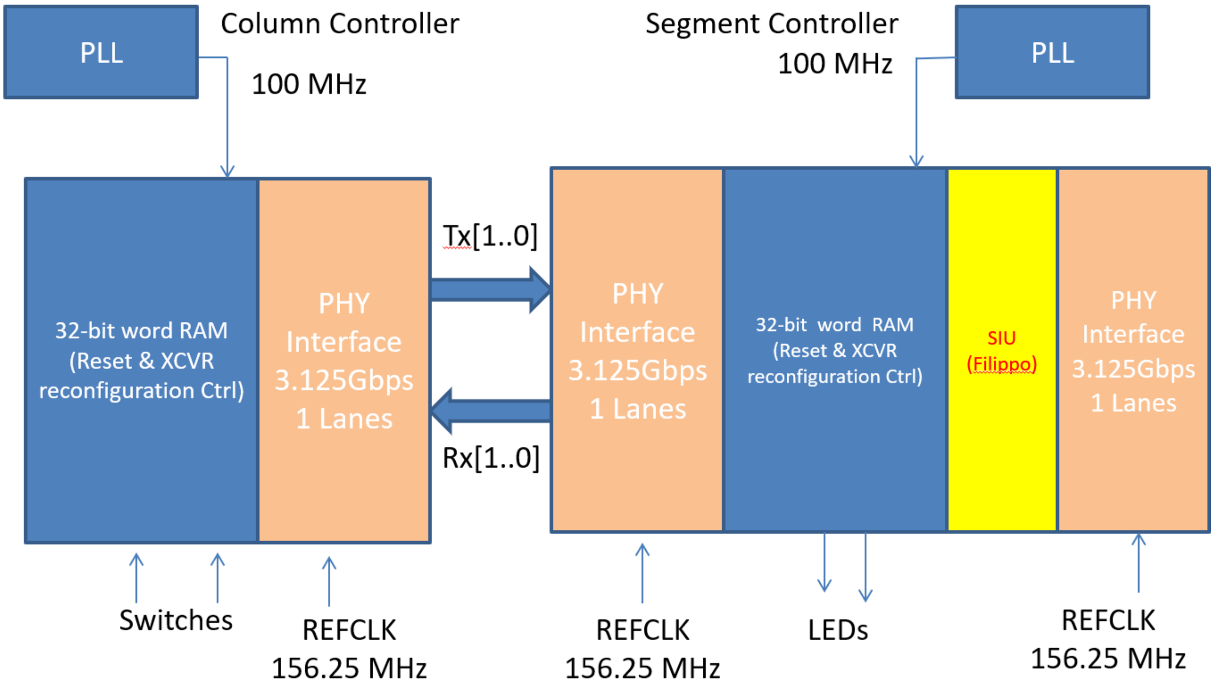
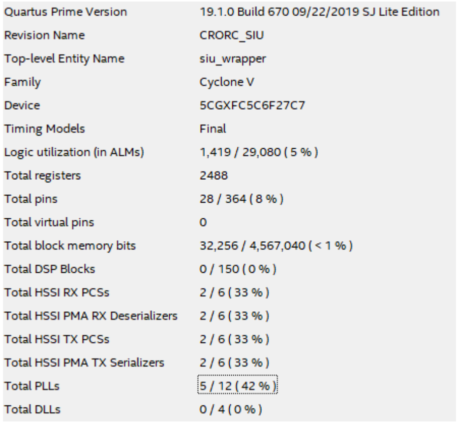

PLL
Column Controller
100 MHz
Segment Controller
100 MHz
PLL
32-bit word RAM (Reset & XCVR reconfiguration Ctrl)
PHY Interface 3.125Gbps 1 Lanes
Tx[1..0]
Rx[1..0]
PHY Interface 3.125Gbps 1 Lanes
32-bit word RAM (Reset & XCVR reconfiguration Ctrl)
SIU (Filippo)
PHY Interface 3.125Gbps 1 Lanes
Switches
REFCLK 156.25 MHz
REFCLK 156.25 MHz
LEDs
REFCLK 156.25 MHz

Figure 36: Block diagram illustrating the main VHDL entities of the prototype layout shown in Figure 35.

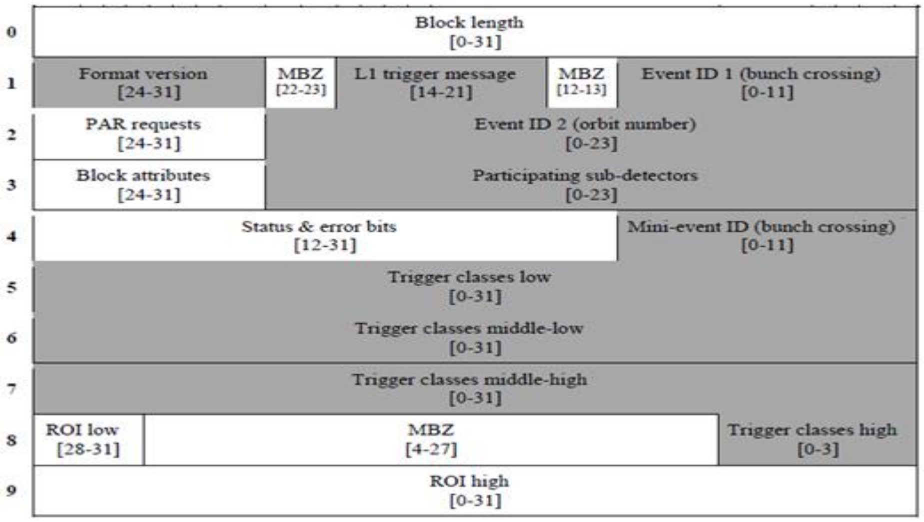
The segment controller contains the implementation of the standard DDL2 SIU interface required to transfer event data between CC and CRU DAQ experiment recorders. ALICE DDL2 is designed to meet all data transfer requirements between the detectors and the ALICE Data Acquisition System. The maximum DDL2 data rate is 5.125 Gb/s full duplex over a multipurpose fibre optic link with sufficient bandwidth for the DDL2 prototype transceiver to allow bi-directional data transmission between the front-end electronics CC and the DAQ system during event data acquisition. The use of this bidirectional DDL2 SIU link makes it possible for configuration control messages to be sent to the detector as needed during the start of each run.



Quartus Prime Version	19.1.0 Build 670 09/22/2019 SJ Lite Edition
Revision Name	CRORC_SIU
Top-level Entity Name	siu_wrapper
Family	Cyclone V
Device	5CGXFC5C6F27C7
Timing Models	Final
Logic utilization (in ALMs)	1,419 / 29,080 (5 %)
Total registers	2488
Total pins	28 / 364 (8 %)
Total virtual pins	0
Total block memory bits	32,256 / 4,567,040 (< 1 %)
Total DSP Blocks	0 / 150 (0 %)
Total HSSI RX PCSs	2 / 6 (33 %)
Total HSSI PMA RX Deserializers	2 / 6 (33 %)
Total HSSI TX PCSs	2 / 6 (33 %)
Total HSSI PMA TX Serializers	2 / 6 (33 %)
Total PLLs	5 / 12 (42 %)
Total DLLs	0 / 4 (0 %)

Figure 37: Cyclone V GX Resource Utilisation for the implementation of SIU, DDL2 and TLK2501 emulator.

The implemented VHDL netlist view for SIU, DDL2 and TLK2501 emulator consists of four main entities requiring a total of 32, 256 block FPGA memory bits and two transceivers. The TLK2501 entity implements the logic for a 2.5 Gb/s point-to-point transceiver [44] as shown in Figures 43 and 44. The DILOGIC Data Generator (DDG) instance creates the Common Data Header Version 3 (CDH V3), shown in Figure 38 below.



0 Block length [0-31]

1 Format version [24-31] | MBZ [22-23] | L1 trigger message [14-21] | MBZ [12-13] | Event ID 1 (bunch crossing) [0-11]

2 PAR requests [24-31] | Event ID 2 (orbit number) [0-23]

3 Block attributes [24-31] | Participating sub-detectors [0-23]

4 Status & error bits [12-31] | Mini-event ID (bunch crossing) [0-11]

5 Trigger classes low [0-31]

6 Trigger classes middle-low [0-31]

7 Trigger classes middle-high [0-31]

8 ROI low [28-31] | MBZ [4-27] | Trigger classes high [0-3]

9 ROI high [0-31]

Figure 38: Common Data Header (CDH) V3.

The CDH describes the associated data block(s), the trigger conditions, the error and status conditions and other information depending on the FEE RO electronics. Each data event to be sent over ALICE DDL must be preceded by a CDH, which contains information, like bunch crossing, orbit number, region of interest, trigger classes (physics or software), and other. Further, it provides the necessary information needed to identify and label the data within the downstream systems (DAQ, HLT, and Offline). While the header is mandatory and must be created for all data blocks sent over the DDL, the data blocks are optional and may be skipped in case there is no valid data associated to a given event. It is prohibited to send two CDHs with the same event identification information within the same run. CDH version 3 implements the required ALICE specifications and the 100 trigger classes. As shown in Figure 38, CDH V3 makes use of 10 words. Before proceeding with the actual design and development of all CPV electronic modules performance measurements were done to establish the preliminary readout rate using Cyclone V GX and DDL2 link. A Test jig shown in Figure 39 below, consisting of a Common Readout and Receiver DAQ module, including both segment and controller cards was setup at CERN.

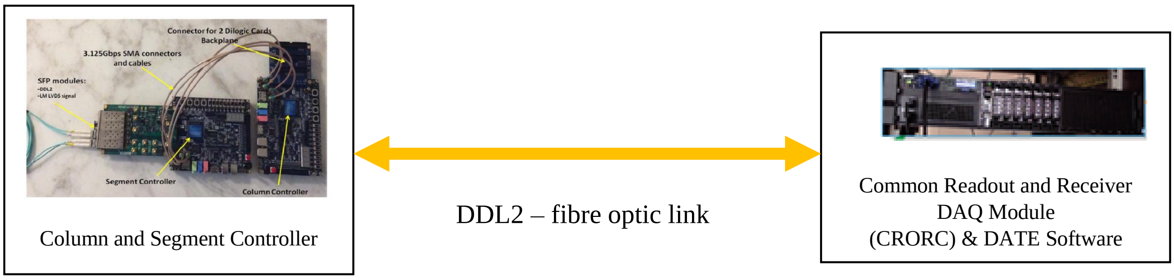


Figure 39: Setup for preliminary measurement performance of initial CPV prototype electronics.

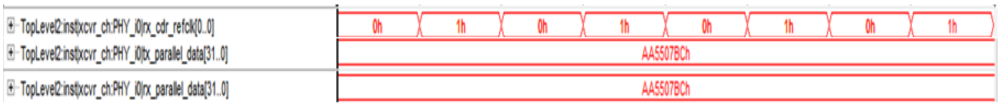

TopLevel2:inst|xcvr_ch:PHY_i0|rx_cdr_refclk[0..0]
TopLevel2:inst|xcvr_ch:PHY_i0|tx_parallel_data[31..0]
TopLevel2:inst|xcvr_ch:PHY_i0|rx_parallel_data[31..0]
0h
1h
0h
1h
0h
1h
0h
1h
AA5507BCh
AA5507BCh

Figure 40: Dummy data in loopback mode via DDL2 links.

Dummy data transmitted over bidirectional DDL2, 3.125 Gbps SFP links is shown in Figure 40. In addition, CRORC DDL2 [45] [25]is accessed via the software DATE. The following is an example of physics event data transmitted from the segment controller board to CRORC and accessed via DATE. This dummy physics event has a size of 1152 bytes and a header of 80 bytes. A total of 10,000 events were recorded at a trigger rate of 5.8 kHz, resulting in 0.02% of the frames received by CRORC being corrupted, with a frame size of 648 bytes instead of 1152 bytes. These problems were due to link stability and SIU first-in-first-out memory buffer overflow. The transfer of a 32-bit data word between column and segment controllers was found to be stable at a transceiver data transfer rate of 3.125 Gb/s. The timing diagram in Figure 42 shows a total transfer time of 20.2 µs for 8192 data bytes. At 15% occupancy, the device has a total of 1280 bytes, resulting in a transfer time of about 3.2 µs, which is slightly higher than the expected (3 µs) estimate given in Table 11 for reading all columns in parallel. The slight extra effort is related to the firmware.

```
Size:1152 (header:80) Version:0x0003000e Type:PhysicsEvent
RunNb:1190 Period:0 Orbit:23 BunchCrossing:0 ldcId:1 gdcId:VOID
time:Wed Aug 16 09:51:44 2017 +393974usec
Attributes:Orbit/BC+OriginalEvent(00000000.00000000.00000220)
triggerPattern: 0)00000001 1)00000001 2)00000001 3)80000001
detectorPattern:802fffff=(0=SPD-HW+1=SDD-HW+2=SSD-HW+3=TPC-HW+4=TRD-HW+5=TOF-
HW+6=HMPID-HW+7=PHOS-HW+8=CPV-HW+9=PMD-HW+10=MUON_TRK-HW+11=MUON_TRG-HW+12=FMD-
HW+13=T0-HW+14=V0-HW+15=ZDC-HW+16=ACORDE-HW+17=TRIGGER-HW+18=EMCal-HW+19=DAQ_TEST-
HW+21=AD-HW)

0) 00000430 00000011 00000801 00000000 | 0... .... .... .... |
16) 00000000 00000000 00000004 ffffffff | .... .... .... .... |
```

Figure 41: Partial data log for a dummy Physics event data recorded by CRORC, DATE software application.

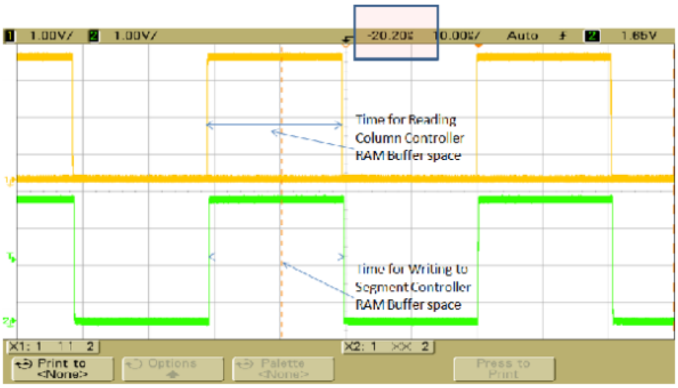


Figure 42: Transfer time from Column Controller RAM to Segment controller RAM buffers. Total Time = 20.20us, 2048 words, 32-bits/word, 8192 bytes.

Description	**Estimated Transfer Time (us)**
Analogue readout time of Gassiplex devices	5
Parallel Readout of all Dilogic cards	11.2
Parallel Readout of all Columns	3
Estimated Transfer time to DAQ using DDL2	8
Estimated Total Write Transfer Time	27.2

Table 11. Estimated RO rate for Parallel RO of Two DIL-5 cards.

Based on the above preliminary performance measurement results, the estimated CPV read-out rate using SIU-DDL2, 3.125 Gbps transceivers over SFP links, the simultaneous readout of two 5-Dilogic cards was estimated to be as described in Table 11 above. The chosen clock frequency for both DILOGIC and GASSIPLEX cards is 7.5 MHz, and an occupancy of 15% (72 channels) per column were taken into consideration. In reviewing the above preliminary estimated measurement results, both the ALICE collaboration and Russian Institute of High-Energy Physics accepted to proceed further with the upgrade design and production of the CPV readout electronics as will be explained in the following sections.

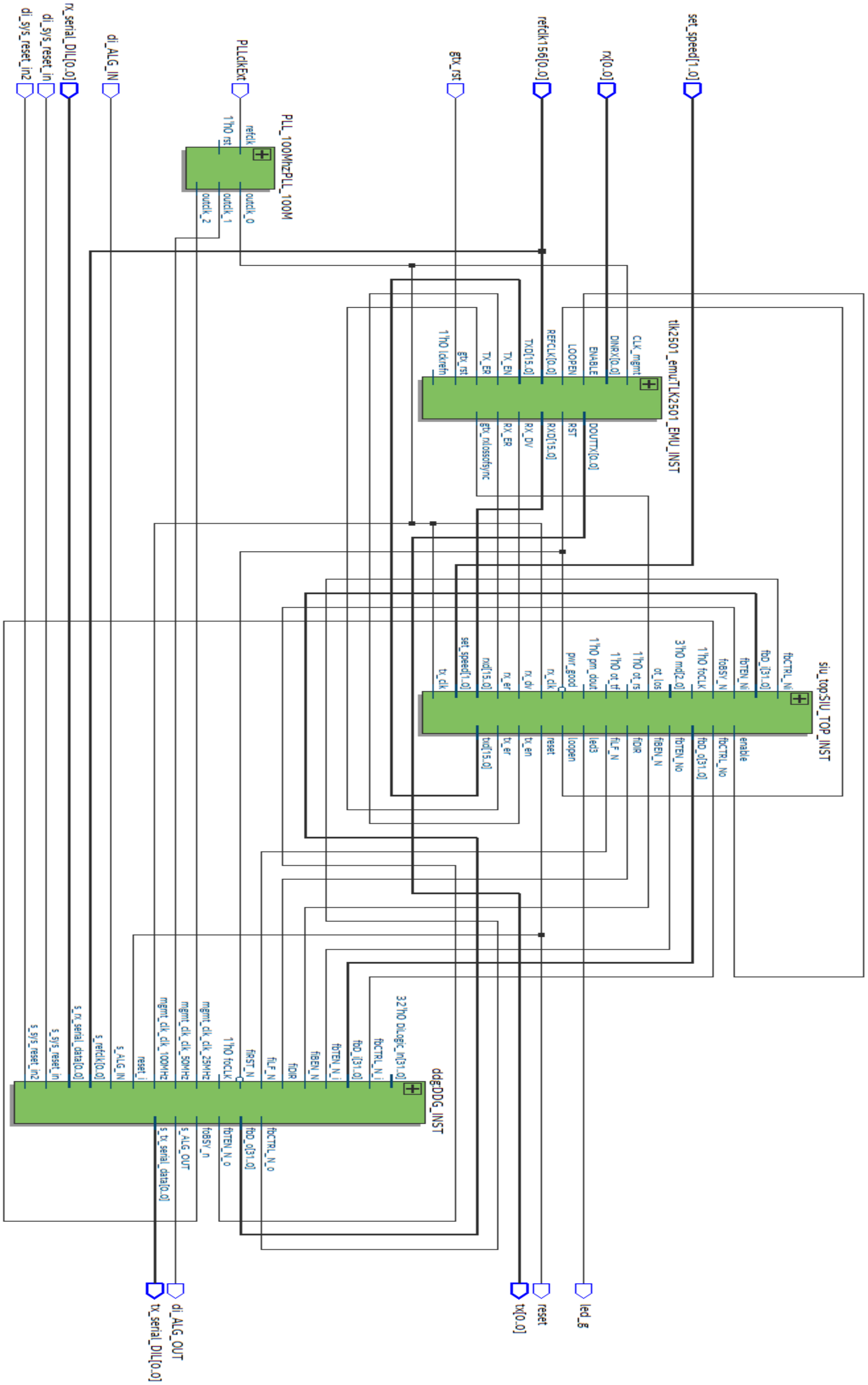


Figure 43: Netlist View of TLK2501 Emulator for the implementation of SIU and DDL2 protocols using a Cyclone V GX FPGA.

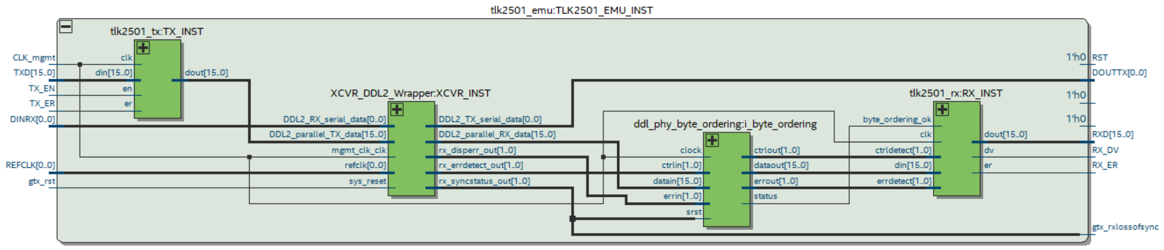


Figure 44: Netlist view for the VHDL implementation of the TLK2501 device, including DDL2 transceivers.

3.2 Proposed FEE Readout Architecture for CPV Detector

In Run3, the CPV is divided into 3 independent modules installed before 3 PHOS modules in sectors 260-280°, 280-300°, 300-320°. One module was installed during LS1 and is now working in Run 2. Two more modules will be installed in LS2 for Run 3. The goal of CPV is to improve photon identification in PHOS by suppressing charged clusters, and the events recorded by PHOS and CPV should be synchronous. Since PHOS plans to upgrade the readout firmware to 30-40 kHz, CPV should be no slower than PHOS (possibly even faster). The current CPV electronics have a maximum readout rate of 5 kHz, which can be increased to 10 kHz with a firmware upgrade alone. A further increase of the readout rate is not possible without a hardware upgrade.

The two new CPV modules are not equipped with readout cards at all, as it is difficult to reproduce old cards due to obsolete components and technologies. The front-end cards (GASSIPLEX, DILOGIC and 10-bit ADCs) were already manufactured for all 3 modules before the actual hardware upgrade for Run 3 was required. New readout cards had to be developed and produced, maintaining the same data flow as the old readout cards, but taking the opportunity to increase speed. For the original CPV upgrade project for Run 3, the readout hardware was already in the works before an official proposal for this new CPV FEE readout architecture was submitted. As requested by ALICE, the goal is to retain the old charge-sensitive amplifiers (3-GAS - 480 cards) and ADC (5-DIL cards - 96 cards), while upgrading the Column Controllers CC (from 16 CC /module to 8 CC /module), with each CC controlling two columns of 480 GASSIPLEX channels each, for a total of 24 cards. In addition, the segment main boards are upgraded to two per module, resulting in a total of 6 segment boards. The upgrade for the readout control board includes parallel readout of CC via 3.125 Gb/s FPGA serial transceiver links and replacement of SIU with GBT for communication with CRU DAQ. The implementation of this new CPV FEE readout system architecture is mainly based on the custom development of reliable high-speed serial bus technologies and parallel processing techniques, enabling the system to use more channels and resources simultaneously and synchronised. The system's parallelism was achieved through a custom hardware design for the segment and column controller interface boards, which enables simultaneous processing of 960 analogue channels per CC, doubling the space required compared to the previous electronic CPV system. In addition, this new system upgrade includes the complete redevelopment of new VHDL system hardware models, behaviours, and structures such as column controller firmware for simultaneous data acquisition and transmission, resulting in a tenfold increase in data throughput compared to the previous electronics. Remote programming of all CC cards is accomplished via a 25 m shielded cable between the control room and CC FPGAs using Ethernet and Joint Test Action Group (JTAG) communication protocols, as explained in the following section. In addition, the proposed architecture includes the complete redesign of the digital communication interface between the DILOGIC cards and the column controller, the use of 28nm Cyclone V GX FPGA technologies, the use of 3.125 Gb/s high-speed transceiver full-duplex serial links, and the complete redesign of the firmware for continuous readout trigger mode. Each detector is divided into sub-parts (modules, partitions, etc.) with a finite number of channels. The newly selected CPV detector topology is based on three independent modules, corresponding to 6 partitions or 2 segment cards per module, each with four 3.125 Gb/s high-speed transceiver links per module. This new electronic detector scheme implements VHDL for DILOGIC's segment and column controller boards on Cyclone V GX FPGA technology. A trade-off between cost and performance was made in the selection of the FPGA.

In the previous system CPV FEE, the sequential analogue readout from the daisy-chained DILOGIC chips occurs after the asynchronous L1 message trigger arrives through the FPGA column controller and takes an estimated 100µs, depending on the readout frequency of the DILOGIC clock, which ranges from 5 MHz to 20 MHz. To achieve an event readout rate of 50 kHz, the readout time of the analogue pattern of the DILOGIC card must be drastically reduced. Such a reduction in readout time can be achieved by reading out DILOGIC chips simultaneously in parallel. Reading out all ten DILOGIC chips could result in too many copper traces on the PCB, contributing to trace resistance and degrading effects. This is because each DILOGIC has an 18-bit digital bus. It would also require a larger FPGA footprint and a new

DILOGIC board design, which would double the implementation cost of such an upgrade. Therefore, another alternative is to use two independent DILOGIC IOBUS data buses and a VHDL controller for the independent groups of 5 DILOGIC boards, as shown in Figure 45. This approach can be implemented on currently available DILOGIC boards without incurring additional costs to manufacture DILOGIC boards. With this architecture, data can be transmitted over IOBUS_1 and IOBUS_2 simultaneously. The digitised amplitude information (12 bits) and address for a given analogue channel are stored in a FIFO buffer (256 x18-bit words). Two separate buffers are implemented for each column, each containing raw data. A VHDL controller scans both FIFO buffers and serially transmits the collected data to the segment controller over a full-duplex serial link at an estimated speed of 3.125 Gb/s in one direction.

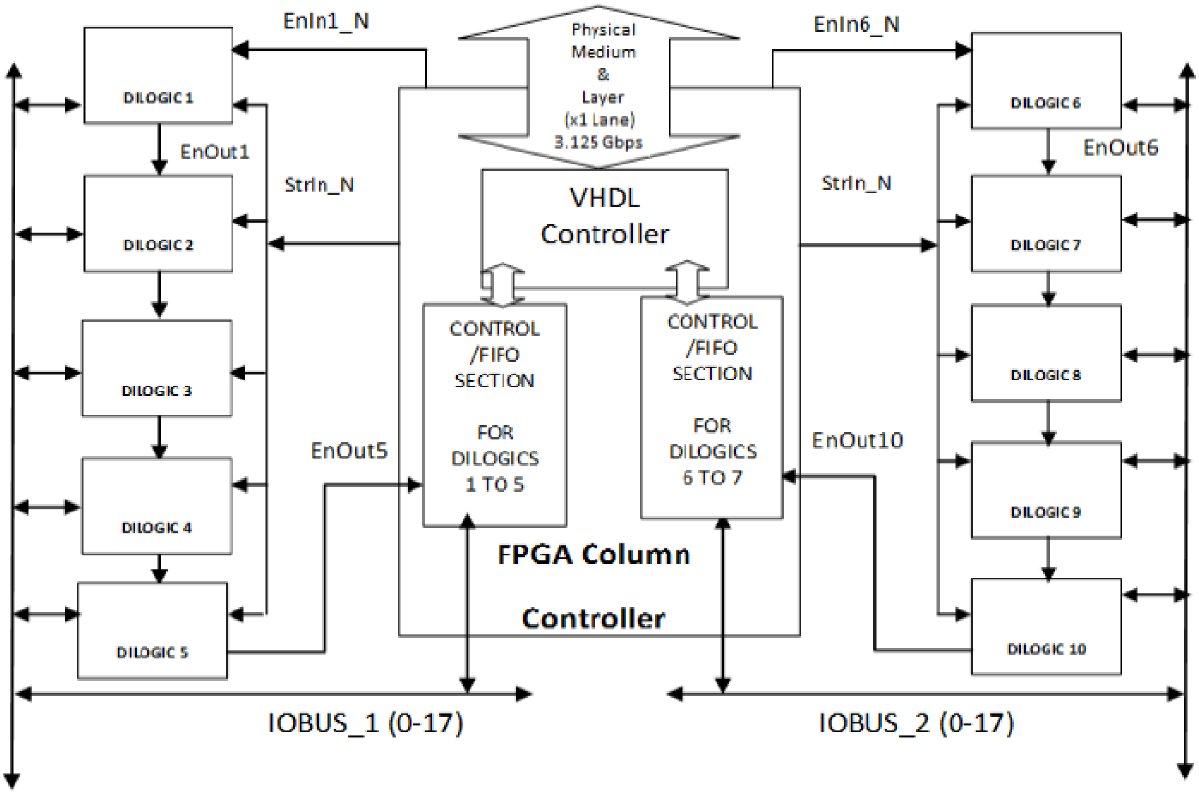


Figure 45: Layout topology for a new Column Controller card.

All upgraded boards are shown in Figure 46, which illustrates the complete system architecture for the new CPV FEE Readout detector hardware. Each module includes the re-design of a motherboard interface card called Segment board, which allows the concurrent processing of four DILOGIC cards via an FPGA column controller card containing a 28nm Cyclone V GX Altera device.

A low-material Inner Tracking System (ITS) readout unit is used for transmitting information to run the experiment over a radiation-hard Gigabit transceiver (GBT) link chipset to a Data Acquisition (DAQ) Common Readout Unit (CRU) at a high speed of (~5 Gb/s). Furthermore, the radiation hard GBT link provides the simultaneous transmission of trigger and experiment control data over the same optical link.

The newly developed hardware, shown in Figure 46, allows simultaneous readout of two analogue column channels with 480 channels per column. This drastically reduces the readout time of the DILOGIC cards by more than 50% compared to the current CPV and High Momentum Particle Identification Detector (HMPID) readout detector systems. In addition, this architecture lowers implementation costs and improves detector luminosity because, unlike the existing system, each FPGA column controller processes signals from two columns instead of one. The segment motherboard is designed to house four 5-DIL cards and one Column Controller (CC) card. The purpose of the segment board is to connect the data from each group of four 5-DIL cards to a column controller. Voltage translators are required to translate from a 5 V voltage level of the DILOGIC card to a 3.8 V voltage level as dictated by the Altera Cyclone V GX FPGA general purpose input and output (GPIO) pins. In addition, the segment card is used to distribute the DC power supply and the Joint Test Action Group (JTAG) for FPGA programming. One CPV module provides space for 8 segment cards.

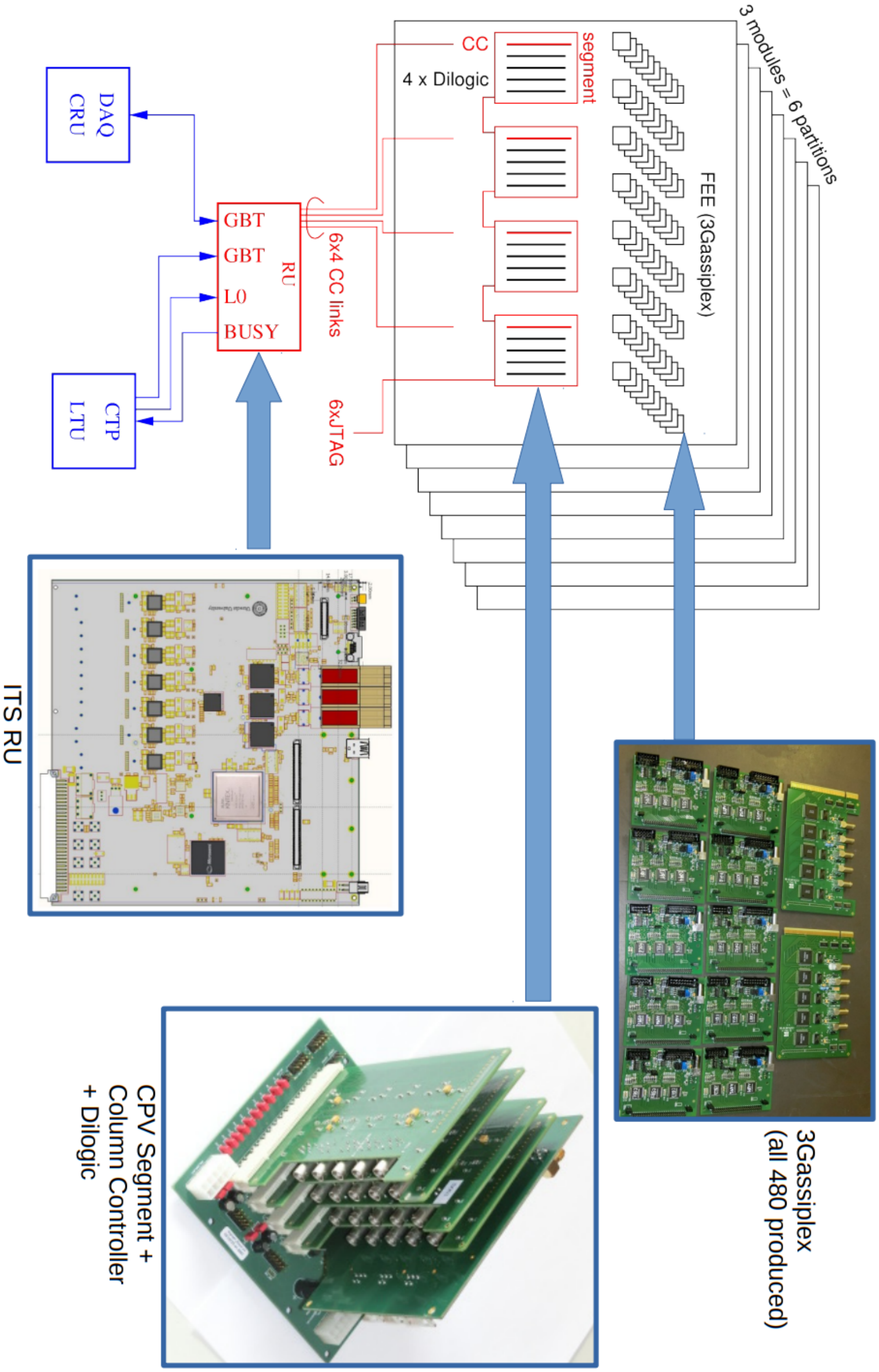


Figure 46: Complete System Block diagram for the proposed New CPV Readout Topology.

3.2.1 Data Transfer Scheme between FEE and ITS RU

All CPV front-end readout electronics are housed in three independently installed CPV modules, with access to FEE only available from LS3. The ITS readout unit is located outside the CPV modules at 5m. Therefore, an LVDS patch panel is required along with a transition board to provide the necessary interface between CPV FEE and ITS-RU. Figure 48 shows the data flow diagram between FEE CC, LVDS patch panels and transition board. The SMA coaxial 5 m shielded cables are intended for transmitting event data via gigabit transceivers. The purpose of the ERF8-025 Firefly 16 Gbps Samtec high-speed shielded cables is to transmit control commands from ITS-RU to FEE CC boards using a standard LVDS signalling scheme. The transfer rate for control and event data over the Gigabit and LVDS transceivers is 3.125 Gbps and 0.4 Gbps, respectively. The synchronous LVDS interface consists of the following:

- **LVDS_CLK_IN** – Clock signal from the ITS-RU RX module,
- **LVDS_RX_IN** – A Single serial lane differential input from the ITS-RU RX module.

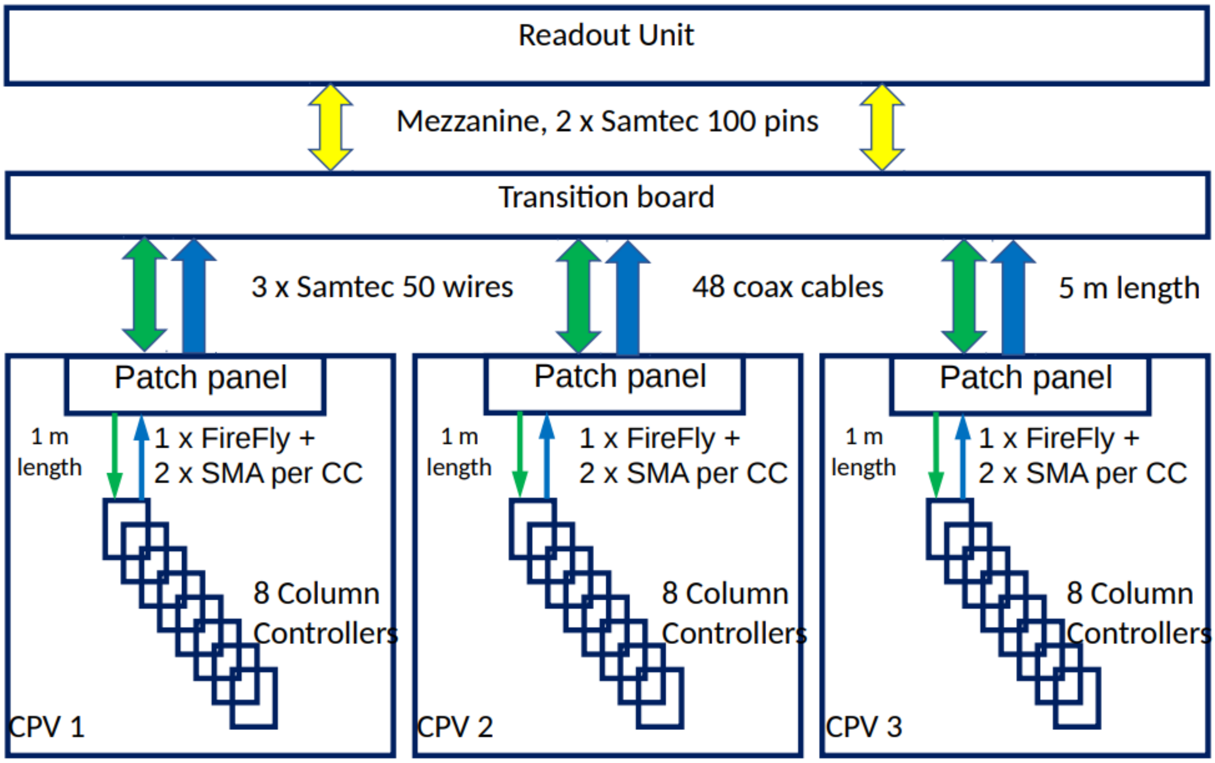
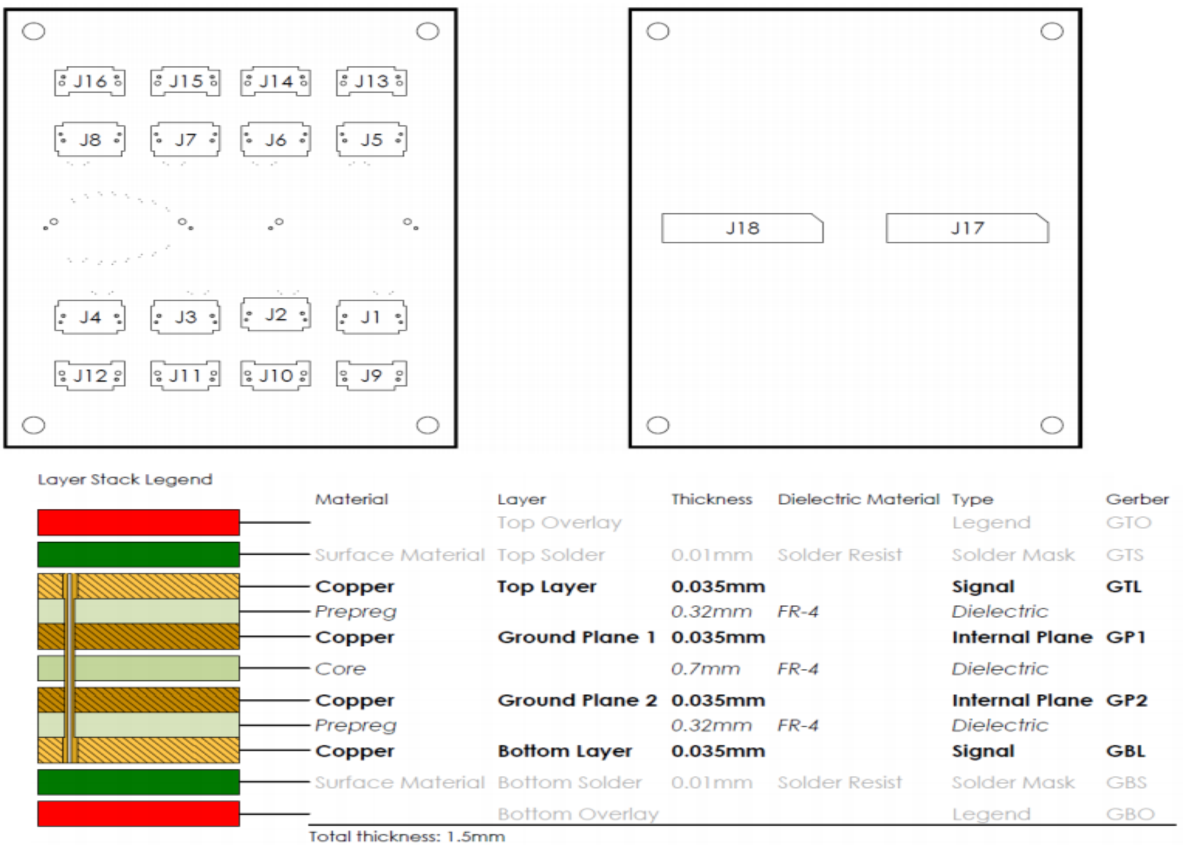


Figure 47: Data flow and transfer scheme between FEE and ITS RU.



Layer Stack Legend

	Material	Layer	Thickness	Dielectric Material	Type	Gerber
🟥		Top Overlay			Legend	GTO
🟩	Surface Material	Top Solder	0.01mm	Solder Resist	Solder Mask	GTS
▨	**Copper**	**Top Layer**	**0.035mm**		**Signal**	**GTL**
□	*Prepreg*		*0.32mm*	*FR-4*	*Dielectric*	
▨	**Copper**	**Ground Plane 1**	**0.035mm**		**Internal Plane**	**GP1**
□	*Core*		*0.7mm*	*FR-4*	*Dielectric*	
▨	**Copper**	**Ground Plane 2**	**0.035mm**		**Internal Plane**	**GP2**
□	*Prepreg*		*0.32mm*	*FR-4*	*Dielectric*	
▨	**Copper**	**Bottom Layer**	**0.035mm**		**Signal**	**GBL**
🟩	Surface Material	Bottom Solder	0.01mm	Solder Resist	Solder Mask	GBS
🟥		Bottom Overlay			Legend	GBO

Total thickness: 1.5mm

Figure 48: The 4-layer PCB Stack layout for the LVDS patch panel interface.

3.2.1 DILOGIC Column Controller Card

Figures 51 and 52 show the hardware layout of the eight-layer column controller electronic board. The ALTERA Cyclone V GX FPGA performs parallel RO of all four 5- DIL (two columns) via two separate 18-bit digital buses (D0 - D17), with event data then to be transmitted serially via high-speed transceivers over SMA cables at a rate of 3.125 Gb/s. Control data will be received via LVDS (Low Voltage Differential Signal) Firefly cables at a rate of 0.4 Gb/s. The implementation uses an edge card connector that connects to an HSMC slot with a 200-pin connector with 0.8 mm pitch on the segment board. A CPV module provides space for 8 column controllers. The Si5338C programmable clock based on ROM provides the 156.25 MHz reference clock signal for the 3.125 Gbps transceiver SMA link. The Si5338C can be reprogrammed using the Inter-Integrated Circuit (I2C) serial communication protocol. The three voltage regulators provide power to the FPGA core, the IO banks, and the transceivers. Due to the 1 to 2 T magnetic field strengths present in the cavern, the use of linear regulators rather than switching power supplies was recommended to power the FPGA.

To update the firmware version, it is necessary to develop the new design in a dedicated development environment (ALTERA Quartus II), test and debug the new firmware version, and finally load the binary file into the FPGA. The firmware can be rewritten each time an enhancement operation is required. The upload operations are subject to pit access control during accelerator activity. Therefore, it is useful to have the ability to perform uploads of new firmware versions remotely to avoid direct access to the pit. The ALTERA Ethernet Blaster adapter can convert the JTAG protocol to an Ethernet TCP/IP socket in the pit. The programming access points become accessible via the technical network ALICE without the need for direct access to the cavern.

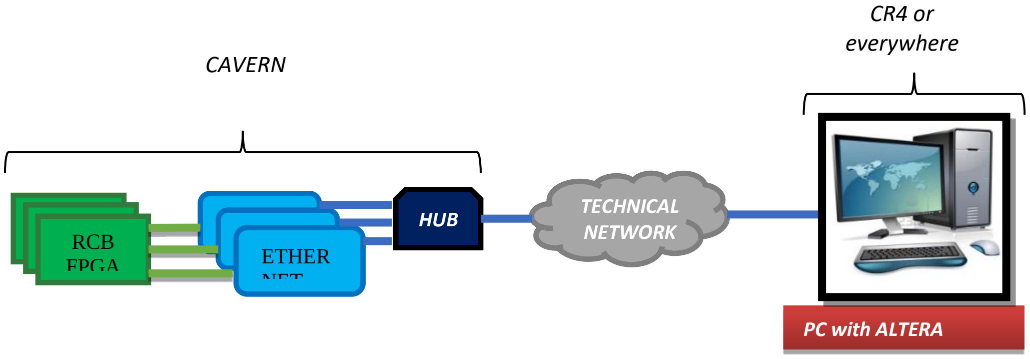


Figure 49: Remote programming system.

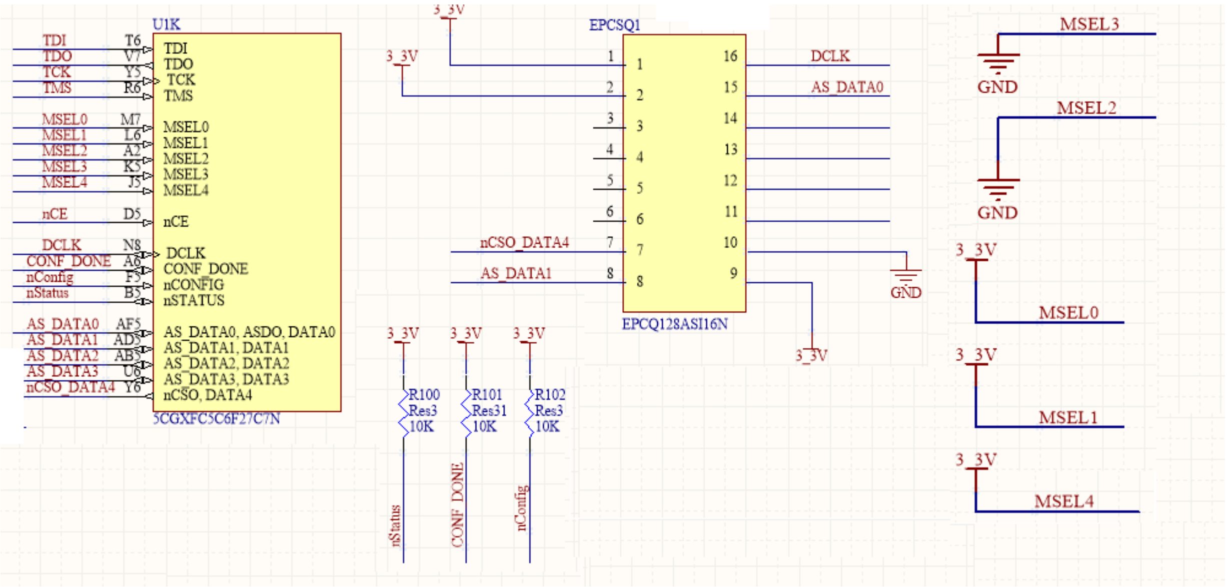


Figure 50: Circuit diagram for the Active Serial FPGA configuration.

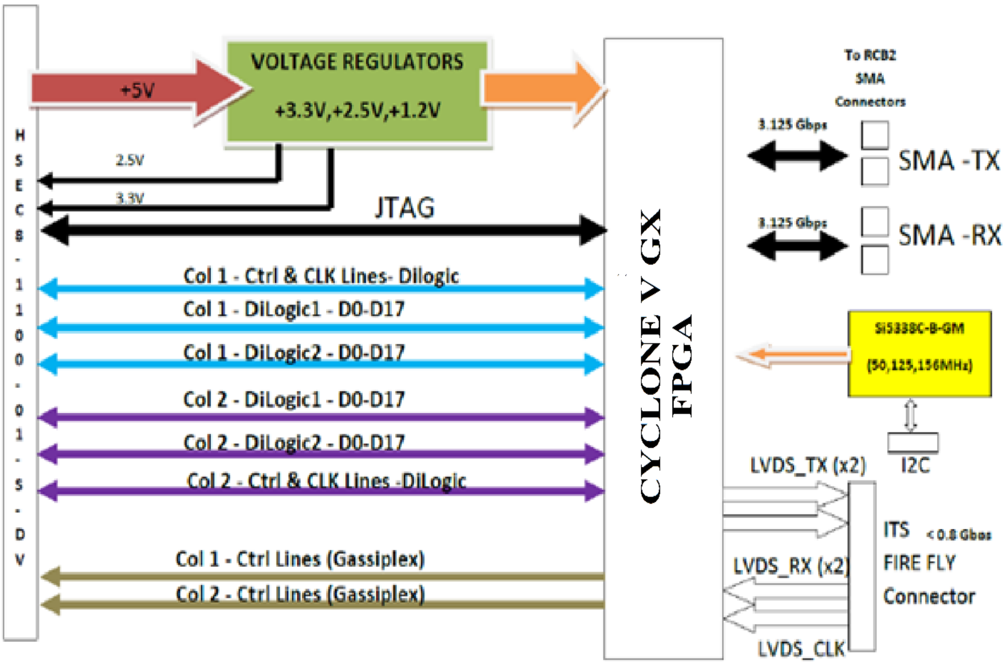


Figure 51: System Block Diagram for Column Controller Card.

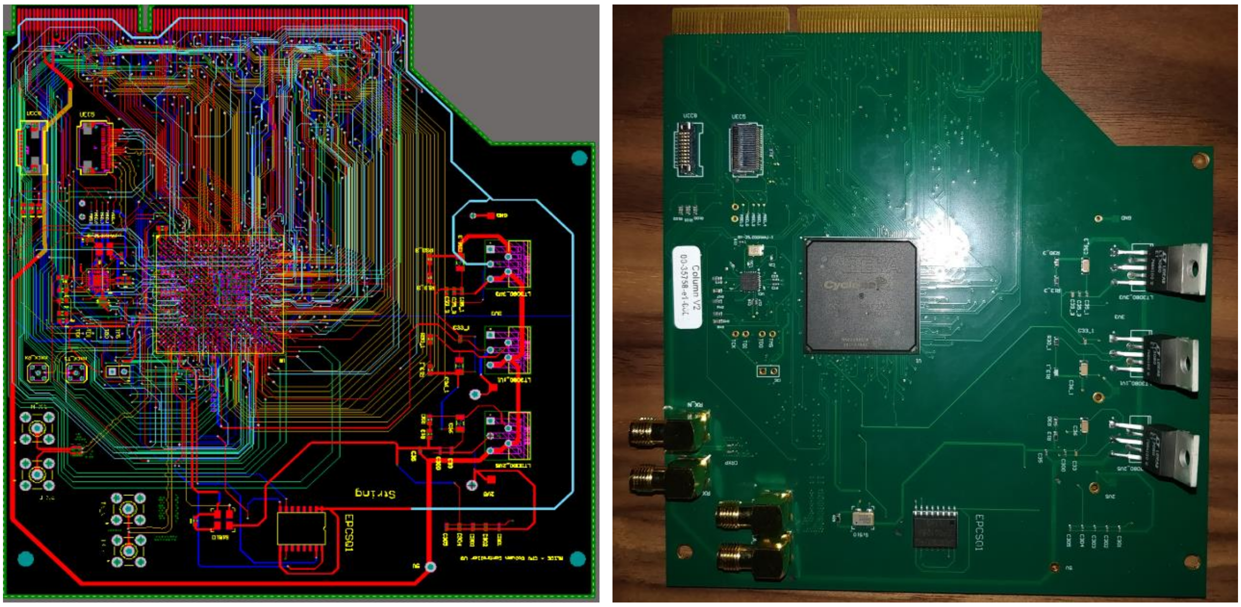


Figure 52: 8-layer PCB Layout and Manufacturing - Column Controller Card.

Due to the length of the flat cables (about 25 m) from the programming workstations to the FPGA in the magnet, which do not conform to the JTAG standard, remote programming is successful only at low programming frequencies: in tests, the maximum frequency for uploading an integer file to the FPGA is 1 MHz, which is a tenfold increase compared to earlier electronics. Such improvement is due to the introduction of the SN74LVTH241DBR buffer device.

The 3.3V EPCQ128ASI16N device with 134,217,728 memory bits located on the column controller board is used to configure and programme the SRAM FPGA -based column controller using the Active Serial (AS) scheme [46].

Among the items one should consider when designing printed circuit boards is the use of power supply planes, which, whether it is a full plane or a split plane, should have an adjacent ground GND plane. Also, the separation between the power supply plane and the GND plane should be thin to increase coupling and capacitance between the planes for example typically between 3 to 4 mils. Also, the power supply pins should preferably be connected directly to the corresponding power supply planes to minimize inductance. Narrow traces should be avoided to connect the power supply pins to the power supply plane. Spacing between power supply planes can range from 25 to 100 mils. Sharing a single via hole with multiple power supply pins should be avoided. Each power supply connection should have at least one dedicated through-hole connection to the power supply plane. Other PCB design guidelines have been followed and discussed with the manufacturer.

The number, type, and value of decoupling capacitors needed to implement an efficient PCB decoupling strategy for the column controller board were estimated using Intel's Power Delivery Network (PDN). In addition, for each of the FPGA power supply pins shown in Figure 51, a specific type and number of bulk or ceramic decoupling capacitors must be selected using the PDN tool. The use of such a tool is recommended by the FPGA manufacturer so that you do not need to perform extensive design simulations before layout [47]. The recommended supply settings for the Cyclone V GX device are shown in Table 16. PCB decoupling capacitors are designed to reduce the impedance of PCB PDN in the 50 to 100 MHz range. The inductance of discrete capacitive network components is usually higher than the capacitance between power ground planes, which is why the latter is more effective up to frequencies of 100 MHz. This is due to the parasitic inductances of FPGA, PCB, packing and ball grid array (BGA). The PDN tool accurately models such effects using lumped inductances and resistances, making it possible to determine the number and type of decoupling capacitors required. System logic and timing issues can arise due to supply voltage fluctuations. Furthermore, according to Ohm's law, such voltage fluctuations are proportional to circuit impedance and current flow. Such fluctuations can be reduced by minimizing the PDN impedance, i.e., by calculating the target impedance Z_{TARGET}, as follows:

$$\boldsymbol{Z_{TARGET}} = \frac{\boldsymbol{Voltage\ Rail\ x\ \frac{\%Noise\ Tolerance}{100}}}{\boldsymbol{Maximum\ Cahnge\ in\ Dynamic\ Current}} \qquad \ldots (1)$$

Table 16. Recommended Power Rail Configuration Parameters for Cyclone V GX 5CGXFC5C_F27 device.

Description	Value
Device	GX 5CGXFC5C_F27
Family	Cyclone V
Total Thermal Power dissipation	510 mW
Core Dynamic Thermal Power Dissipation	132.33 mW
Core Static Thermal Power Dissipation	272 mW
IO Thermal Power Dissipation	106.36 mW
DC supply Voltage – VCC core	1.1 V
Switcher VRM Efficiency	90%
Switcher VRM Input Current	0.5 A
Total Current	0.5 A
Dynamic Current Change	50%
Die noise Tolerance	5%
Core Clock Frequency	500 MHz
Current Ramp Up Period (core clock cycles)	75
VCCIO - IO Bank	2.5 V
VCCE_GXBL - Transceiver TX & RX (Analog)	1.1 V
VCCH_GXBL - Transceiver CDR, Tx Buffer and PLL	1.1 V
VCCL_GXBL – Transceiver Core Clock	1.1 V
VCC_AUX – Auxiliary Power	2.5 V
VCCPD – I/O pre-drivers	2.5 V
VCCBAT – Battery Backup Power Supply	2.5 V

Further, taking into consideration the simulated power report generated by the Quartus Power estimation tool, the Power Rail parameters for the Cyclone V GX device shown in Table 16, were obtained for calculating the required Z_{TARGET} value of 0.22 Ω. The PCB decoupling via discrete PCB capacitors becomes ineffective at higher frequencies which are beyond their range, thus leading to an increase in PCB space, BOM and cost. As calculated and modelled by Intel' PDN design tool, and as shown in Figure 53, the recommended PCB decoupling design cut-off frequency ($F_{EFFECTIVE}$) is 60 MHz. Therefore, the PCB low impedance profile (Zeff) should be less than Z_{TARGET} up to $F_{EFFECTIVE}$, which can be achieved through the correct use, value, and quantity of decoupling capacitors. Z_{SPREAD} and Z_{VIA} shown in Figure 53, represent the plane spreading and BGA via inductances, which their series combination adds up to Z_H (Z_{SPREAD} + Z_{VIA}).

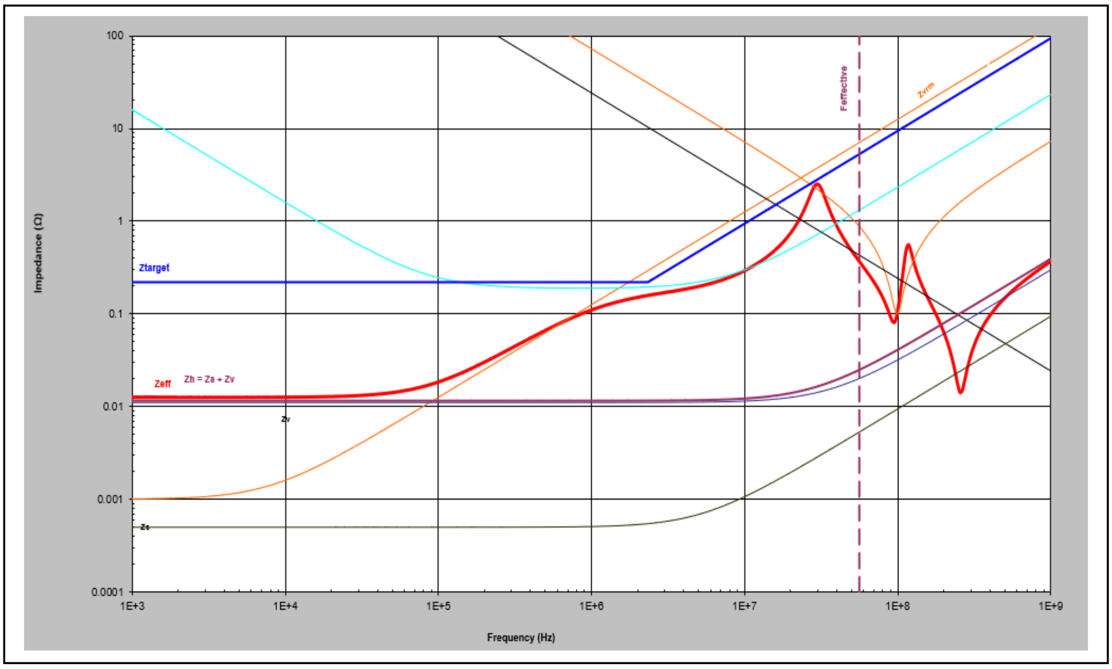
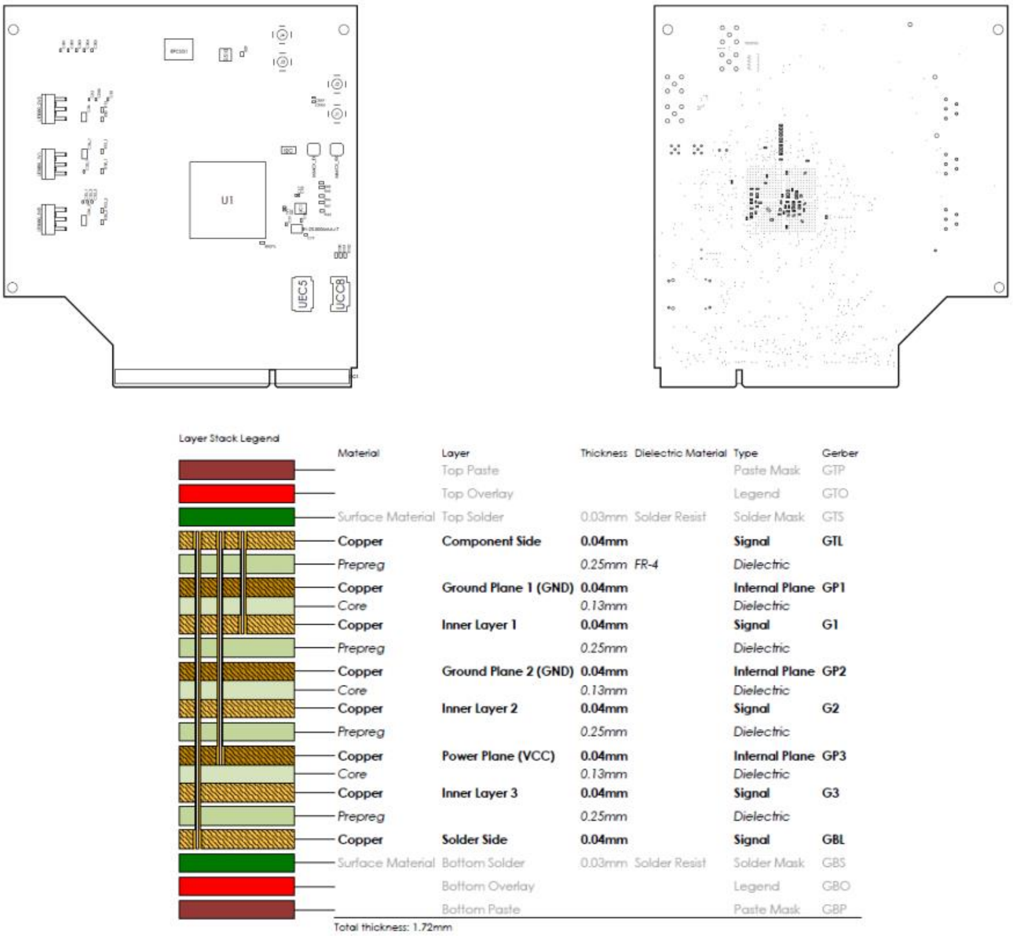


Figure 53: Illustration of low impedance profile (Zeff) below the target impedance (Z_{target}) for the selected 8-layer PCB layout design.



Material	Layer	Thickness	Dielectric Material	Type	Gerber
	Top Paste			Paste Mask	GTP
	Top Overlay			Legend	GTO
Surface Material	Top Solder	0.03mm	Solder Resist	Solder Mask	GTS
Copper	Component Side	0.04mm		Signal	GTL
Prepreg		0.25mm	FR-4	Dielectric	
Copper	Ground Plane 1 (GND)	0.04mm		Internal Plane	GP1
Core		0.13mm		Dielectric	
Copper	Inner Layer 1	0.04mm		Signal	G1
Prepreg		0.25mm		Dielectric	
Copper	Ground Plane 2 (GND)	0.04mm		Internal Plane	GP2
Core		0.13mm		Dielectric	
Copper	Inner Layer 2	0.04mm		Signal	G2
Prepreg		0.25mm		Dielectric	
Copper	Power Plane (VCC)	0.04mm		Internal Plane	GP3
Core		0.13mm		Dielectric	
Copper	Inner Layer 3	0.04mm		Signal	G3
Prepreg		0.25mm		Dielectric	
Copper	Solder Side	0.04mm		Signal	GBL
Surface Material	Bottom Solder	0.03mm	Solder Resist	Solder Mask	GBS
	Bottom Overlay			Legend	GBO
	Bottom Paste			Paste Mask	GBP

Total thickness: 1.72mm

Figure 54: Column Controller PCB Board Layer Stacking.

3.2.2 Column Controller Firmware

The Column Controller firmware was designed and implemented using VHDL and Intel IP cores. The main modules of CC VHDL include a main controller, a data wrapper, a 3.125 Gbps Pseudo Current Mode Logic (PCML) transmitter, an LVDS receiver data wrapper, DILOGIC controller, front-end controller, PLL, FIFO, RAM. The LVDS controller consists of Intel IP LVDS _RX and an alignment finite state machine (FSM) in the LVDS wrapper. Alignment of the link is achieved when two consecutive "IDLE" words are received, otherwise a bitslip procedure is performed on the LVDS_RX link by the LVDS wrapper. In addition, such an alignment is performed automatically after power-up reset. The realignment can be done via x'F. Decoding and deserialization of an incoming bit stream from the ITS RU via the LVDS link is performed by the module LVDS receiver module.

The Column Controller receives both the L0 trigger, thresholds for each channel, and control commands from ITS-RU via the LVDS receive link. The LVDS link is configured to operate at 200 Mbps and transmission is over a 4-bit wide data frame. The received data format consists of a StartOfFrame (SOF), 3 data words, and an EndOfFrame (EOF). As shown in Table 17, the main controller converts the three sequential LVDS words into an 18-bit word that DILOGIC can process.

17 16	13	12 9	8 0
W1 [0]	Word 2	Word 3	Zero
Compare Threshold Memory			Pedestal Subtraction Memory

Table 17. DILOGIC Data Word format.

The CPV DILOGIC subtraction memory is always zero. When you enable zero suppression (ZS), DILOGIC performs a pedestal subtraction that suppresses data below the threshold. Data values above the threshold are stored in memory with an unsubtracted value. No commands are available to start the write configuration process. After receiving the SOF, the main controller starts filling the RAM. There is a total of four 5 DILOGIC cards with 64 channels, so the main controller should receive 1280 threshold values, which are then distributed to each of the four RAM modules accordingly. Upon completion of writing the thresholds to RAM, the main controller generates a pulse to transfer the socket configuration from RAM to all the DILOGIC chips. After the power-on reset, the configuration memory is filled with zeros. When the RST off, the Column controller awaits the LVDS command, and the main controller is set to idle state. Table 18 illustrates the list of available LVDS commands.

Command	4-bit code	Description
Idle	0010	Code to indicate LVDS alignment. RU is connected and operational.
Start of Frame	0001	Used by the main controller to switch between commands and data. The next 3 words after the SOF will be data, not command.
End of Frame	0011	Used by the main controller to switch between commands and data. The EOF must follow the 3 words of data.
Configuration Read	0100	Read configuration memory of all four cards 5-DIL and transmit data to the GX.
Zero Suppression off	0101	Disable zero suppression (disabled by default).
Zero Suppression on	0110	Enable zero suppression.
Trigger	0111	Start the readout of GASSIPLEX, then DILOGIC and send the data to GX.
Break the readout	1000	Stop the analogue readout, reset the GASSIPLEX, clear DILOGIC FIFO.
Reserved	1001….0111	Not used
Realign	1111	Reset the LVDS link and start alignment procedure.

Table 18. List of available commands.

The 3.125 Gbps transceiver modules are based on the Intel IP core and provide full control of the GX transmitter, serialisation, and 8b/10b encoding of 32-bit event data words. The front-end controller module provides the necessary GASSIPLEX control signals (T/H, Clear, Clock), the ADC clock, and other signals required to write digitised amplitudes to the DILOGIC's internal FIFO. The DILOGIC controller provides all the necessary control signals and 18-bit digital busses to transfer data between the four 5-DIL cards and the column controller. It also handles configuration reads and writes to DILOGIC's internal memory and performs analogue reads. The DILOGIC chip clock frequency for this controller is set at 10 MHz, as this is a limit for most operations with the DILOGIC chip. The dual-port random-access memory) RAM with different read and write clocks is used to transfer configuration data from the LVDS link to DILOGIC. FIFO memory allows data to be transferred from the DILOGIC module to the 3.125 Gbps transmitter. The design includes four sets of controllers, RAM and FIFO, one per 5-DIL card, and processes all 5-DIL cards simultaneously. Arbitration of the LVDS receiver, DILOGIC and front-end controller modules is handled by the main controller. Optionally, the LVDS connection can be used to reset various modules by external commands. An additional reset controller is used for the transceiver.

The column controller runs with the following clock sources:

- 50 MHz clock generated from the 200 Mbps LVDS link,
- 10 MHz clock generated by the PLL from the on-board oscillator of 50 MHz,
- 156.25 MHz reference clock for the 3.125 Gbps transceiver.

Figure 55 illustrates the main modules related to the firmware implementation of the column controller card.

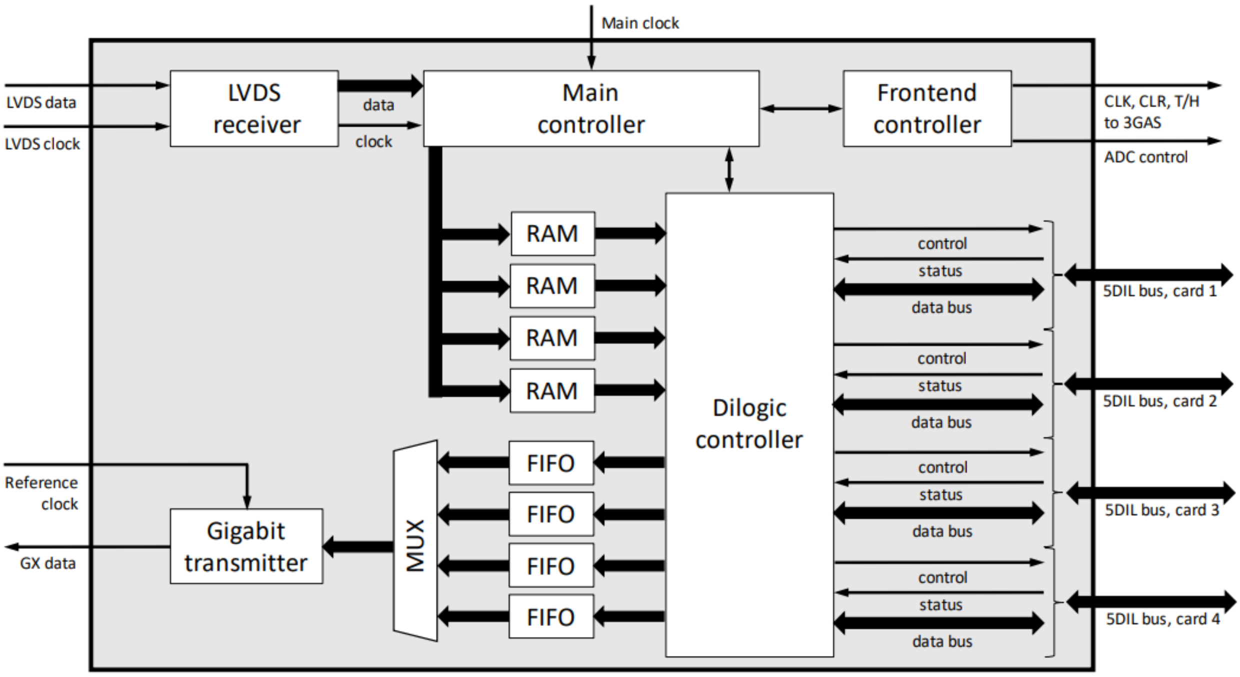


Figure 55: Illustration of the main VHDL Column Controller modules.

The front-end controller reads all digital data acquired by the GASSIPLEX and the 10-bit ADC in parallel to store them simultaneously in the internal FIFO memory of the respective DILOGIC chip. An FSM generates the required signals, which are distributed to all corresponding output pins of the CC interface board. The purpose of the front-end controller is to provide all the following signals:

- for the 3GAS cards (T/H, CLK, CLR),
- clock for the ADC chip (CLK_ADC),
- TRIG, CLKD, and channel counter for the 5-Dilgic card.

The procedure RO is started by CC when a trigger command or signal is received. This issues a command START to the frontend controller to set the T/H pin and issue the signal TRG_N for all 5 DIL boards. A total of 48 clock pulses (analogue channels per DILOGIC chip) are generated by the front-end controller after a total delay of 200 ns. There is a 180° phase shift between the 3Gassiplex (3GAS) clock and ADC clock signals, as shown on Figure 56. The channel number is represented by an inverted 6-bit counter. An additional CLKD pulse is generated by the front-end controller at the end of the sequence to write the EndOfEvent word to the DILOGIC FIFO. After the T/H signal is released and after a delay of 200 ns, a CLR pulse is generated for the 3GAS cards. A BREAK command received over the LVDS link can be used to interrupt an analogue readout, instructing the front-end controller to immediately release the T/H signal and output the CLR signal.

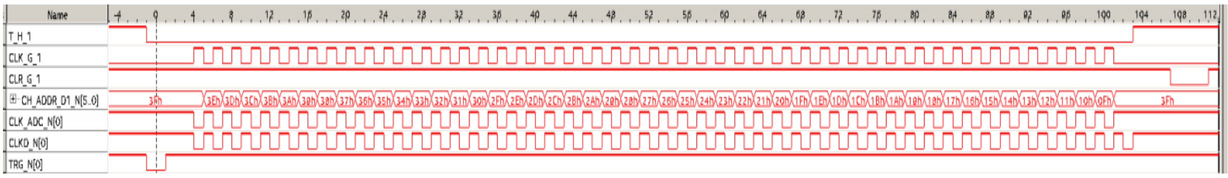


Figure 56: Generated signals by the Frontend CC (50ns per division).

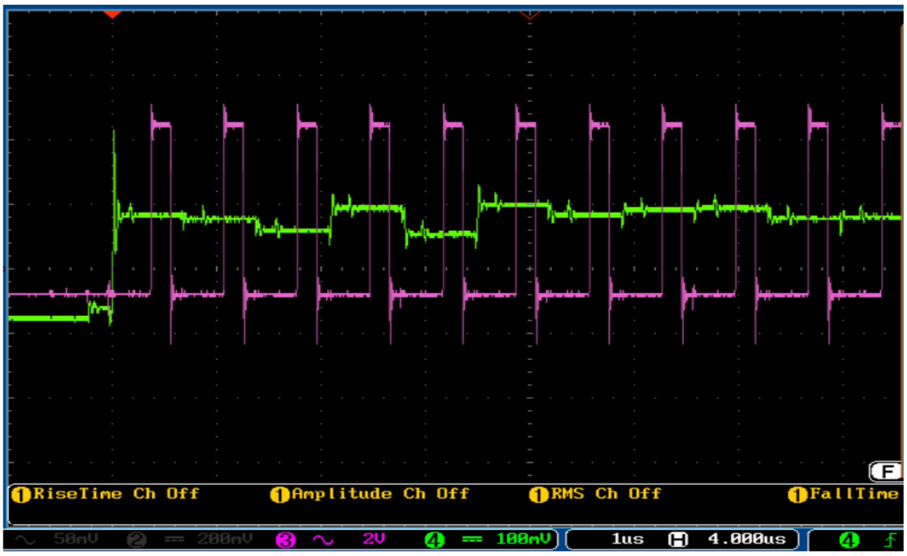


Figure 57: 3GAS Analogue output (green) and ADC clock (purple) generated by CC.

The DILOGIC controller module consists of two levels, a top-level unit that provides synchronization and generates FCODE and other common control signals for all boards, and a lower-level unit consisting of 4 independent individual controllers for each 5-DIL board. Each individual controller communicates with the 5-DIL cards via the 18-bit digital DATABUS and the STRIN, ENIN, ENOUT control signals.

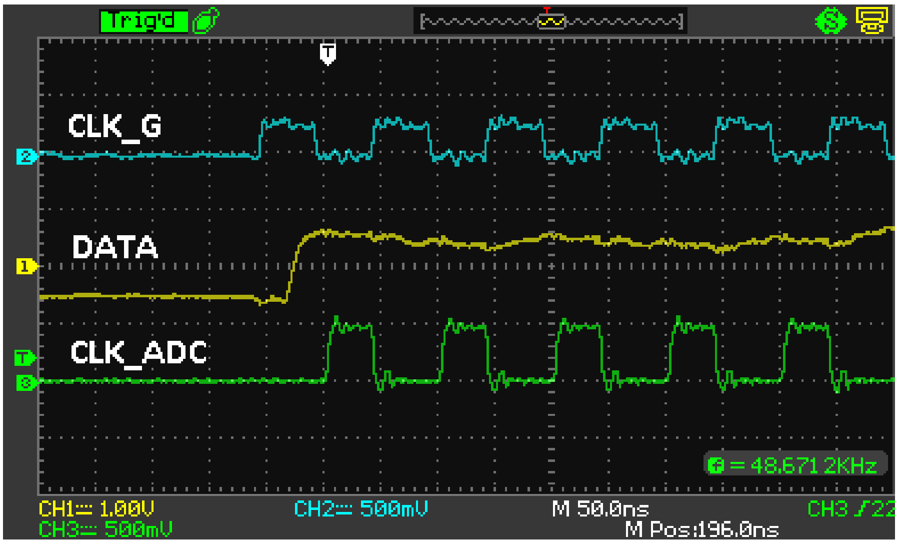


Figure 58: Analogue input voltage (yellow), 3GAS clock (blue), and ADC clock (green) generated by Column Controller.

The 3.125 Gbps transceiver controller, implemented by various Intel IP cores, includes a gigabit transmitter, a separate reset controller and a reconfiguration controller. The data wrapper module includes 4 FIFO instances and a multiplexer to sequentially read data from all 5-DIL cards and transmit it through the Gigabit controller. A 10 MHz clock signal is used to write one 32-bit to each FIFO. The multiplexer works on the round-robin principle, i.e., it reads the data from each individual FIFO with a clock frequency of 78.125 MHz. An "idle" word (0xC5BCC5BC) is sent to the Gigabit transmitter when the "FIFO" is empty. The format of this "idle" word is shown in Table 19.

Zero <31:28>	FCode <27:24>	Z 23	Col <22:21>	Dil <20:18>	Channel <17:12>	Amplitude <11:0>

Table 19. Format of 32-bit sent by column controller.

The 32-bit word consists of an 18-bit digital value representing the amplitude and channel number of the 3GAS card, followed by the identification number for the Dilogic chip or 5DIL card. Bit 23 should be zero. In addition, the FCode 0xA for analogue data or 0xF for reading the threshold memory is transmitted. The column controller also sends the words StartOfData and EndOfData.

3.2.2 Segment Interface Card

The segment interface card consists of five vertical card slots: one 0.8 mm 200-pin HSEC8-1100-01-S-DV-A high-speed vertical card connector for interfacing the FPGA-based column controller card with four 5145154-4 120-way 2-row vertical card connectors, each accommodating a DILOGIC card. Two 10-pin connectors provide the necessary control signals for the pass-plex clock (CLK), track & hold (T/H) and clear (CLR). In addition, the segment has a specially designed JTAG interface for configuring up to four FPGA boards in series via header pins and Ethernet blasters, as shown in Figure 59. FPGA configuration and programming is performed at a stable JTAG clock frequency of 1 MHz, which represents a tenfold time savings compared to previous CPV electronics. The contribution to such a reduction in programming time is related to the use of the SN74LVTH241PWR integrated buffer circuit, which provides stable JTAG digital signals for a correctly assembled chain of up to four segments.

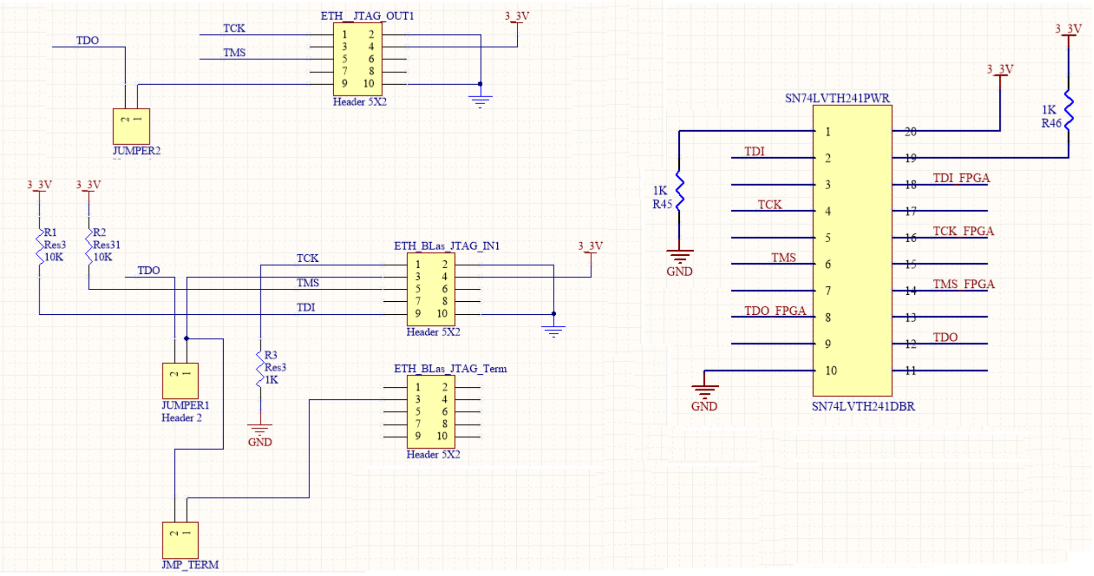


Figure 59: JTAG interface for configuring and programming up to four daisy chained FPGA column controller cards.

The first card is the one to which the Ethernet blaster cable is connected. One then needs to connect in daisy chain the 10-wire ribbon cables as shown in Figure 60 and described below,

- ETH_JTAG_OUT1 on board 1 withETH_Blas_JTAG_IN1 on board 2;
- ETH_JTAG_OUT1 on board 2 with ETH_Blas_JTAG_IN1 on board 3;
- ETH_JTAG_OUT1 on board 3 with ETH_Blas_JTAG_IN1 on board 4;
- ETH_Blas_JTAG_Term on board 4 with ETH_Blas_JTAG_Term on board 1.

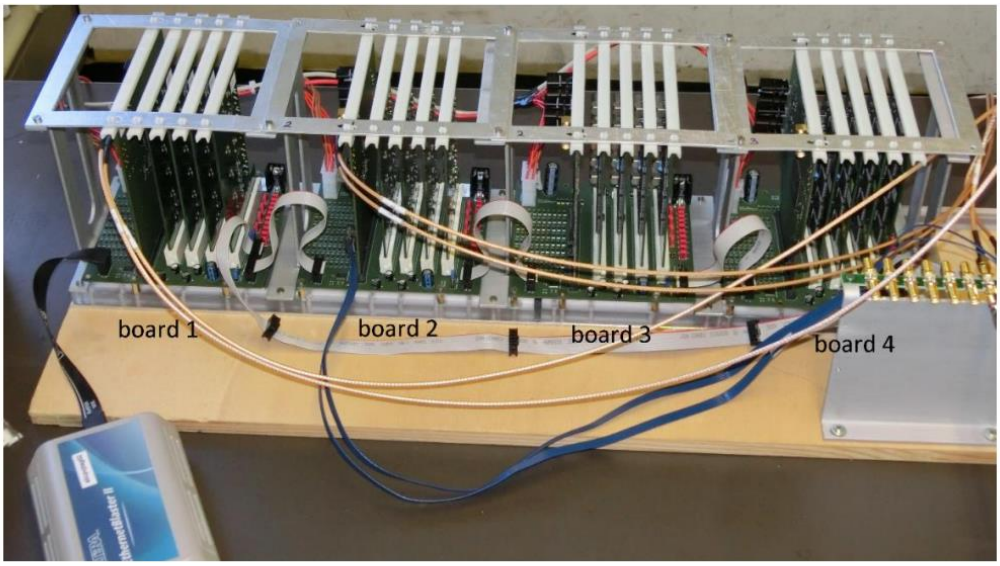


Figure 60: Four segment daisy Chained connection.

The first pin for each connector is marked on the segment board. Further, the following jumpers should be included and checked that are properly connected.

- Board 1 Jumper 2,
- JMP_term Board 2,
- Jumper 2 Board 3,
- Jumper 2 Board 4,
- Jumper 1, and JMP_term.

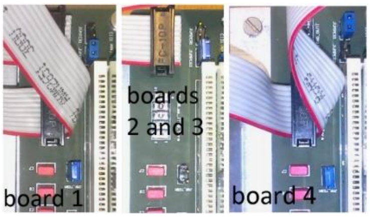


Figure 61: Setting jumpers on segment board for programming the four daisy chained FPGA cards.

In addition, one hundred TXS0102 [48] non-inverting bidirectional two-bit level converters are used to establish a compatible digital voltage level conversion from 2.5V to 5V between all DILOGIC chips and the FPGA controller board at a maximum clock frequency of 20 MHz. As shown in Figure 62, each segment card allows parallel readout of four DIL5 cards or 960 analogue GASSIPLEX channels via four independent 18-bit data buses and control lines. The segment interface card thus contributes to a fourfold reduction in the processing time of all four DILOGIC cards compared to the previous CPV electronics.

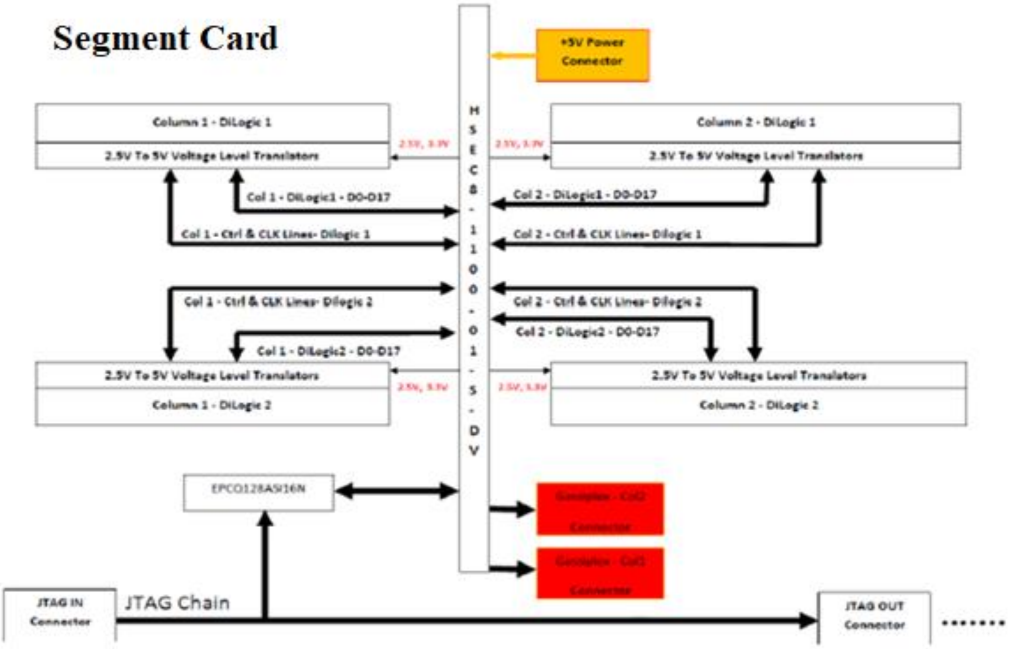

Segment Card
+5V Power Connector
HSEC8-1100-01-S-DV
Column 1 - DiLogic 1
2.5V To 5V Voltage Level Translators
2.5V, 3.3V
2.5V, 3.3V
Column 2 - DiLogic 1
2.5V To 5V Voltage Level Translators
Col 1 - DiLogic1 - D0-D17
Col 1 - Ctrl & CLK Lines- Dilogic 1
Col 2 - DiLogic1 - D0-D17
Col 2 - Ctrl & CLK Lines- Dilogic 1
Col 1 - Ctrl & CLK Lines- Dilogic 2
Col 1 - DiLogic2 - D0-D17
Col 2 - Ctrl & CLK Lines- Dilogic 2
Col 2 - DiLogic2 - D0-D17
2.5V To 5V Voltage Level Translators
Column 1 - DiLogic 2
2.5V, 3.3V
2.5V, 3.3V
2.5V To 5V Voltage Level Translators
Column 2 - DiLogic 2
EPCO128ASI16N
Gassiplex - Col2 Connector
Gassiplex - Col1 Connector
JTAG IN Connector
JTAG Chain
JTAG OUT Connector
•••••••

Figure 62: System Block Diagram for Segment Card.

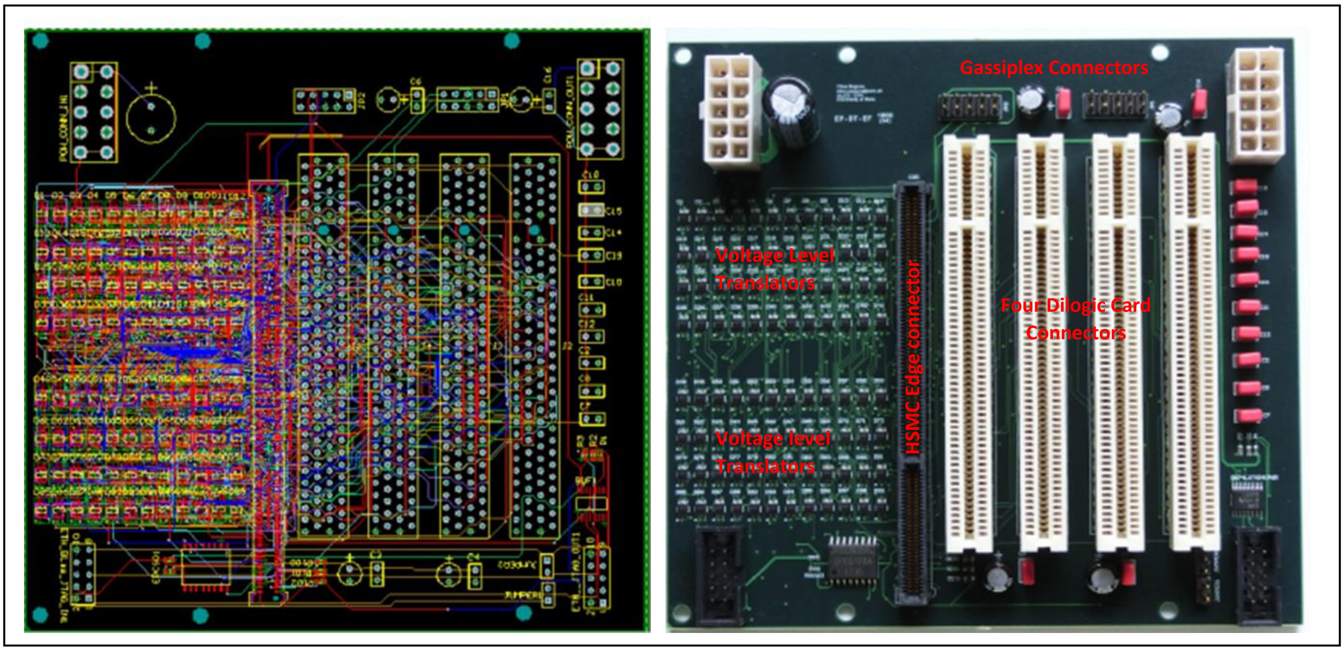
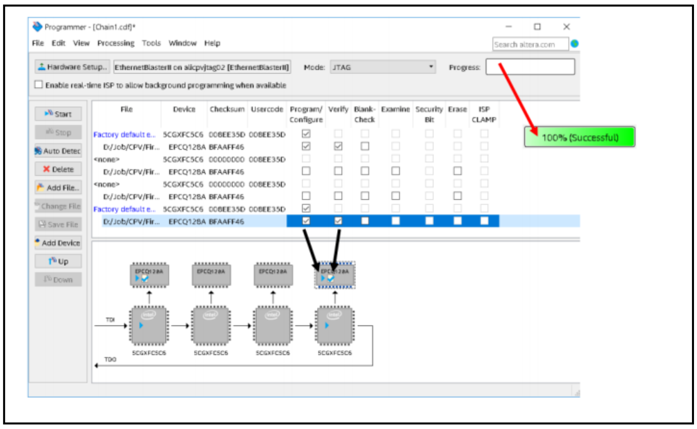

Gassiplex Connectors
Voltage Level Translators
Voltage level Translators
HSMC Edge connector
Four Dilogic Card Connectors

Figure 63: PCB Layout and Manufacturing - Segment Card.

The 200-pin 0.8 mm HSEC8-1100-01- XX-DV -A Edge Card connector is the interface between the FPGA-based Column Controller card and the four 5DIL cards. Without loading any firmware, the total power consumption per segment card is 1.25 A. When the system is powered up, a segment and column controller card require only an estimated 0.22 A of current. Once the FPGA firmware is loaded, the current increases to about 0.44 A. The FPGA-based column controller card is programmed via the JTAG interface using the segment interface card. Each segment contains eight column controllers connected in active serial configuration.

As shown, programming of four (half-segment) FPGA boards is successfully accomplished via Quartus.



Programmer - [Chain1.cdf]*

File Edit View Processing Tools Window Help

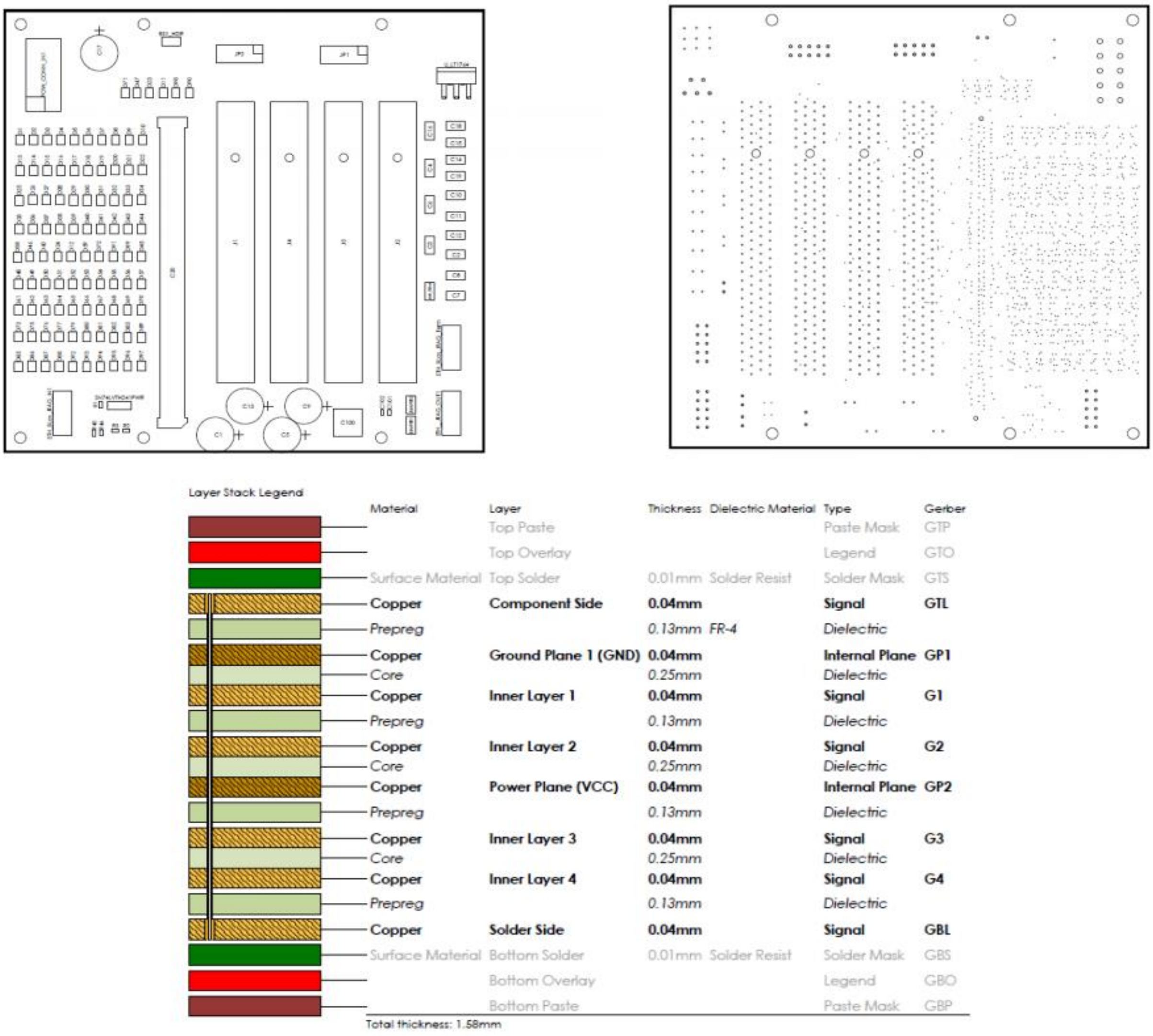
Hardware Setup... EthernetBlasterII on alicpvjtag02 [EthernetBlasterII] Mode: JTAG Progress:

□ Enable real-time ISP to allow background programming when available

File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank-Check	Examine	Security Bit	Erase	ISP CLAMP
Factory default e...	5CGXFC5C6	00BEE35D	00BEE35D	☑	□	□	□	□	□	□
D:/Job/CPV/Fir...	EPCQ128A	BFAAFF46		☑	☑	□	□	□	□	□
<none>	5CGXFC5C6	00000000	00BEE35D	□	□	□	□	□	□	□
D:/Job/CPV/Fir...	EPCQ128A	BFAAFF46		□	□	□	□	□	□	□
<none>	5CGXFC5C6	00000000	00BEE35D	□	□	□	□	□	□	□
D:/Job/CPV/Fir...	EPCQ128A	BFAAFF46		□	□	□	□	□	□	□
Factory default e...	5CGXFC5C6	00BEE35D	00BEE35D	☑	□	□	□	□	□	□
D:/Job/CPV/Fir...	EPCQ128A	BFAAFF46		☑	☑	□	□	□	□	□

100% (Successful)

Figure 64: Successful programming of four Cyclone V FPGA CC cards, through segment board in active serial configuration.



Layer Stack Legend

	Material	Layer	Thickness	Dielectric Material	Type	Gerber
🟥		Top Paste			Paste Mask	GTP
🟥		Top Overlay			Legend	GTO
🟩	Surface Material	Top Solder	0.01mm	Solder Resist	Solder Mask	GTS
▨	Copper	Component Side	0.04mm		Signal	GTL
	Prepreg		0.13mm	FR-4	Dielectric	
▨	Copper	Ground Plane 1 (GND)	0.04mm		Internal Plane	GP1
	Core		0.25mm		Dielectric	
▨	Copper	Inner Layer 1	0.04mm		Signal	G1
	Prepreg		0.13mm		Dielectric	
▨	Copper	Inner Layer 2	0.04mm		Signal	G2
	Core		0.25mm		Dielectric	
▨	Copper	Power Plane (VCC)	0.04mm		Internal Plane	GP2
	Prepreg		0.13mm		Dielectric	
▨	Copper	Inner Layer 3	0.04mm		Signal	G3
	Core		0.25mm		Dielectric	
▨	Copper	Inner Layer 4	0.04mm		Signal	G4
	Prepreg		0.13mm		Dielectric	
▨	Copper	Solder Side	0.04mm		Signal	GBL
🟩	Surface Material	Bottom Solder	0.01mm	Solder Resist	Solder Mask	GBS
🟥		Bottom Overlay			Legend	GBO
🟥		Bottom Paste			Paste Mask	GBP

Total thickness: 1.58mm

Figure 65: 8-layer PCB Board Stacking for Segment card.

3.2.2 ITS Readout Unit

An alternative option for the RCB card is to use the ITS Readout Unit (RU) [49][50], which has already been manufactured and tested. ITS RU [51] can serve as a readout module for all FPGA Column Controllers boards. The development of the hardware and firmware interfaces for ITS RU was done in collaboration with the Russian institute IHEP. This section gives an overview of the main ITS RU hardware modules including firmware that will be used to control CPV FEE RO. The transmission of control commands from ITS RU to the CC board is done via the equipped LVDS FireFly ports, while the transmission of event data from CC to ITS-RU is done via 3.125 Gbit/s transceiver links. The interface of a ITS RU can read up to 28 CC, while CPV has only 24 CC. Additional requirements for the integration of ITS RU with CPV FEE is that ITS RU should include the following changes:

- LM input on LVDS,
- BUSY LVDS output to CTP,
- Cooling system for ITS RU because of the 1.5 kg cold plate,
- Update of VHDL firmware.

The ITS Readout Units (RU) will replace RCB in CPV electronics. ITS RU does not support SIU and DDL2 but uses CRU instead of DDL2. Only the Common Readout Unit (CRU) will be the main interface with the ITS RU, which in turn manages the control signals, trigger, formatting, and data transfer from FEE. The Gigabit Transceiver Optical Link (GBT) consists of a GBTx Serializer/Deserializer ASIC, a GBT-SCA Slow Control ASIC and a VTRx/VTTx Optical Transceiver/Transmitter Module. CRU is an optical GBT connected to the ITS FEE, where one CRU can control more than one RU. The design of ITS RU consists of a SRAM FPGA, which is also used to configure the GBTx via I2C. CRU communicates with O^2 FLP via PCI express. Event and control data is transmitted from ITS RU to CRU at a rate of 24 bits per 40 MHz clock period (parallel GBT data frequency). The Single Word Transaction Protocol (SWT) is used to implement communication via GBTx.

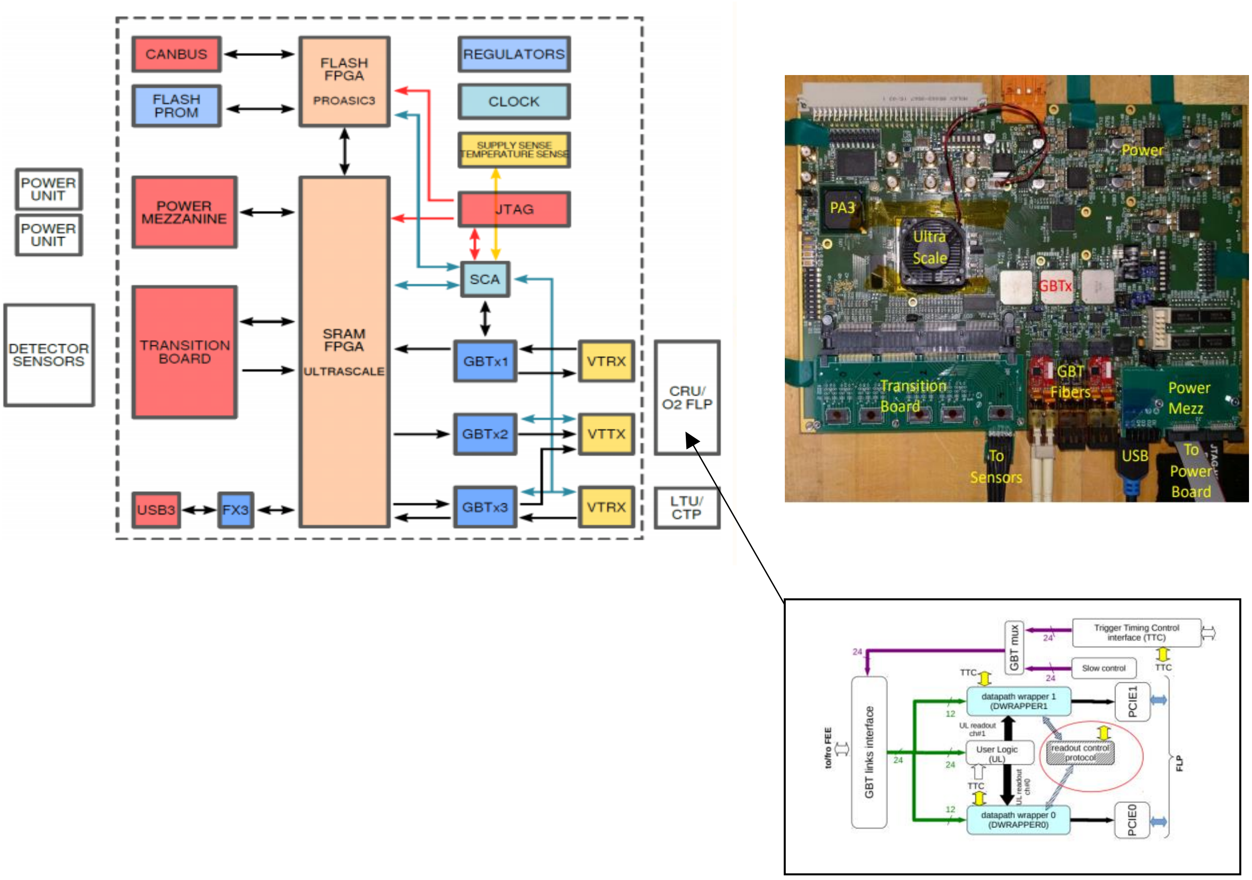


Figure 66: Block diagram and Printed Circuit Board for ITS RU and CRU.

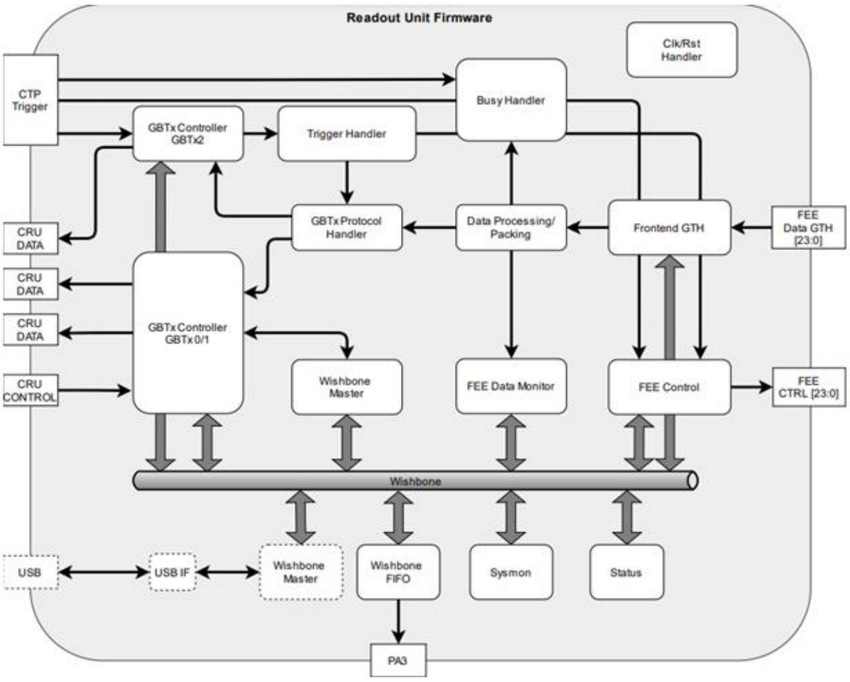


Figure 67: Firmware organization of ITS Readout Unit.

The Command format for both the uplink and downlink packets sent to the GBTx module is shown below.

```
DV  79 76 75       64 63                           32 31                                    0

0   SSSS|XXXXXXXXXXXX|XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX|CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

Data Valid (DV): 0
Bit [79:76] SWT identifier (0x3)
Bit [75:64] Don't care
Bit [63:32] Don't care
Bit [31:0] Control
```

As shown in Figure 67, the internal Wishbone master connected to the GBTx0 control port uses an address width of 15 bits and a data width of 16 bits. The address is divided into a module part and an address part. The module part selects a Wishbone slave connected to the bus; the address part accesses an address of the selected Wishbone slave. To send requests to the Wishbone bus, the communication word to the GBTx is defined as follows:

```
Command format (control only) sent to the GBTx module:

31     24 23     16 15      8 7       0
TMMMMMMM|AAAAAAAA|DDDDDDDD|DDDDDDDD

Bit 31       Access type (1... Write, 0... Read)
Bit [30:24] Module
Bit [23:16] SubAddress
Bit [15:0] Data (For read access, this is Don't care)
```

Each word sent to the GBTx is processed in order by the Wishbone Master. A write command is executed without acknowledgement. For read commands, the result will be sent back with the following structure:

```
Result format received by the GBTx module:

31     24 23     16 15      8 7       0
EMMMMMMM|AAAAAAAA|DDDDDDDD|DDDDDDDD

Bit [31]      Read Error Flag (active high)
Bit [30:24]   Module of data read
Bit [23:16]   Address of data read
Bit [15:0]    Data read
```

The read transaction has a bit that indicates a read error: This bit is set every time the read command on the Wishbone master returns an error. This can be caused by a register not being readable or not existing.

3.5 Verification, Validation and Testing

The design of FEE presented in this section has been tested to verify the expected functionality once it has been produced. This section presents the techniques, simulation and measurement results associated with the testing for the newly developed CPV FEE RO electronics. The first part of this section describes the verification testing of the implemented VHDL digital design using the software simulations used. This section then continues with the verification tests for the PCB hardware. The focus is then on the validation and testing of the hardware device, with emphasis on the tests that impact the digital design. Other tests involving ITS RU and CRU were also performed but are not covered here because they do not directly impact the digital design developed in this work.

3.5.1 Functional Verification

Functional testing is about verifying the operation and functionality of the system according to the intended requirements. This is critical because it can impact cost and schedule at a later stage of production. The main reason for this is that once the PCBs and hardware are manufactured, it will be very difficult to troubleshoot and solve problems. Once a VHDL module is designed in VHDL RTL, its correct functionality must be verified. The designed module is called Device Under Test (DUT). A testbench is used and implemented according to the DUT's interfaces to provide the required stimuli at the designated times and verify the DUT's response. Such functional verification tests help verify the entire design operation, reducing the risk of problems occurring after manufacturing. Modelsim software was used to simulate various testbenches. VHDL uses several special instructions such as assert, severity, and report to detect illegal transactions and signalling in the code. When an assert statement is used and a condition is not met, an error message is displayed. Such assertion violation can be indicated with the report clause. If the test condition is true, no message is displayed. VHDL uses different severity levels to indicate the degree of violation. These severity levels are hint, warning, error, and failure. These special instructions are used in testbenches to provide the necessary input combinations to test a VHDL model or design under test. Fully testing and verifying the entire functionality or behaviour of a VHDL module can be very complicated and time-consuming due to the number of top-level inputs available, increasing simulation time or the number of VHDL instructions required. Therefore, it is preferable to test the functionality of each module independently.

However, testing a module independently may not capture all the functional internal communication details of the system. Therefore, functional verification via a top-level testbench is still important for both post-place-and-route and timings. To reduce the time required for system verification, Intel Quartus provides several real-time debugging tools for the system. An example of such a tool is the SignalTap Embedded Logic Analyser, which is included in the Intel Quartus II software. Using the SignalTap logic analyzer can help developers closely monitor data changes and simplify logic debugging. This can be done at any time through single-step debugging and signal tapping while the system is running on the actual FPGA device and hardware. SignalTap allows functional verification of the design on the hardware. The required debugging logic is integrated into the design by the compiler after a programming file for the FPGA device has been generated.

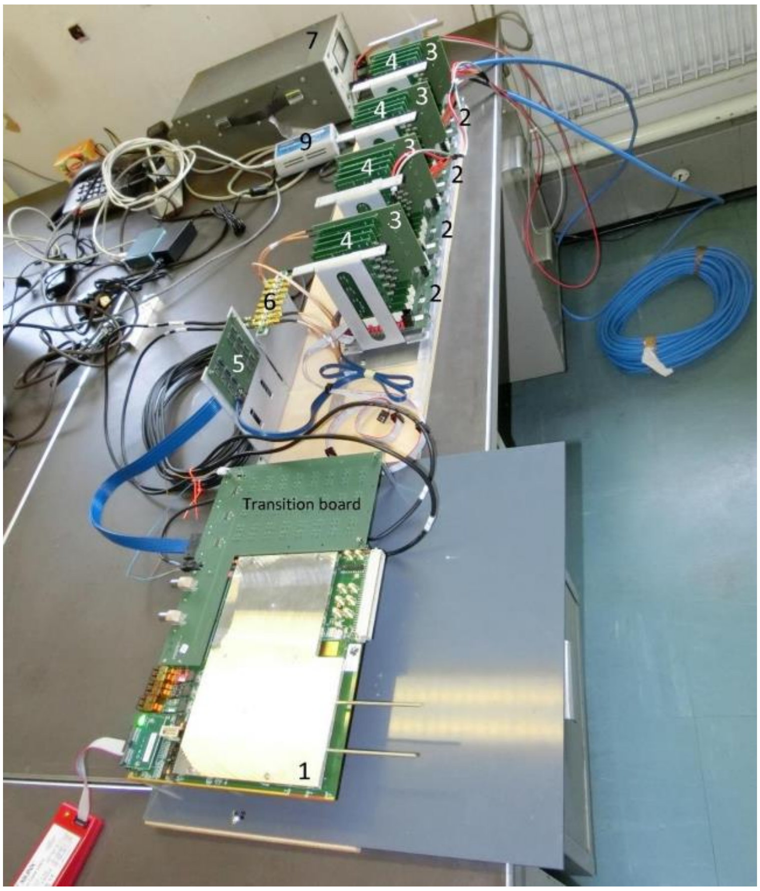


Figure 68: Testbench for the New CPV RO FEE set in CERN lab 595.

The test jig shown in Figure 68 was set up in CERN lab 595-R-027 to test and validate the functionality and integration of the various hardware components of the system. The main components of the installed CPV electronics test rig include the following:

1) Readout Unit with installed Transition Board
2) 4 Segment board,
3) 4 Column controller,
4) 16, 5-Dilogic cards,
5) Patch panel PP-01 for flat cables FireFly and Samtec,
6) Patch panel PP-02 for SMA coaxial cables,
7) Power supply with arrow indicator,
8) Power supply with digital indicator (not shown in the Figure 67),
9) Altera Ethernet Blaster programmer.

The power supply is set to 5.5 V and supplies the segment cards, the column controller and DILOGIC. A grey cable supplies power for 4 segments. When the power source is switched on, a current of about 6 A flows for four fully populated segments (segment board and column controller and four DIL5 cards). A few seconds after loading the firmware from memory to the FPGA, the current increases to 7.5 A. The current may vary depending on the version of firmware downloaded. Without loading the firmware, the system current is 5 A (1.25 A per segment). A second power supply is set to 7V. It supplies power to the readout unit and the transition board. The current value for the firmware loaded from memory is about 2 A. To transfer data from the Column Controller to the Readout Unit, SMA coaxial cables are connected. Pay attention to the correct polarity: TX _P connected to GX_P one cable pair connects the patch panel to the Column Controller, the other to the Transition Board.

Figures 69 and 70 show the prototype of the CPV FEE RO, including FPGA column controller, a segment interface board and two 5DIL boards capable of simultaneously processing 240 analogue channels per DILOGIC board. A 400 Mbps LVDS Firefly cable connection allows control commands to be transmitted from ITS-RU to the CC card. Specially designed electronics enable firmware programming of the FPGA card via USB Blaster.

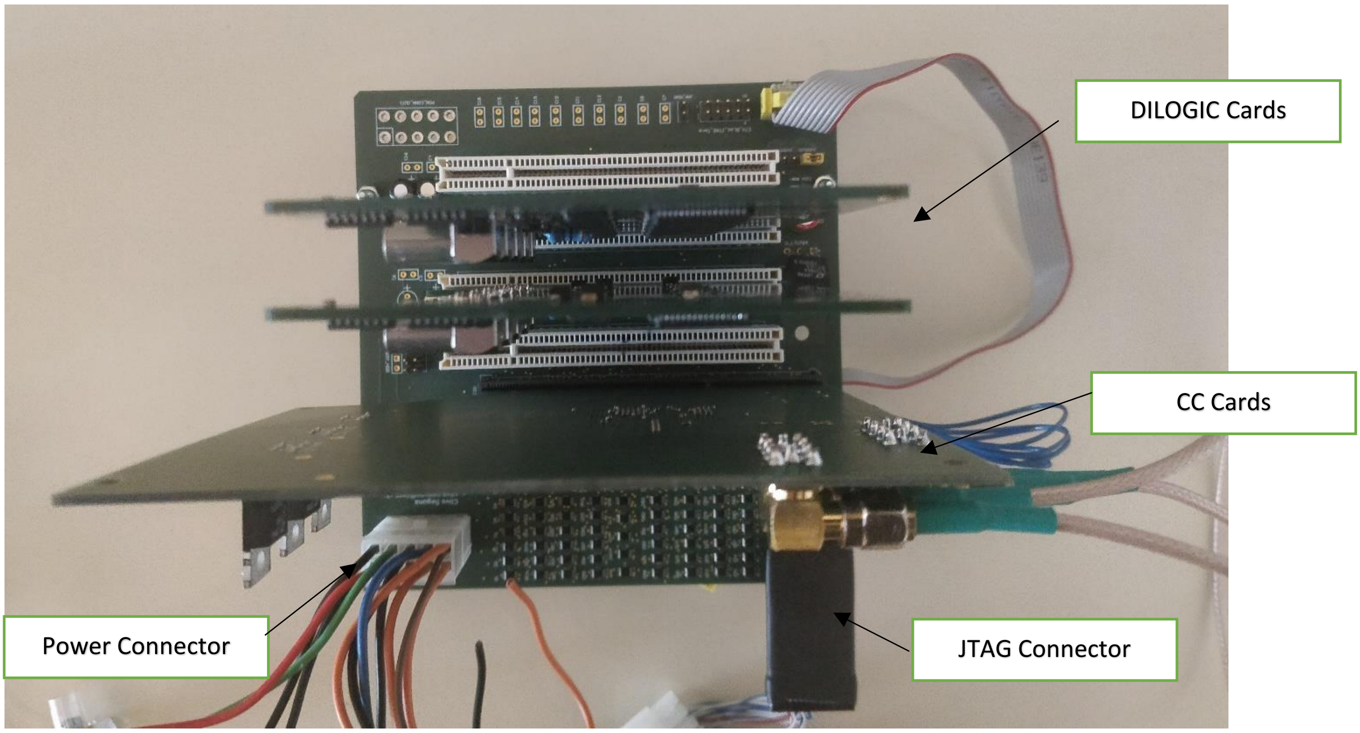


Figure 69: Top View of prototyping jig.

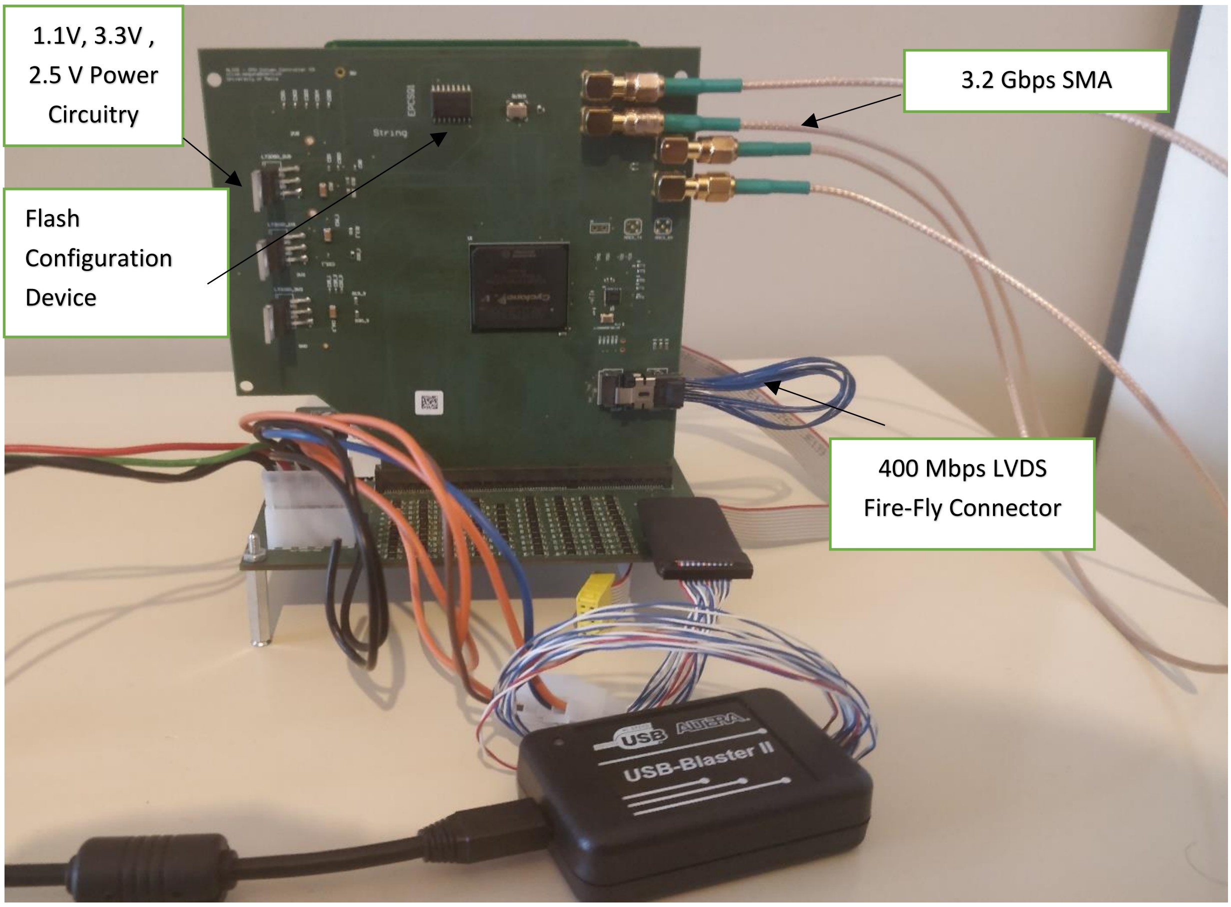


Figure 70: Side View of prototyping jig.

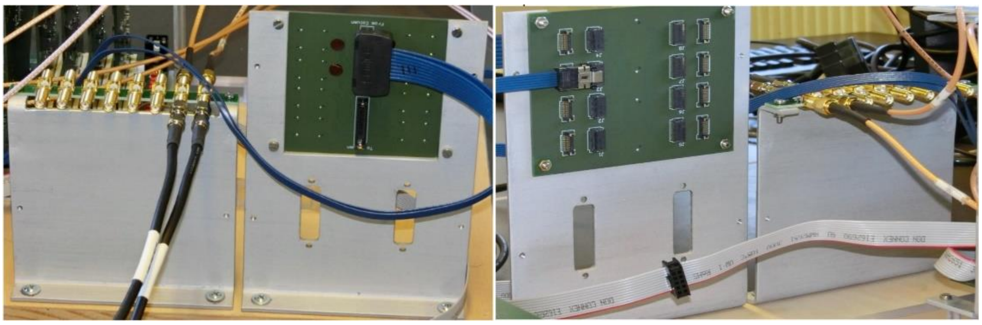


Figure 71: Front and Rear Views of LVDS Patch

The Samtec Firefly shielded blue cable [26] connects the Colum controller with the patch panel. A maximum of eight column controller card can be connected to each patch panel. A description of the verification and testing results obtained through simulation and on hardware is given in the following section. On analysing the various results, redundant processes have been discovered, and firmware was then either updated or removed accordingly.

3.5.1 Column Controller System Hardware Architecture

The following Figure 72 illustrates the top view of the main column controller hardware architecture implemented in the Cyclone V FPGA based electronics. The main controller hardware consists of the following entities:

- **analog_read:** The purpose of this module is to generate the necessary control (CLR _G, T_H, TRG_N) and clock signals (CLK _G, CLKD, CLK _ADC, CLR _D) required for the acquisition of the analogue signal through the circuit 3 GAS and the DILOGIC cards.
- **dilogic_ctrl_top:** Used to read and write pedestal thresholds and ADC digital codeword DILOGIC memory. It also allows simultaneous processing of two 5-DILOGIC cards.
- **sys_ctrl:** this module generates the required timings to control and sequentially activate the various FPGA fabrics based on the decoded received LVDS commands. It also performs clock domain crossing of various control signals from 50M Hz to 10 MHz and provides the necessary control logic to fill the internal column controller RAM with pedestals.
- **lvds_wrapper:** Enables conversion and data alignment of the LVDS input receive bit stream to 4-bit words from the unit ITS RO.

- **data_packer:** Multiplexer for compiling and sending data from 4 DILOGIC cards to the high-speed transceiver. All DILOGIC data is transferred to a FIFO memory, then multiplexed and assembled before being transferred serially to ITS RO. Reading from FIFO is done using a round-robin scheduling algorithm.
- **XCVR_wrapper:** Enables conversion of 32-bit words to a serial bitstream. It includes the implementation of a reset and reconfiguration controller so to set up the required data rate and interface width.

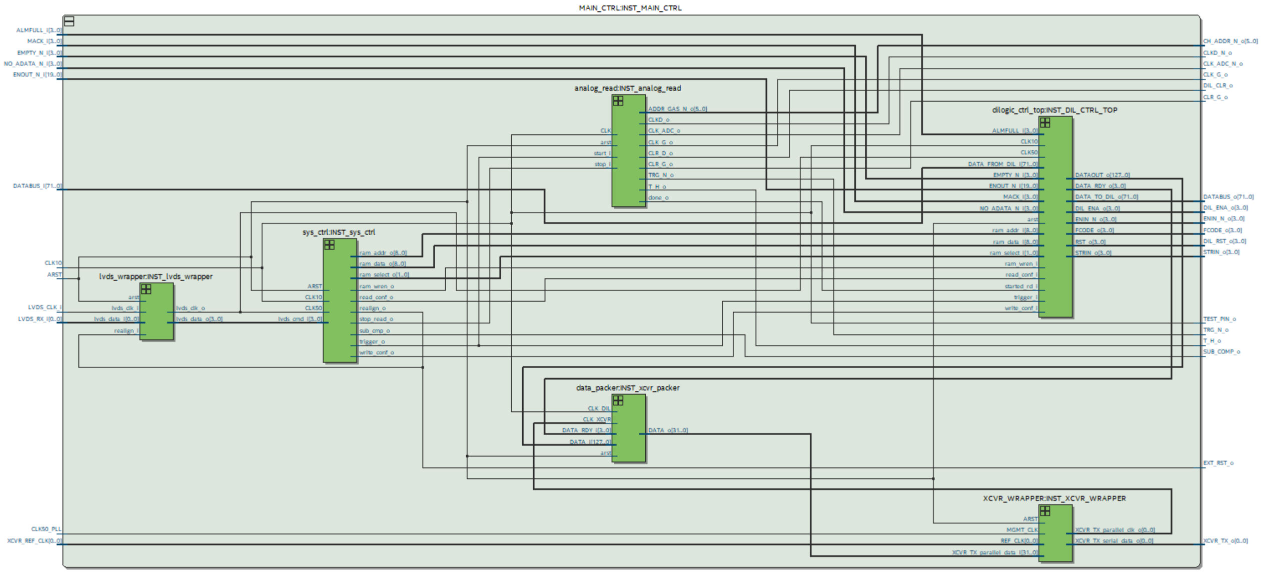


Figure 72: Overall system architecture for the CPV front-end column controller FPGA-based system.

3.5.1.2 Functional Verification of Front-End Operations

Using custom designed testbenches and Modelsim, simulations were done to verify the column controller firmware functionality under the following operating modes:

- Front-end Operations
  - Data Acquisition mode,
  - Test Mode operation,

- Back-End Operations
  - Read and write configurations,
  - Analogue Readout,
  - Pattern Readout.

3.5.1.3 Simulation results for Front-end Operations

Before recording the event data, the DILOGIC processor must fill the pedestal threshold memory with the operating thresholds for each analogue channel. Also, the CC must activate the CLR _N and RST to initialise the DILOGIC processor prior to data acquisition. Each event starts with a pulse on the TRIGN pin and several clock cycles on the C1_CLKD_N and CLK _ADC_N pins, depending on the number of channels indicated by the binary value "10" on the C1_GX pin. An additional clock cycle is necessary to store the end event word and to switch off the reading of an event. C1D1_ADG < 5:0 > and C1D2_ADG < 5:0 > are 6-bit counter registers that contain the memory and analogue channel addresses for DILOGIC number one and two of the first column, respectively. Two other similar registers are used for DILOGIC numbers three and four of the same controller cards in the column. Figure 73 shows the timings for the test mode, which has a similar sequence to the data acquisition mode, but instead of enabling the amplitude and channel address on the input pins, they are entered via the digital I/O bus, namely the channel address at C1_ DATA _BUS0/1 <17:12> and the channel amplitude at C1_ DATA _BUS0/1 <11:0>.

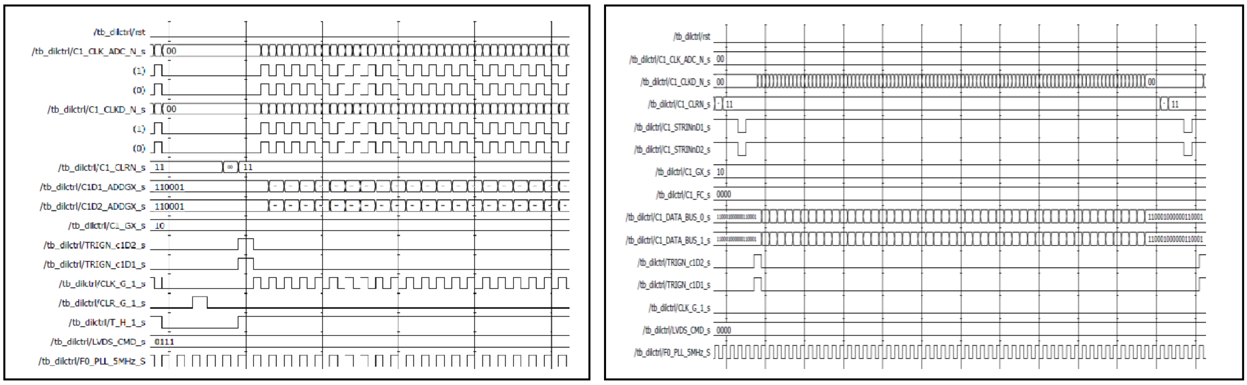
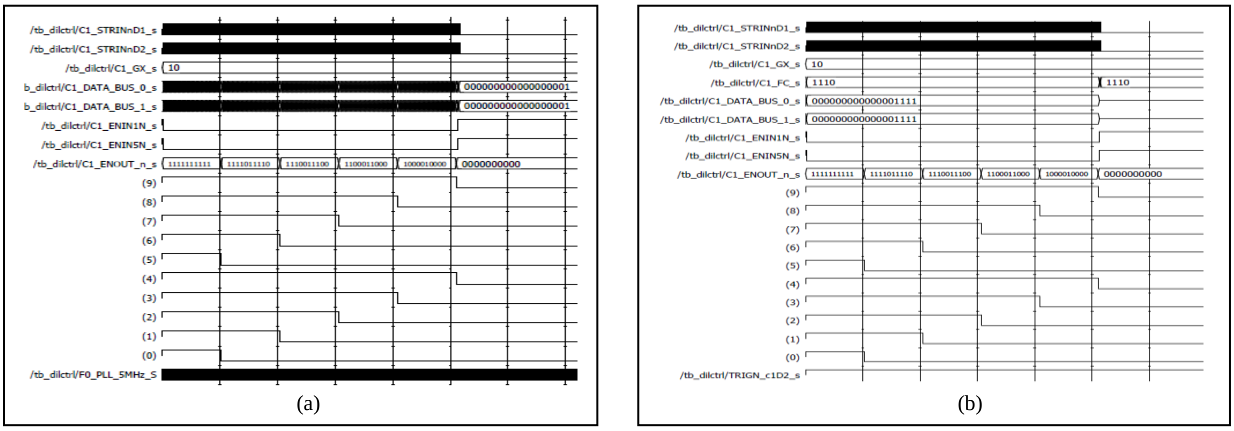


Figure 73: Timing Diagram (a) Data Acquisition and (b) Test Operations.

The DILOGIC threshold and offset memory can be read using the function code “1111”, where pins ENIN1N, ENIN5N are set to low and STRINnD1, STRINnD2 strobe cycles are applied. The data will appear on the data bus after the falling edge of strobe and should be stable until the rising edge of STRINnD1, and STRINnD2 pins. The threshold and offset memory of DILOGIC chip can be loaded using the function code “1110” (C1_FC_s), where the ENIN1N and ENIN5N pins are set to low and STRINnD1, STRINnD2 strobe cycles are applied.



(a)

(b)

Figure 74: Timing Diagram for DILOGIC (a) Read and (b) Write configuration.

At the end of the operation, a reset of the daisy chain must be performed. Once the chain is reset, the ENOUT_n_s pin is set low and ENIN1N_s is set high. As can be seen in Figure 74, the ENOUT_n_s signals on each DILOGIC board are activated simultaneously for the respective ten DILOGIC chips at the end of the data transfer, thus enabling them. A PLL_5MHz clock signal was used in these simulations. To validate the read and write configuration of the threshold memory, the Quartus SignalTap and Test jig platform shown in Figure 68 was used. An 18-bit dummy word x "2AAAF" representing the pedestal values was written to each DILOGIC chip memory using the "1110" command. The read operation was then performed using the "1111" command. The snapshot shown in Figure 75 shows that the pedestal values were read with the correct values. To confirm the actual reading of the dummy word x "2AAAF", the DILOGIC power supply was turned off and the socket was read again. As can be observed from Figure 76, after turning off the power supply, a value of x "00000" was read, confirming that it was correct.

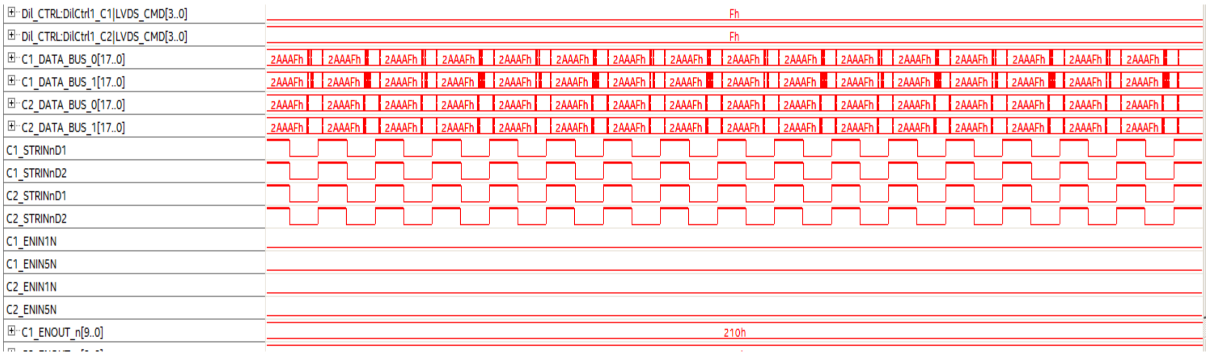


Figure 75: Reading 18-bit pedestal value x “2AAAF”.

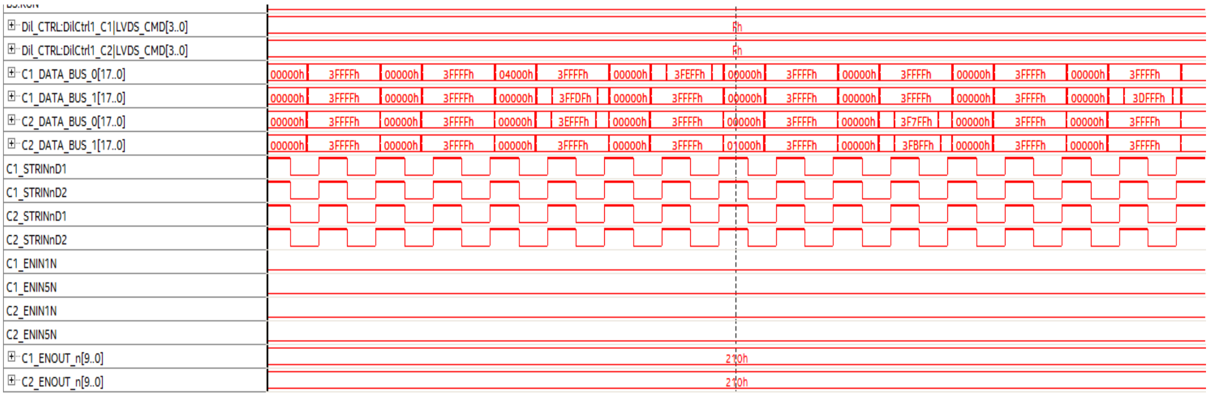


Figure 76: Reading 18-bit pedestal value x “00000”, after powering up the system.

The DILOGIC processor is put in analogue readout mode on receiving the function code “1010” via the LVDS_CMD_s input port, whereas the ENIN1N and ENIN5N pins are set low. Successive STRINnD1, and STRINnD2 cycles will cause all DILOGIC modules in the chain to place their digitised data on the data bus one at a time, starting with the first module in the chain. An enable signal is passed from the ENOUT_N pin to the ENIN_N pin of the next chip when the module has finished transferring the analogue event data to the C1_DATA_BUS <17:0> bus. The MACK_1N and MACK_2N pins indicate the occurrence of the end event word and the end of the analogue readout on that specific DILOGIC chip. The Bitmap memory contains the event profile. A “1” or “0” is in the bitmap memory if the channel amplitude is higher or lower than the threshold memory. To perform a pattern readout of the Bitmap from a specific DILOGIC chip (Fig. 77(b)), the operation code must be set to “1000”, and both ENIN1N, ENIN5N pins must be set low. Once the STRINnD1, and STRINnD2 are set low the patterns will appear on the data bus where each channel status word is 16 bits. Similarly, to the analogue RO sequence, the pattern readout procedure will be terminated when the last DILOGIC module set EnOut_N pin low. The EnOut_N pins are then turned off by issuing a reset daisy chain.

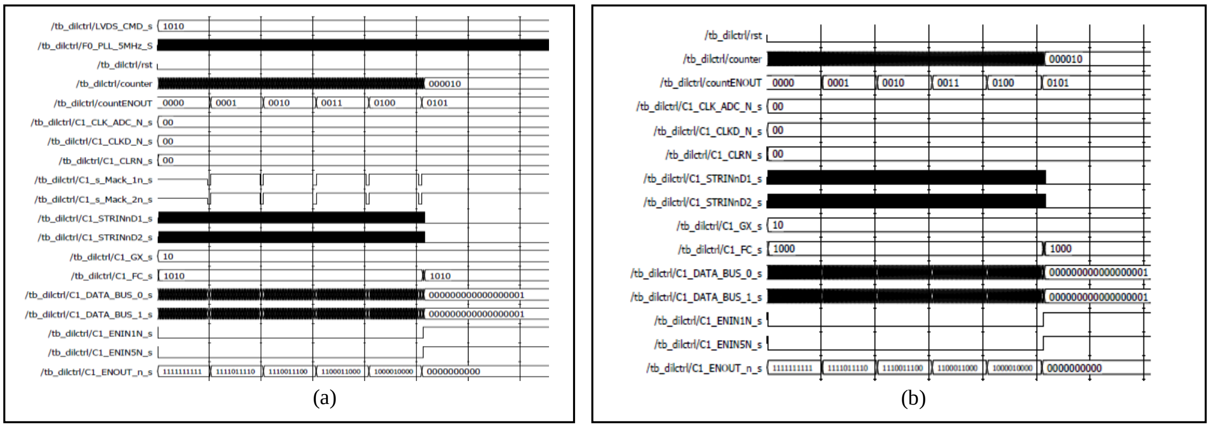


Figure 77: Timing diagrams for (a) Analogue Readout and (b) Pattern Readout.

Figure 78 shows the timing diagram for the CC LVDS receiver where 4-bit data commands were sent from the ITS-RO unit to the FPGA-based column controller board. For verification, these 4-bit commands were generated by an up-counter from the ITS-RO. All 4-bit patterns were received correctly in the correct order on repeated attempts. Alignment and synchronization between the LVDS receiver and ITS-RO unit was accomplished by implementing a bit-slip procedure state machine.

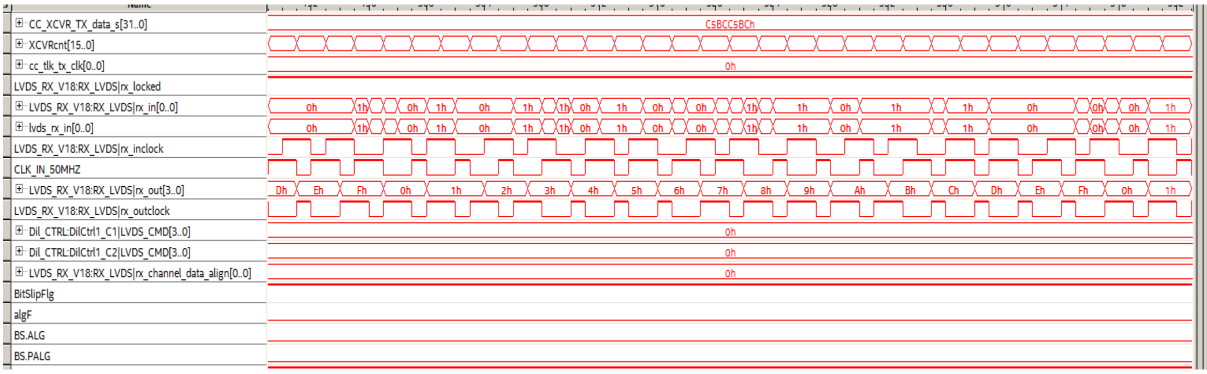

CC_XCVR_TX_data_s[31..0]
C5BCC5BCh
XCVRcnt[15..0]
cc_tlk_tx_clk[0..0]
0h
LVDS_RX_V18:RX_LVDS|rx_locked
LVDS_RX_V18:RX_LVDS|rx_in[0..0]
lvds_rx_in[0..0]
LVDS_RX_V18:RX_LVDS|rx_inclock
CLK_IN_50MHZ
LVDS_RX_V18:RX_LVDS|rx_out[3..0]
Dh Eh Fh 0h 1h 2h 3h 4h 5h 6h 7h 8h 9h Ah Bh Ch Dh Eh Fh 0h 1h
LVDS_RX_V18:RX_LVDS|rx_outclock
Dil_CTRL:DilCtrl1_C1|LVDS_CMD[3..0]
Dil_CTRL:DilCtrl1_C2|LVDS_CMD[3..0]
LVDS_RX_V18:RX_LVDS|rx_channel_data_align[0..0]
BitSlipFlg
algF
BS.ALG
BS.PALG

Figure 78: Data measurement for LVDS receiver at 200 Mbps.

Gigabit receivers (3.125 Gb/s) are AC coupled to OCT and use 8b/10b encoders/decoders, byte ordering, and an automatic synchronisation state machine. To avoid common-mode noise generated by non-identical characteristics (e.g., unequal length, diameter, twist, or material), AC coupling is used for the common-mode signal, while the intra-pair skew can be set via the transceiver's slew-rate programmer. AC-coupling allows the transceivers to operate with large common-mode offsets. The low-speed (200 Mbps) LVDS links are DC coupled with 100 Ω terminators connected across the link pairs and placed as close as possible to the receiver. Since we use DC coupling instead of AC for LVDS, the 8b/10b coding scheme was not necessary. In addition, Intel FPGAs do not support AC coupling on ALTLVDS_RX and ALTLVDS_TX.

Figure 79 illustrates the entire chain of analogue readout waveforms as an example. As soon as the column controller receives the L0 trigger signal over the LVDS link, 3GAS card readout is initiated. The DILOGIC controller is idle while the front-end controller is running. The top level DILOGIC controller outputs FCODE 0xA when the front-end readout is ready, and then waits for all individual controllers to return 'done'. The FCODE 0xD is then generated to reset all 5DIL cards to their initial state (reset daisy chain). When the individual controller receives the FCODE 0xA, it sets the ENIN signal low and emits a burst of STRIN pulses (10 MHz clock) until all five ENOUT of the 5DIL are set low. Then another STRIN pulse is issued to reset the daisy chain and finally release the ENIN signal. A similar procedure is used to perform the read operation, but instead the FCODE 0xF is used, which starts immediately when the command is received from the LVDS link. The same procedure is used to configure the top-level DILOGIC controller module, but FCODE 0xE is used instead. A single controller sends data from the RAM to the respective external DILOGIC chip RAM, which is located on the 5DIL card. In addition, the DILOGIC controller reads the status of the 'Output Enable' signal ENOUT of each card. It could be the case that a DILOGIC chip or card is defective. In such a case, the controller will not receive a logic high signal on ENOUT and the process will hang. To avoid such a situation, a timeout procedure has been implemented. Each DILOGIC chip needs 65 pulses for the read/write configuration or 45 STRIN pulses for the analogue RO. A time counter was implemented that is slightly higher than the clock cycles required to read all five DILOGIC chips per card. The chain is reset if the top-level pulse 'done' is not received before the time expires.

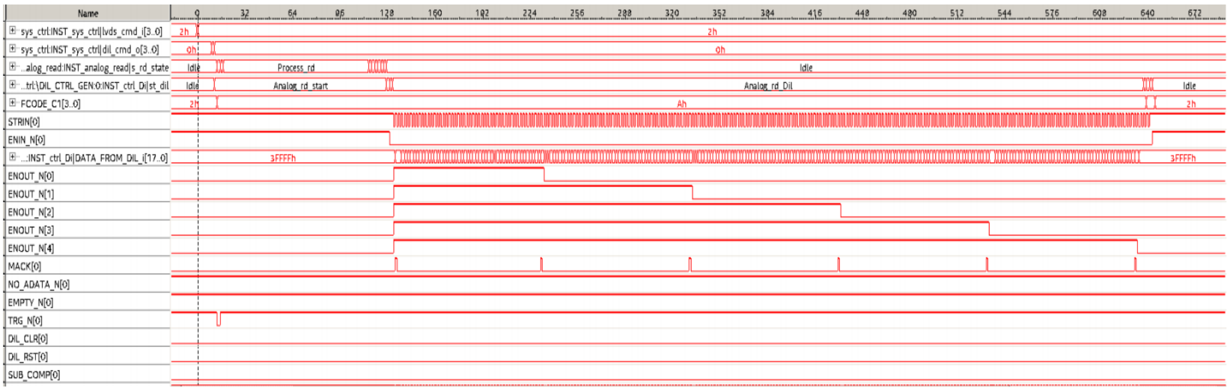


Figure 79: Issue of LVDS trigger to perform a full chain analogue RO.

Each Column Controller card uses a 3.125 Gbps Full Differential Transmitter over which event data is transmitted. A Sub Miniature Version A (SMA) coaxial cable connects the FPGA-based Column Controller cards to ITS-RU at 5 m distance. The performance and quality of such a connection is critical to minimising the bit error rate. Such performance can be analysed using eye diagram measurements. This is to ensure that the receiver model of ITS-RU is able to consistently distinguish between one and zero in the presence of jitter or timing noise. In addition, the eye diagram was used to verify any losses associated with the high-speed interconnects on the PCB, ensuring that the transmitter output complies with the required BER specifications. The eye diagram for a high-speed digital signal is recorded by superimposing 1s, 0s, and transition measurements at a high sampling frequency. In addition, these samples can be used to analyse the noise, root-mean-square (RMS), and peak-to-peak jitter of the signal. The rising and falling edges of an eye diagram are used to measure such jitter deviations. The quality of the digital signal is represented in an eye diagram by the bathtub graph and shows the bit error rate (BER), which depends on the eye closure. An eye diagram is a common indicator of signal quality for high-speed digital transmissions. Connections (3 Gb/s and low speed) must be tested with higher statistics and random patterns (PRBS) to evaluate BER /bathtub curves with realistic cable connections over a long measurement period.

The VHDL implementation of a 16-bit PRBS generator was used to obtain the necessary eye diagrams and jitter measurements to test the physical quality of the high-speed transceivers of the FPGA CC cards. The implementation of this PRBS generator is based on a linear feedback shift register with logical operations XOR and AND, which generates a predefined sequence of 1s and 0s, as shown in Figures 80 and 81.

The VHDL-based PRBS generator was simulated using ModelSim-Altera, as shown in Figure 77. The PRBS generator allows the generation of a random bit pattern, since the eye diagram is a statistical average of many thousands or even millions of samples of a waveform. An oscilloscope with a sampling frequency of 6 GHz was used to capture the full characteristics of the waveform in clock recovery mode to show the overhang of millions of waveforms in the time domain.

The jitter and eye diagram measurements shown in Figures 82, 83 and Table 20 were recorded using a Tektronix 6 GHz real-time oscilloscope with clock recovery enabled. An eye diagram is a common indicator of the quality of signals in high-speed digital transmissions. The data itself was used as a trigger. The test was performed over a 72-hour period with 5 m SMA cables. The measured differential input jitter, PRBS pattern at zero crossing is +/- 0.25 UI or 0.5 UI. Rise and fall times of approximately 100 ps were measured, with a peak jitter value of +/- 20 ps at a data rate of 3 Gbps. These measurement results were compared to the characteristic features of the TLK3102 3 Gbps transceiver currently used in the HMPID detector. All SMA cables were certified using eye diagram measurements in order to select cable pairs of equal length to use in one differential line.

The TLK3102 also exhibits a typical deterministic jitter of 0.6 UI [53], i.e., a peak-to-peak jitter of 40 ps. Therefore, the performance of the developed transceiver interface between CPV Column controller cards and ITS-RO is within the similar limits as the current HMPID gigabit transceiver interface. On the other hand, the measured bit error rate (BER) of Bathtub is 1 x 10^{-12}, which is also an acceptable value according to the digital recommendation standards mentioned in [54].

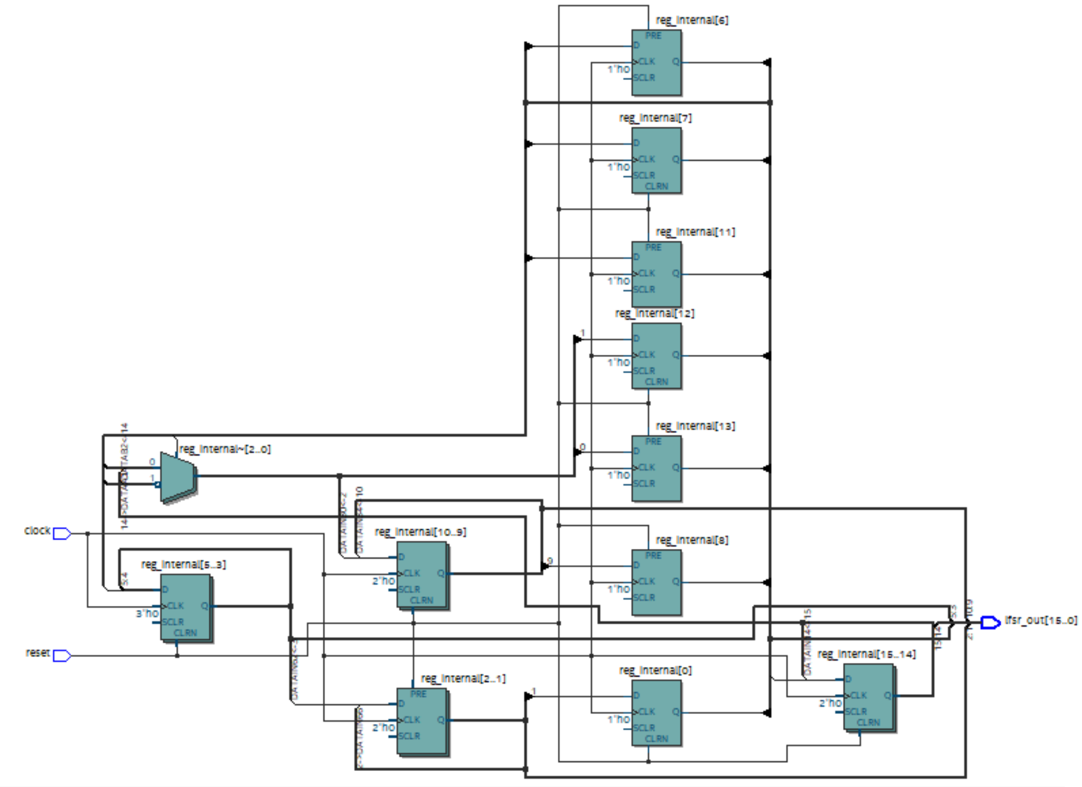


Figure 80: Netlist View of VHDL PRBS generator.

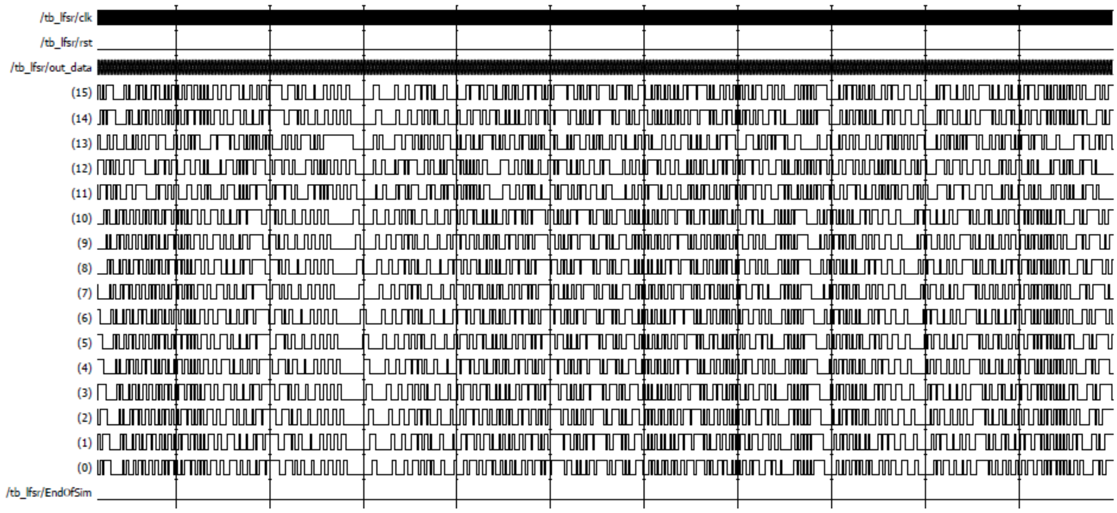
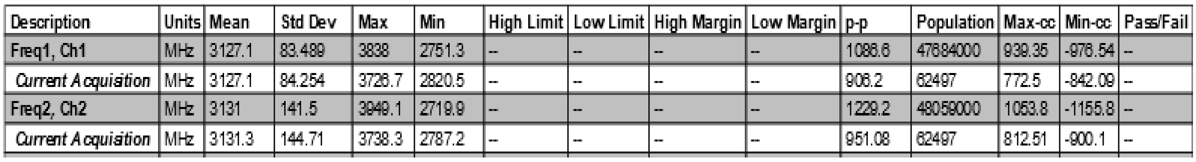


Figure 81: ModelSim-Altera Simulation Results for VHDL PRBS generator.



Description	Units	Mean	Std Dev	Max	Min	High Limit	Low Limit	High Margin	Low Margin	p-p	Population	Max-cc	Min-cc	Pass/Fail
Freq1, Ch1	MHz	3127.1	83.489	3838	2751.3	--	--	--	--	1086.6	47684000	939.35	-976.54	--
Current Acquisition	MHz	3127.1	84.254	3726.7	2820.5	--	--	--	--	906.2	62497	772.5	-842.09	--
Freq2, Ch2	MHz	3131	141.5	3949.1	2719.9	--	--	--	--	1229.2	48059000	1053.8	-1155.8	--
Current Acquisition	MHz	3131.3	144.71	3738.3	2787.2	--	--	--	--	951.08	62497	812.51	-900.1	--

Table 20. Transceiver frequency measurement results.

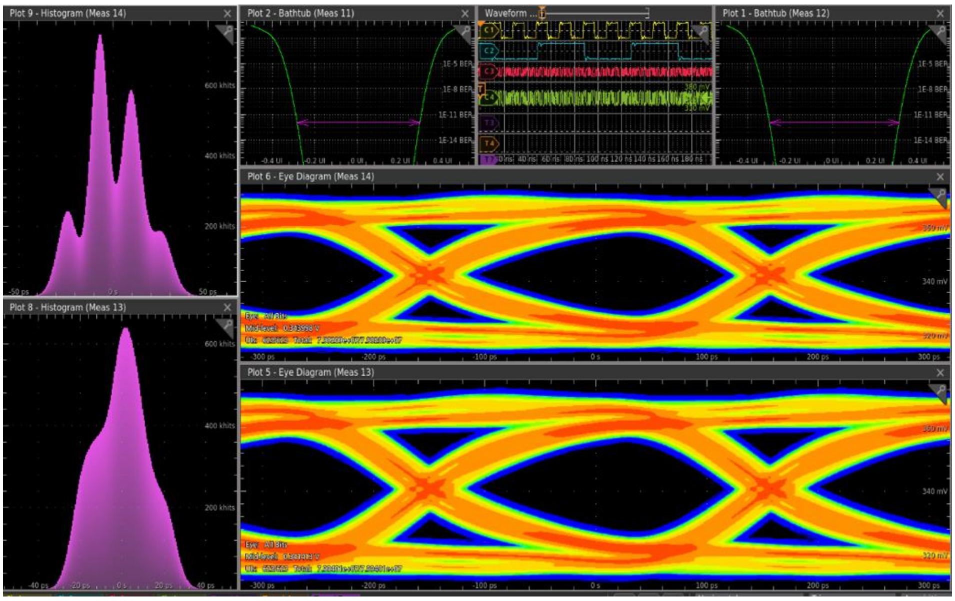


Figure 82 (a): Eye and Bath-tub diagrams for CC card high-speed Gigabit transmitter, obtained after a duration of 72 hours using 5m SMA cables.

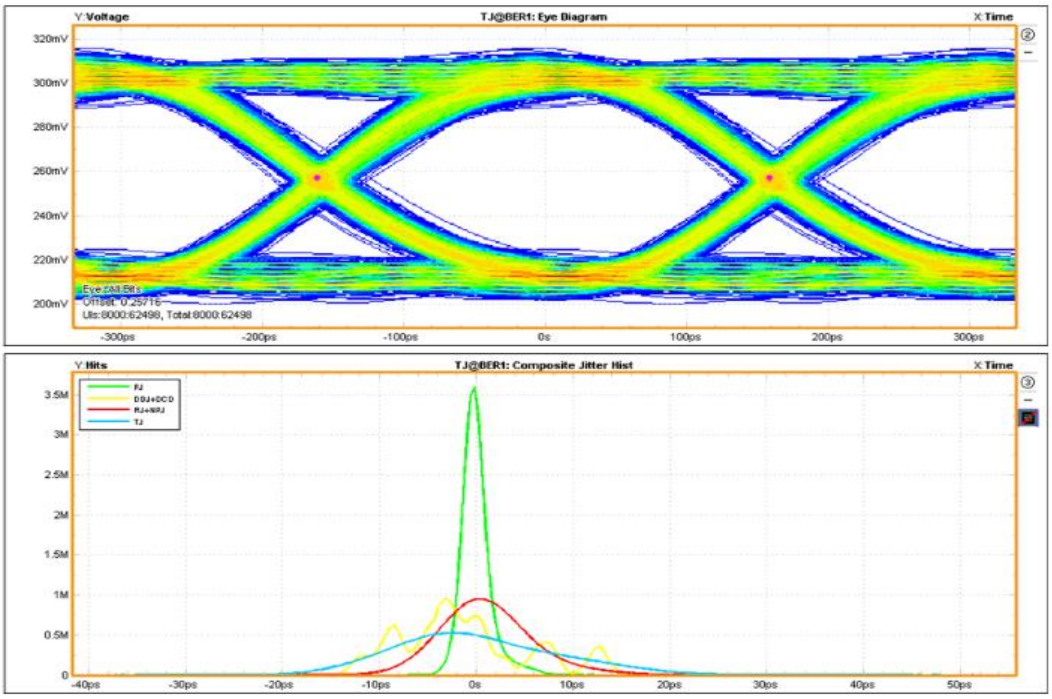


Figure 83: Full-Duplex transceiver eye diagram, composite jitter histogram.

Gigabit receivers (3 Gb/s) are AC coupled to OCT and use 8b/10b encoder/decoder, byte ordering, and an automatic synchronisation state machine. To avoid common-mode noise generated by non-identical characteristics (e.g., unequal length, diameter, twist, or material), AC coupling is used for the common-mode signal, while the skew rate within the pair can be adjusted via the transceiver's slew rate programmer. AC -coupling allows the transceivers to operate with large common-mode offsets. The low-speed (200 Mbps) LVDS links are DC coupled with

100-ohm terminators connected across the link pairs and placed as close as possible to the receiver. Since LVDS DC coupling is used instead of AC coupling, the 8b/10b coding scheme is not required. In addition, Intel FPGAs do not support AC coupling on ALTLVDS_RX and ALTLVDS_TX.

In order to test the behaviour of the FPGA CC board in the presence of magnetic fields up to 2 Tesla, a special test fixture (see Figure 84) was set up. While the magnetic field strength was varied, the quality of the digital signals was monitored with a digital oscilloscope. No distortions were recorded, leading to the conclusion that the implemented FEE is insensitive to changes in the magnetic field.

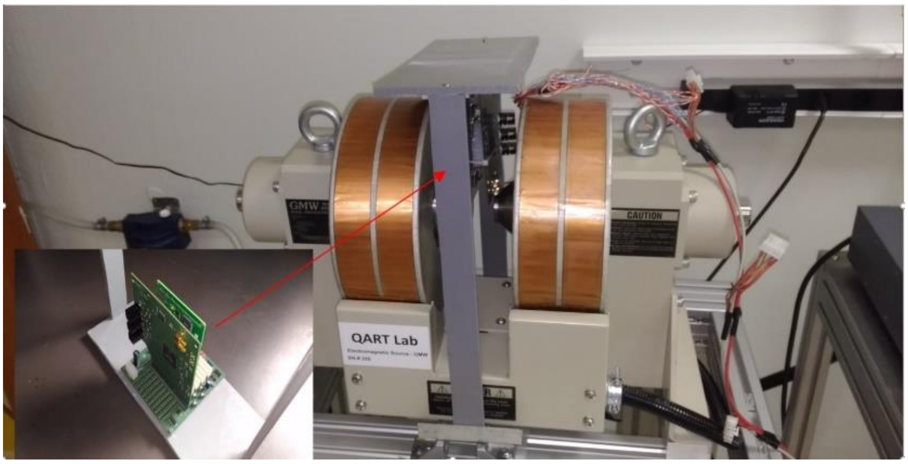
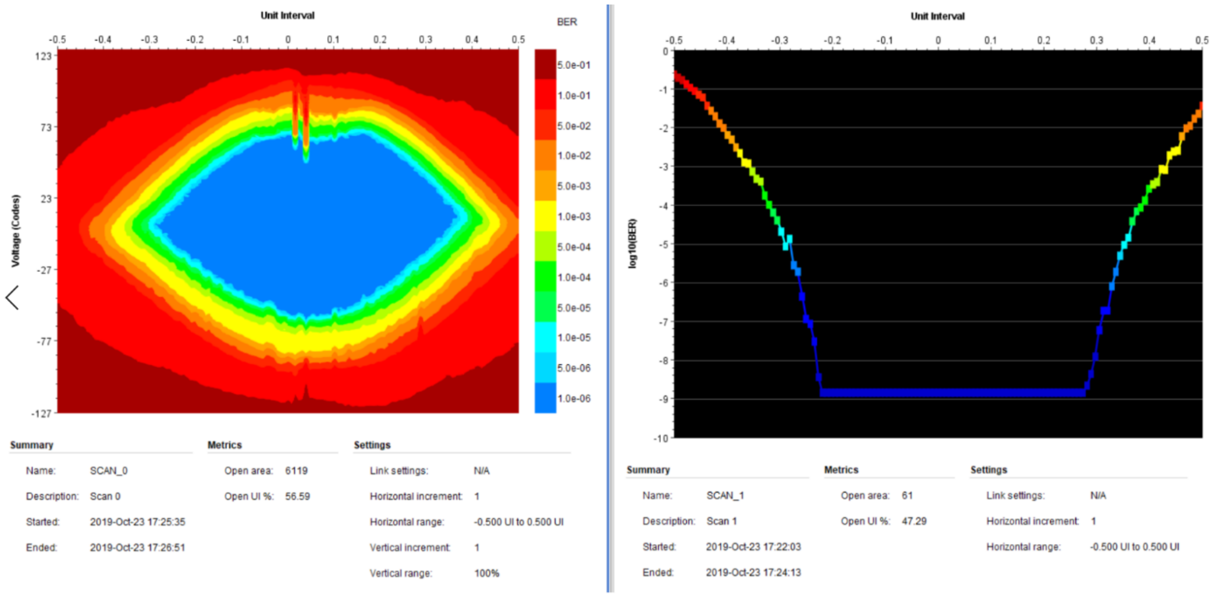


Figure 84: Test rig setup for evaluating the functionality of the FEE operating under magnetic field variations of 0 up to 2 T.



Summary

Name: SCAN_0
Description: Scan 0
Started: 2019-Oct-23 17:25:35
Ended: 2019-Oct-23 17:26:51

Metrics

Open area: 6119
Open UI %: 56.59

Settings

Link settings: N/A
Horizontal increment: 1
Horizontal range: -0.500 UI to 0.500 UI
Vertical increment: 1
Vertical range: 100%

Summary

Name: SCAN_1
Description: Scan 1
Started: 2019-Oct-23 17:22:03
Ended: 2019-Oct-23 17:24:13

Metrics

Open area: 61
Open UI %: 47.29

Settings

Link settings: N/A
Horizontal increment: 1
Horizontal range: -0.500 UI to 0.500 UI

Figure 85: Eye diagram and bathtub from the FPGA ITS-RO, and LVDS patch panel.

The BER for the FPGA ITS-RU, and LVDS patch modules have been measured. As shown in Figure 83, the best case, BER is 1 x 10^{-9}, which is very dependent on cable pair length. Therefore, the cable length has been set to not more than 5 m. Additionally, further modifications had to be done so to use an external clock source instead of the internal PLL FPGA clock source as recommended my manufacturer. Signal Tap has been successfully used to measure the pedestals of all CPV modules, as shown in Figure 86. This trial has been performed so to identify any contact or trace connection issues. Further testing was performed to measure the firmware readout rate CC at ITS-RU with full event size and zero suppression disabled. The readout is stable at 25 kHz and 30 kHz. The measured busy time of the FEE RO system is about 31µs. With empty events and zero suppression enabled with a maximum threshold of (0x1FF), the measured busy time was slightly shorter, about 8 µs. In addition, the readout was stable at a trigger rate of 125 kHz, as shown in Figure 81. As can be observed, a trigger pulse is set at pin TRG_N, whereupon the sequence ENIN-ENOUT from a DILOGIC card and the FCODE pattern are activated. Event data from all four 5DIL cards is then transmitted simultaneously to the ITS RO. The time scale over the waveforms is 100 ns/division.

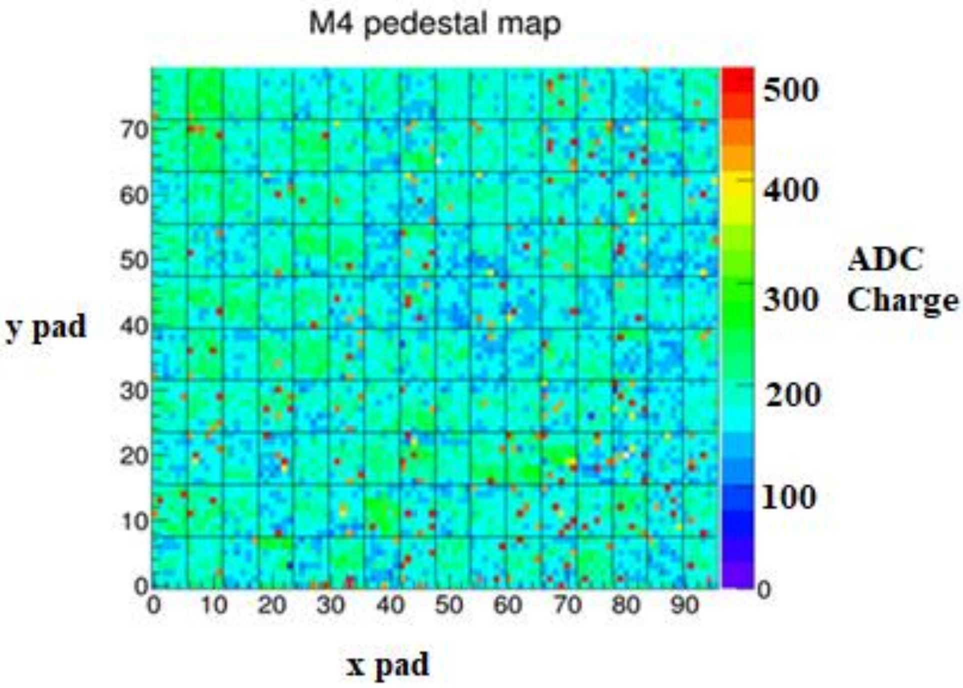
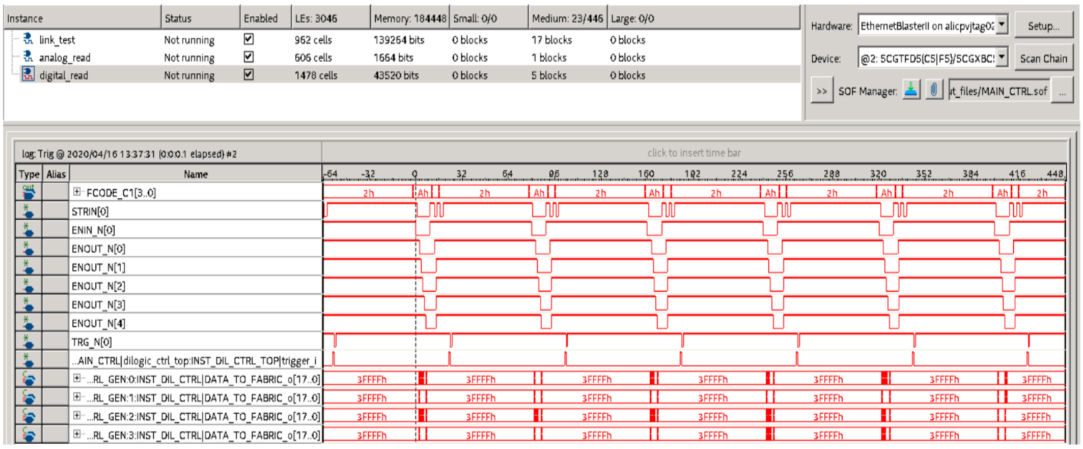


Figure 86: Pedestal measurement for all CPV modules

The gap between a trigger and the start of the Function Code (FCODE) command "1010" is the time needed to read the data from GASSIPLEX to DILOGIC. It is always about 5.8 µs.

Based on the above measurement, a readout rate of 50 kHz (periodic, not random) can be achieved with a maximum occupancy of 27 channels per 3-GASS card. This is more than half of our detector. Also, if we increase the column controller readout rate, the GBT link becomes a bottleneck as it takes data from all 24 CCs. Therefore, no further timing or firmware improvements may be needed for the CC firmware.



Instance	Status	Enabled	LEs: 3046	Memory: 184448	Small: 0/0	Medium: 23/446	Large: 0/0
link_test	Not running	☑	962 cells	139264 bits	0 blocks	17 blocks	0 blocks
analog_read	Not running	☑	606 cells	1664 bits	0 blocks	1 blocks	0 blocks
digital_read	Not running	☑	1478 cells	43520 bits	0 blocks	5 blocks	0 blocks

Hardware: EthernetBlasterII on alicpvjtag0:

Device: @2: 5CGTFD5(C5|F5)/5CGXBC:

SOF Manager: ...t_files/MAIN_CTRL.sof

log: Trig @ 2020/04/16 13:37:31 (0:0:0.1 elapsed) #2

Setup...
Scan Chain
>>
...
click to insert time bar
Type
Alias
Name
-64
-32
0
32
64
96
128
160
192
224
256
288
320
352
384
416
448
FCODE_C1[3..0]
STRIN[0]
ENIN_N[0]
ENOUT_N[0]
ENOUT_N[1]
ENOUT_N[2]
ENOUT_N[3]
ENOUT_N[4]
TRG_N[0]
...AIN_CTRL|dilogic_ctrl_top:INST_DIL_CTRL_TOP|trigger_i
...RL_GEN:0:INST_DIL_CTRL|DATA_TO_FABRIC_o[17..0]
...RL_GEN:1:INST_DIL_CTRL|DATA_TO_FABRIC_o[17..0]
...RL_GEN:2:INST_DIL_CTRL|DATA_TO_FABRIC_o[17..0]
...RL_GEN:3:INST_DIL_CTRL|DATA_TO_FABRIC_o[17..0]
2h
Ah
3FFFFh

Figure 87: Zero Suppression On, empty events 125 kHz (100ns/division).

With reference to the adopted CPV electronics ground scheme shown in Figure 87 there are three galvanically isolated low-voltage power supplies which feed different electronics modules:

- +5V: for the segment board, column controller and DILOGIC,
- ±2.8V: for 3Gassiplex cards
- +7V: for Readout unit.

In fact, all three groups of electronics are connected by cable grounding braids, and the quantity of such connecting cables is large:

- 160 coax cables connect the ground of 3Gassiplex cards and the ground of DILOGIC cards in each CPV module,
- 16 SMA coax cables connect the ground of Column Controller and the ground of the readout unit in each CPV module,
- 16-way twinaxial cable connect the ground of each Column Controller and the ground of the readout unit.

The adopted ground scheme ensures good equalization of the ground potentials of the distributed system. Additionally, one can connect the ground of all DC sources either at the Wiener power supply (40 m from the CPV module) or at the patch panel (10 m from the CPV modules). However, such connections can introduce the current loops which are not desired in case of quick change of the magnetic field.

It was also verified that there is no connection between the CPV electronics ground and the zero potential of the high-voltage system which is connected to the cathode plane of the proportional chamber. There is no connection to the CPV frame either, hence no connection to the ALICE space frame. The complete assembly of CPV electronics was run on the test bench at full power for almost a month. No degradation of the electronic boards was observed, and no dangerous hot spots were found during inspection with the IR video camera.

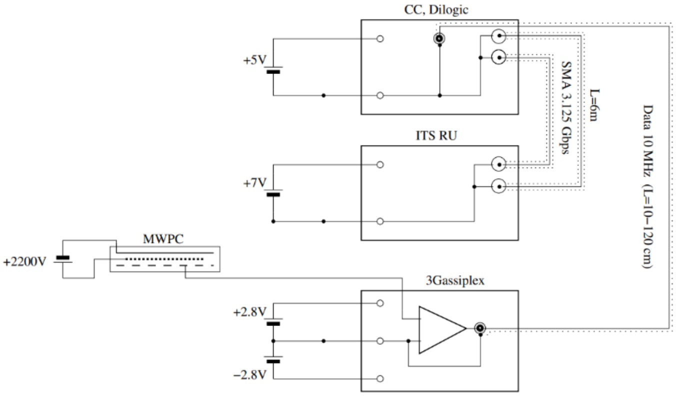


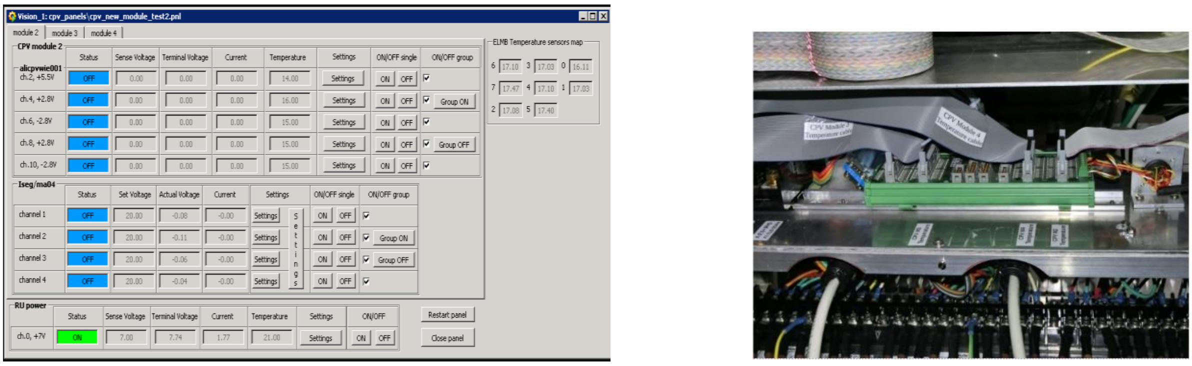
Figure 88: Adopted ground scheme for CPV readout electronics.

3.6 Installation and Commissioning of CPV electronics

Three CPV modules M2, M3 AND M4 were constructed and wired to process a total of 23,040 lead tungsten crystal channels. Each CPV module simultaneously processes 7,680 analogue channels. The all-new electronic front-end readout systems were commissioned in November 2021 and installed in the ALICE CPV detector. Commissioning included the following tests:

- Low voltage power supply test
- JTAG and Data Readout test
- Embedded Local Monitoring Board (ELMB) test

The CPV modules were connected to the readout unit through eight fully differential SMA cable pairs. Power to the CPV electronics and the readout unit is provided by the alicpvwie002 server connection to a low-voltage power supply. As seen on the Distribution Control System (DCS) panel and the power supply web server in Figure 89, both the voltages and currents were correctly read. In addition, the ELMB module was tested using a ribbon cable interface so to monitor and display the temperature of each module on the DCS web panel interface. All module temperature sensors read correctly, confirming proper wiring. The data readout was performed to confirm correct wiring from all the 8 column controllers of the modules.


Vision_1: cpv_panels\cpv_new_module_test2.pnl

module 2 | module 3 | module 4

CPV module 2

alicpvwie001	Status	Sense Voltage	Terminal Voltage	Current	Temperature	Settings	ON/OFF single	ON/OFF group
ch.2, +5.5V	OFF	0.00	0.00	0.00	14.00	Settings	ON OFF	☑
ch.4, +2.8V	OFF	0.00	0.00	0.00	16.00	Settings	ON OFF	☑ Group ON
ch.6, -2.8V	OFF	0.00	0.00	0.00	15.00	Settings	ON OFF	☑
ch.8, +2.8V	OFF	0.00	0.00	0.00	15.00	Settings	ON OFF	☑ Group OFF
ch.10, -2.8V	OFF	0.00	0.00	0.00	15.00	Settings	ON OFF	☑

ELMB Temperature sensors map

6	17.10	3	17.03	0	16.11
7	17.47	4	17.10	1	17.03
2	17.08	5	17.40		

Iseg/ma04

	Status	Set Voltage	Actual Voltage	Current	Settings	ON/OFF single	ON/OFF group
channel 1	OFF	20.00	-0.08	-0.00	Settings	ON OFF	☑
channel 2	OFF	20.00	-0.11	-0.00	Settings	ON OFF	☑ Group ON
channel 3	OFF	20.00	-0.06	-0.00	Settings	ON OFF	☑ Group OFF
channel 4	OFF	20.00	-0.04	-0.00	Settings	ON OFF	☑

Settings

RU power

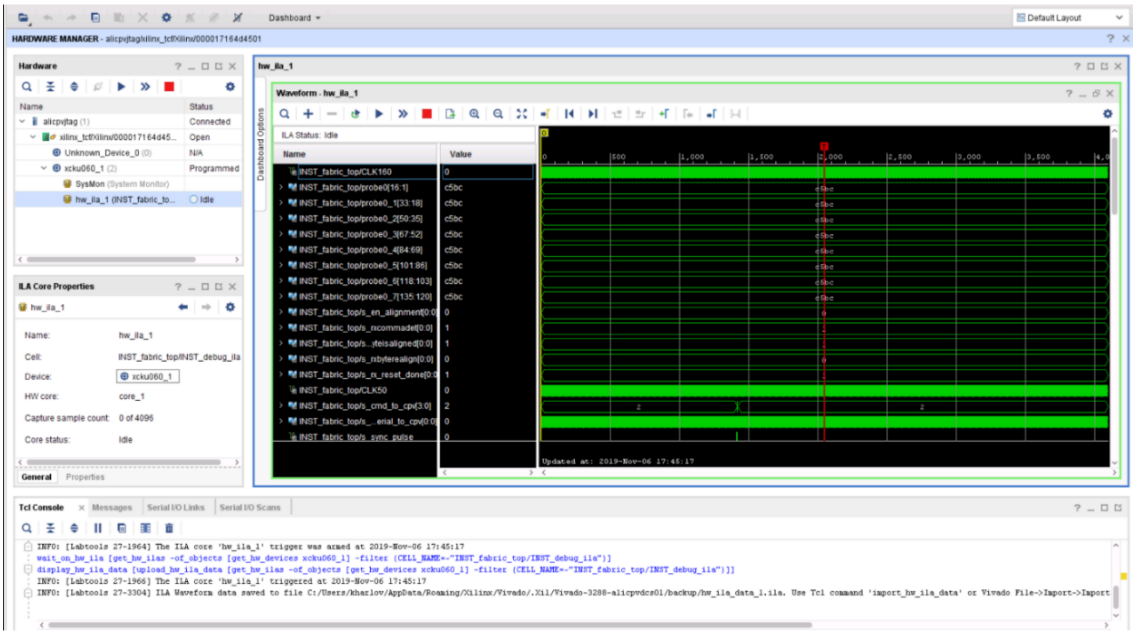
	Status	Sense Voltage	Terminal Voltage	Current	Temperature	Settings	ON/OFF
ch.0, +7V	ON	7.00	7.74	1.77	21.00	Settings	ON OFF

Restart panel

Close panel

Figure 89: DCS panel web server interface and installed temperature sensors for ELMB.

This test was done using Xilinx Vivado software application. As shown in Figure 90 the expected data pattern *0xc5bc* was received from all 8 column controllers of all 3 CPV modules.



HARDWARE MANAGER - alicpvjtag/xilinx_tcf/Xilinx/000017164d4501

Hardware

Name	Status
alicpvjtag (1)	Connected
xilinx_tcf/Xilinx/000017164d45...	Open
Unknown_Device_0 (0)	N/A
xcku060_1 (2)	Programmed
SysMon (System Monitor)	
hw_ila_1 (INST_fabric_to...	Idle

ILA Core Properties

hw_ila_1

Name:	hw_ila_1
Cell:	INST_fabric_top/INST_debug_ila
Device:	xcku060_1
HW core:	core_1
Capture sample count:	0 of 4096
Core status:	Idle

General Properties

hw_ila_1

Waveform - hw_ila_1

ILA Status: Idle

Name	Value
INST_fabric_top/CLK160	0
INST_fabric_top/probe0[16:1]	c5bc
INST_fabric_top/probe0_1[33:18]	c5bc
INST_fabric_top/probe0_2[50:35]	c5bc
INST_fabric_top/probe0_3[67:52]	c5bc
INST_fabric_top/probe0_4[84:69]	c5bc
INST_fabric_top/probe0_5[101:86]	c5bc
INST_fabric_top/probe0_6[118:103]	c5bc
INST_fabric_top/probe0_7[135:120]	c5bc
INST_fabric_top/s_en_alignment[0:0]	0
INST_fabric_top/s_rxcommadet[0:0]	1
INST_fabric_top/s...yteisaligned[0:0]	1
INST_fabric_top/s_rxbyterealign[0:0]	0
INST_fabric_top/s_rx_reset_done[0:0]	1
INST_fabric_top/CLK50	0
INST_fabric_top/s_cmd_to_cpv[3:0]	2
INST_fabric_top/s_...erial_to_cpv[0:0]	0
INST_fabric_top/s_sync_pulse	0

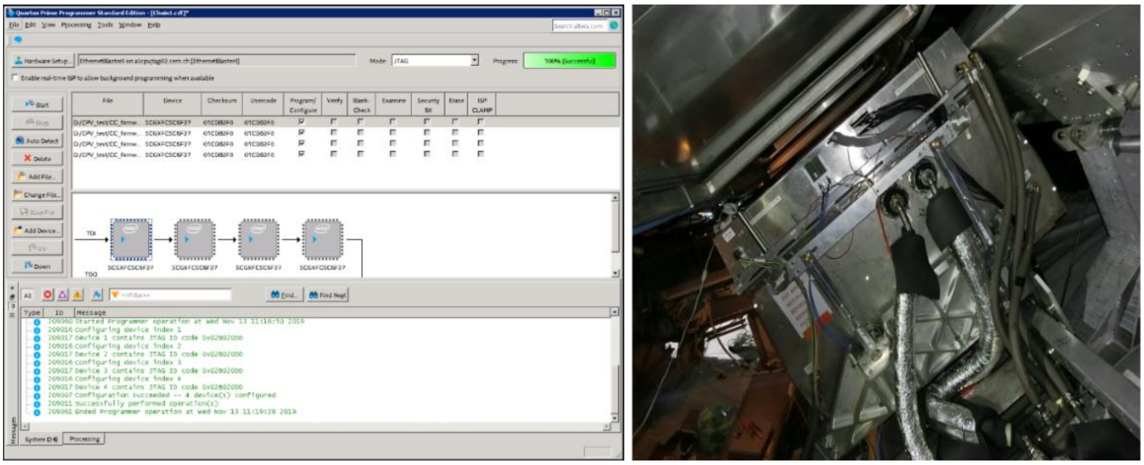
Updated at: 2019-Nov-06 17:45:17

Tcl Console | Messages | Serial I/O Links | Serial I/O Scans

```
INFO: [Labtools 27-1964] The ILA core 'hw_ila_1' trigger was armed at 2019-Nov-06 17:45:17
wait_on_hw_ila [get_hw_ilas -of_objects [get_hw_devices xcku060_1] -filter {CELL_NAME=~"INST_fabric_top/INST_debug_ila"}]
display_hw_ila_data [upload_hw_ila_data [get_hw_ilas -of_objects [get_hw_devices xcku060_1] -filter {CELL_NAME=~"INST_fabric_top/INST_debug_ila"}]]
INFO: [Labtools 27-1966] The ILA core 'hw_ila_1' triggered at 2019-Nov-06 17:45:17
INFO: [Labtools 27-3304] ILA Waveform data saved to file C:/Users/kharlov/AppData/Roaming/Xilinx/Vivado/.Xil/Vivado-3288-alicpvdcs01/backup/hw_ila_data_1.ila. Use Tcl command 'import_hw_ila_data' or Vivado File->Import->Import
```

Figure 90: Data Readout Test for CPV modules.

The Intel Quartus programming software and EthernetBlaster interface module were used to verify JTAG communication with all four FPGA column controller modules and electronic boards. The SRAM object file (SOF) format representing the firmware image was programmed at a rate of 1 Mbps to all 4 FPGAs as shown in Figure 91. All three CPV modules were successfully programmed repeatedly without any problems.



EthernetBlasterII on alicpvtag02.cern.ch [EthernetBlasterII]

Mode: JTAG

Progress: 100% (Successful)

□ Enable real-time ISP to allow background programming when available

File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank-Check	Examine	Security Bit	Erase	ISP CLAMP
D:/CPV_test/CC_firmw...	5CGXFC5C6F27	01CDB2F0	01CDB2F0	☑	□	□	□	□	□	□
D:/CPV_test/CC_firmw...	5CGXFC5C6F27	01CDB2F0	01CDB2F0	☑	□	□	□	□	□	□
D:/CPV_test/CC_firmw...	5CGXFC5C6F27	01CDB2F0	01CDB2F0	☑	□	□	□	□	□	□
D:/CPV_test/CC_firmw...	5CGXFC5C6F27	01CDB2F0	01CDB2F0	☑	□	□	□	□	□	□

```
Type  ID     Message
      209060 Started Programmer operation at Wed Nov 13 11:16:50 2019
      209016 Configuring device index 1
      209017 Device 1 contains JTAG ID code 0x02B02000
      209016 Configuring device index 2
      209017 Device 2 contains JTAG ID code 0x02B02000
      209016 Configuring device index 3
      209017 Device 3 contains JTAG ID code 0x02B02000
      209016 Configuring device index 4
      209017 Device 4 contains JTAG ID code 0x02B02000
      209007 Configuration succeeded -- 4 device(s) configured
      209011 Successfully performed operation(s)
      209061 Ended Programmer operation at Wed Nov 13 11:19:28 2019
```

Figure 91: JTAG programming (left) of all installed CPV modules (right)

3.7 Conclusion

Based on the above measurements, functional verification, and system integration tests, it was found that the newly developed FPGA Column Controller card is capable of simultaneously controlling and processing four sets of 5 DIL cards (2 CPV columns) at a clock frequency of 10 MHz, with each column containing 480 analogue GASSIPLEX channels, which is a two-fold improvement over the previous CPV read-out electronics. In addition, all manufactured PCBs were visually checked for any manufacturing defects. Based on the pedestal channel readings, open traces or connections on the PCB were checked, which was not confirmed. In addition, no overheating of the PCB or FPGA boards was observed (~38 °C) after the system was in operation 24 hours a day for a full month. The JTAG interface used allowed FPGA programming to be successfully performed with a shielded cable length of 30 m and a bandwidth of 1 Mbps, resulting in an additional tenfold reduction in programming time compared to the previous CPV electronics. In addition, the firmware is automatically reloaded from the flash memory after about 2 µs after the device is switched on again. Linear instead of switching voltage regulators were used to power all electronics, making the system insensitive to changes in magnetic field strength. Using a custom 4-bit protocol command, communication between CC and ITS RO was successfully established via the LVDS Firefly cable at 200 Mbps. In addition, the performance of the 3 Gbit/s gigabit transmitter links was evaluated, contributing to a value of 1 x 10^{-12} BER. With this newly developed CPV front-end electronics, a readout rate of 50 kHz is possible with a maximum detector occupancy of 27 channels per 3-Gassiplex card. Therefore, based on the above conclusions and results, both ALICE and the Russian institution HEP have agreed to proceed with the full installation of the system and commissioning of this new electronic readout system in the cavern pit by November 2020, one year ahead of the actual schedule for Run 3.

CHAPTER 4 - Multi-channel Data Acquisition and Real-Time Processing (MDART) ASIC

A common function in many electronic applications is data acquisition and processing. Such a function can be implemented with an inexpensive microcontroller or programmable logic device. Regardless of the application, a common challenge is the need for a reliable, low-cost, low-power, high-speed data acquisition system that can handle a large amount of data. Other desired characteristics of digitizers are related to measurement and data throughput, synchronisation of high channel counts in mixed-signal applications, and ease of automation, use, and integration [55], [56], [57], [58]. Electronic systems based on field-programmable logic gate arrays (FPGAs) and application-specific integrated circuits (ASICs) can help overcome the challenges of managing and processing the large amounts of data. Particle detectors such as CPV and HMPID are needed to find and potentially identify all particles emerging from a scattering event. The development of data acquisition systems in such detectors for colliding beam experiments presents several challenges. These include minimising material requirements, low power consumption due to limited space and power, potentially high data rates, and reliability at certain radiation tolerance levels [59]. Data acquisition systems involve the use of analogue-to-digital converters (ADC) to capture an analogue signal in digital form and send the processed signal over a communication bus such as PCI Express. Acquiring large amounts of data in the hundreds of millions or even bytes per second can easily lead to bus bottlenecks unless a system is designed to keep pace with ever-increasing technological demands for data resolution and speed trends. High-speed data acquisition systems employ FPGA and ASIC devices because they provide the capability for simultaneous continuous sampling, inline data processing and acquisition of analogue channels in a timely and synchronised manner before data is loaded into a memory buffer or transmitted remotely for further processing. In addition, FPGAs allow easy downloading of user-specific firmware written in a hardware design language such as VHDL. For some applications, error detection may not be sufficient, so a mechanism is needed to recover from that error as well. This is especially critical for memory devices where bits may have been corrupted. In addition, error correction and detection are the first step in a communication system, whether the data is transmitted over a wireless or wired communication channel, where some bits may be corrupted due to channel noise. An example of a robust error detection algorithm is the transmission of a Cyclic Redundancy Check field (CRC) along with the data packet. Examples of error-correcting codes include the use of linear, convolutional, and Reed-Solomon codes. Review of several commercially available analogue electronic modules that use data serializers for data transmission found them to lack the hardware protection of fault-tolerant and recovery redundancy functions. Space and safety-critical applications may be exposed to various radiation doses and ionising particle impacts, affecting the bit error rate of data transmission or even the lifetime of a circuit. By applying hardware or information redundancy, including error correction codes, data transmission errors can be mitigated in data acquisition systems. This research also presents the design and development of a new ASIC-based system suitable for implementing a high-speed multichannel data acquisition real-time (MDART) system. The remainder of this chapter provides an overview of the newly developed ASIC-based system architecture, including a description of the implemented LVDS transceiver, CRC, and the zero-suppression algorithm. Simulation and measurement results also follow.

4.1 MDART SPECIFICATION

As described in the previous chapter, the front-end electronics of the RO CPV particle detector uses a 10 MHz digital signal processor in a square, low-profile 64-pin QFN package capable of processing up to 64 analogue channels sequentially, in groups of 16, 32, 48, or 64 channels, with a total power consumption of 60 mW. The CPV DILOGIC processor does not have redundancy or error detection and correction strategies to avoid problems associated with single events or data errors. A single DILOGIC signal processor chip is required to process the 12-bit digital data from ADC, requiring an increase in PCB size and higher cost to process additional analogue channels. Considering the ALICE luminance requirements for run 3 of $6x10^{27}$ $cm^{-2}s^{-1}$, such a DILOGIC processor is challenging to achieve the 50 kHz RO rate. In addition, the DILOGIC processor does not provide solutions to run CPV or HMPID detectors without degrading their characteristics at room or other temperatures. Due to these limitations and bottlenecks, this research will present the development of a new ASIC chip suitable for various DAQ applications, including CPV and HMPID physics experiments or high-speed multichannel counting devices for measurement computer systems. The tentative target requirements for this new ASIC chip called MDART (Multi-channel Data Acquisition Realtime Processing) are listed in Table 21.

MDART ASIC Specification	
Number of Parallel MWPC Channels to be processed	480
Data Rates	4 Gbps
Bit-error Rate (BER)	$>10^{-9}$
Power Consumption (mW) per channel	< 2 mW
Technology (nm)	180 nm
Supply Voltage (Volts)	< 5V
ASIC Die Area	< 10 mm^2
Digital Interfaces	Four 12-bit Digital Input Ports
Operating Clock Frequency (approx.)	100 – 156.25 MHz
Package	CQFP 160
Memory Storage	>1 kbytes
Temperature Range	0 to 80 °C
LVDS Transmitter (SERDES)	To be included
LVDS (overflow/underflow) detection	Yes
Error Detection and Handling	Yes
Zero Suppression	Yes
Pedestal Configuration	Yes
Transmission of Data over distances 5m+	Yes
Serialization of Data	Yes
Integrated Controller	Yes
Typical Applications	High-Speed Data Acquisition Systems, Instrumentation Systems, Particle Physics experiments.

Table 21. ASIC Specifications.

MDART architecture should be easily integrated with current CPV and HMPID electronics, with improved fault and disturbance detection functions replacing the current functionality of the CPV DIL5 cards, so that a minimum event readout rate of 50 kHz can be achieved with lower power consumption. The target ASIC parameters given in Table 21 were realized with reference to the deployed CPV FEE measurement results and the literature reviewed on ASIC technology used in various particle detectors. The MDART device targets applications where performance is the primary goal, with the integrated implementation of specific algorithms for error correction and detection. MDART shall have at least 1 Kbyte RAM, for pedestal storage and subtraction. In addition, provide for the use of integrated FIFO buffers for data synchronisation between the

MDART controller and LVDS transmitter. As described in Table 21, the MDART chip should also support different market segments, such as data acquisition, instrumentation systems, and physical experiments. MDART should be adapted as a replacement for the DILOGIC chip in CPV and HMPID detectors. Therefore, various integrated circuit modules must be custom designed or standard IPs modified as needed. Such modifications may be required to implement the necessary control logic for pedestal configuration, error handling and detection, data transmission, and FIFO boundary conditions. Due to budget constraints and considering speed and power requirements, XFAB-180nm technology will be used, considering gate count and core constraints for a 10 mm^2 chip area.

4.1 MDART System Architecture

The main goal of the MDART ASIC chip is to increase the data transfer rate, i.e., throughput, by at least five times with a reduced PCB form factor compared to the performance of the CPV DILOGIC signal processor chip. Therefore, based on the evaluation and results of the research work explained in Chapters 2 and 3, the following digital design strategies were considered:

- Higher Clock Frequency,
- Parallel Processing,
- Wider Data buses,
- Use of Multi-processors,
- Error detection,
- Fault Tolerant,
- Zero Suppression,
- Cost, Technology, and other project constraints.

After reviewing the current market trade-off, the 80-20 guideline, and the best possible combination of the above techniques, the proposed MDART architecture, shown in Figure 92 below, was adopted.

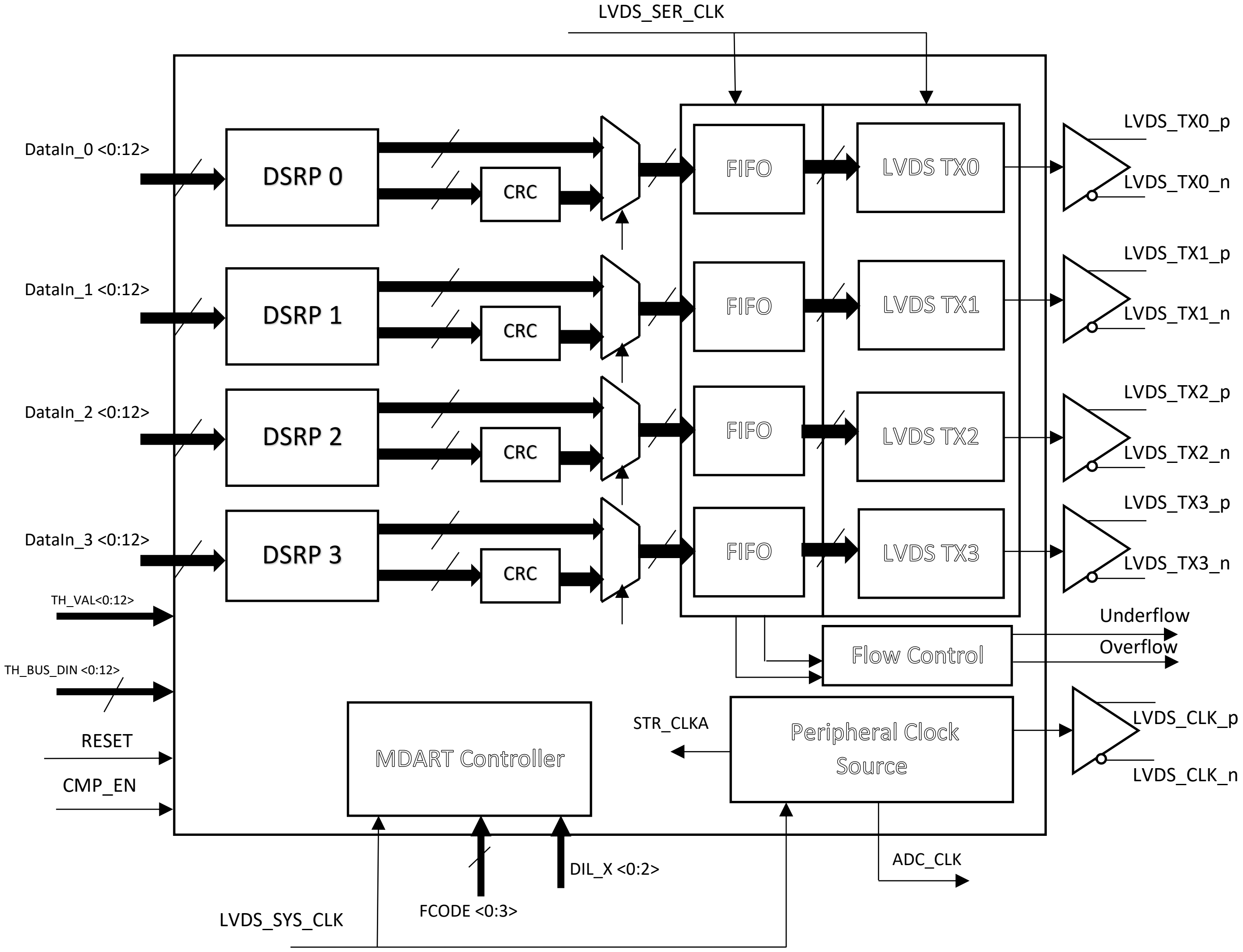


Figure 92: MDART System Architecture

MDART operates with multiple clock domains. The external LVDS_SYS_ CLK is a clock source for the Digital Sparse Readout Processors (DSRPs) and other external devices (e.g., ADCs) connected to MDART. LVDS_SYS_ CLK is also a clock source for the Peripheral Clock Source (PCS) module to clock all external and internal components. For example, the clock STR_CLKA drives the DSRPs logic to perform a parallel synchronous data transfer from an external ADC device to the dual-clock asynchronous FIFO (DCFIFO) data input port. Another clock source is the LVDS_SER_CLK, which clocks both the DCFIFOs and the LVDS shift registers with parallel input and serial output. The serial clock of LVDS_SER_ CLK should be higher than that of LVDS_SYS_CLK by an integer multiple of *r*.

The highest possible clock frequency for LVDS_SYS_CLK and LVDS_SER_CLK depends on the required overall throughput of the system and BER as described in the following sections. The four 12-bit DataIn_n ($0 \leq n \leq 3$) digital ports allow parallel connection of various external digital devices such as ADCs. This feature allows MDART to be used in various data acquisition applications with different resolution requirements.

The 4-bit function code FCODE < 3:0 > is a digital input port that allows you to instruct the MDART controller to perform various operations under the control of the external control device, such as configuring socket or subtraction memories using either the DataIn_n or TH _BUS_ DIN < 11:0 > ports. Various other function codes are required to perform analogue measurements with or without zero suppression. TH _VAL< 11:0 > input port is used to perform zero suppression via digital comparators and a subtractor to eliminate noisy channels when the normal data acquisition process is started. Alternatively threshold or pedestal programming can be performed via the analogue ADCin<x>[11..0] channels, where x represents a digital port number 1 to 4.

The digital comparators and subtractors are activated via the CMP_EN input. Front-end analogue amplifiers such as the GASSIPLEX chip with a serial analogue output can be used in a DAQ system to process a certain number of analogue input channels. The DIL_X< 2:0 > input is used to configure the required number of clock cycles, which depends on the number of analogue channels to be processed. The asynchronous input RESET is used to perform a system reset and clear the necessary system control flags. To avoid overflow conditions or data loss, a flow control module is used. The underflow and overflow output pins indicate the status of the DCFIFOs. Parallel processing is achieved by reading all four 12-bit DataIn_n ports simultaneously. Compared to previous CPV electronics and considering XFAB 180 nm 6-layer silicon technology and cost, data throughput is increased by an internal 21-bit data path.

Error detection is optionally achieved by Cyclic-Redundancy Check (CRC). Different data acquisition systems may need to use and apply CRC. The serial LVDS transmitter optionally transmits the data packet with a CRC field that can later be processed by the receiver. Fault tolerance is possible by using triple modular redundancy (TMR) strategies via tuning circuits, which must be included in DSRPs. By replicating multiple CRC, serial transmitters and DCFIFO memory components, the TMR strategy can be easily applied depending on the application needs. The DSPR EN_FIFO output is used to activate the serial transmitter for data writing in DCFIFO.

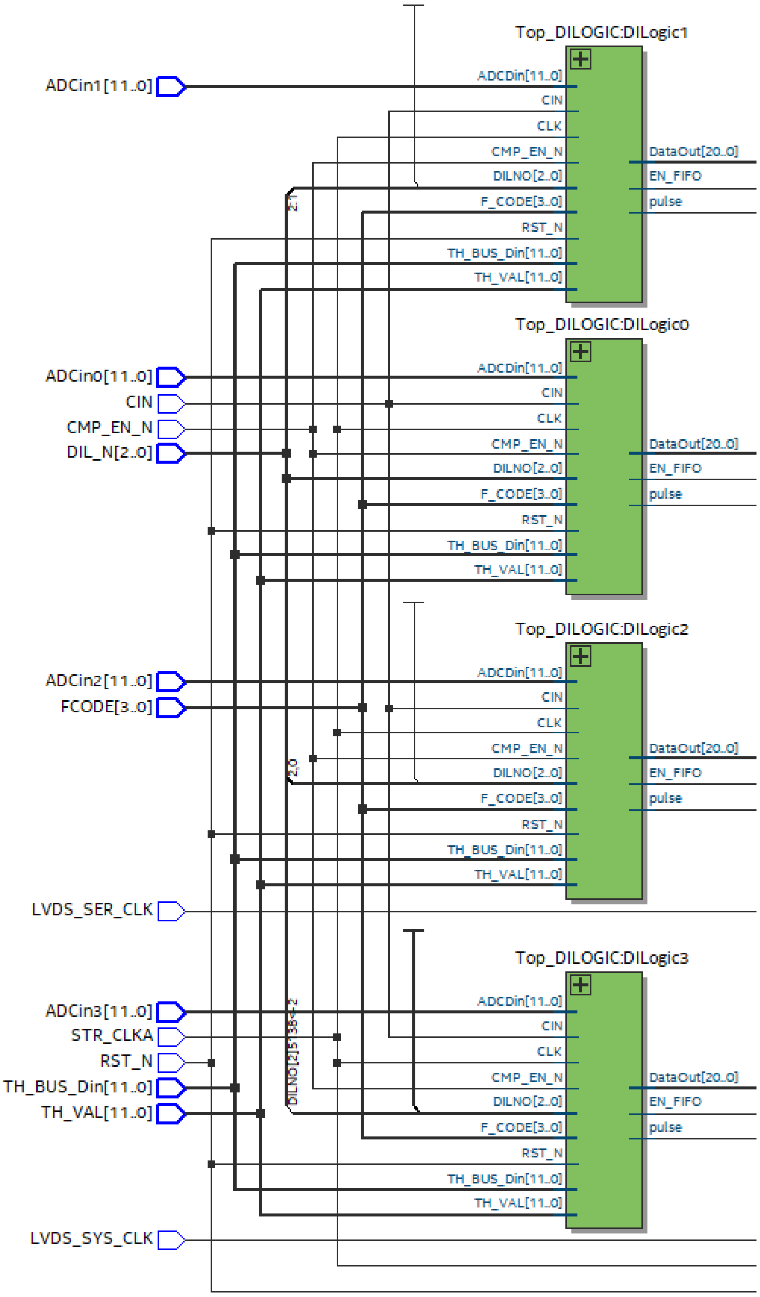


Figure 93: illustration of four DSPR modules called DILOGIC and their interfaces.

Figure 93 illustrates the various interfaces of the DSRP modules, whose 21-bit digital output DataOut[20..0] is then passed to the asynchronous FIFO module and then multiplexed and transmitted serially. EN_FIFO control line activates the FIFO write control line.

4.2 Implementation Guidelines for XFAB XHT018 180nm 6-Layer Technology

Taking into consideration the design requirements, and budget constraints the MDART ASIC prototype chip shall be implemented using the X-FAB XH018 0.18µ CMOS E-FLASH (MET3, MET4, METMID, METTHK), 180 nm 6-metal layer technology, and the standard cell digital library D_CELLS_3V. This technology has a gate density of 101.5 kGE/mm^2 and supports low supply voltage ranges of 3.3V or 2.2 V. The standard cell digital library D_CELLS_3V is suitable for standard speed and low power applications [27]. Figure 84 illustrates the recommended ASIC digital design and implementation flow, consisting of the following four main stages:

- Register Transfer Level (RTL) Design and Verification
- Synthesis
- Physical Implementation (Place and Route)
- Signoff

The Design and Verification stage always starts from the required specification, followed by high level modelling and simulation to verify functionality. At this stage waveforms can be used to debug simulation failures. The output of this design and verification phase will be an RTL code in either VHDL or Verilog, which is then imported to the Synthesis phase. Elaboration via Genus Synthesis Tool is then performed so to obtain a generic netlist in terms of optimised generic logic cells. The mapping and optimisation into logic cells is performed from the standard cell library, using the required timing, area, and power constraint files. The synthesis process will result in a Verilog netlist, which specifies the connectivity between various instances from the process library. The Place and Route stage will then perform the implementation of the full layout, which can then be sent off for fabrication. The layout design can be exported in Graphic Design System (GDSII) layout format for fabrication, Verilog gate level netlist, layout parasitics and Standard Delay Format (SDF) file. All these files are required so to perform a range of signoff checks, before the chip is manufactured.

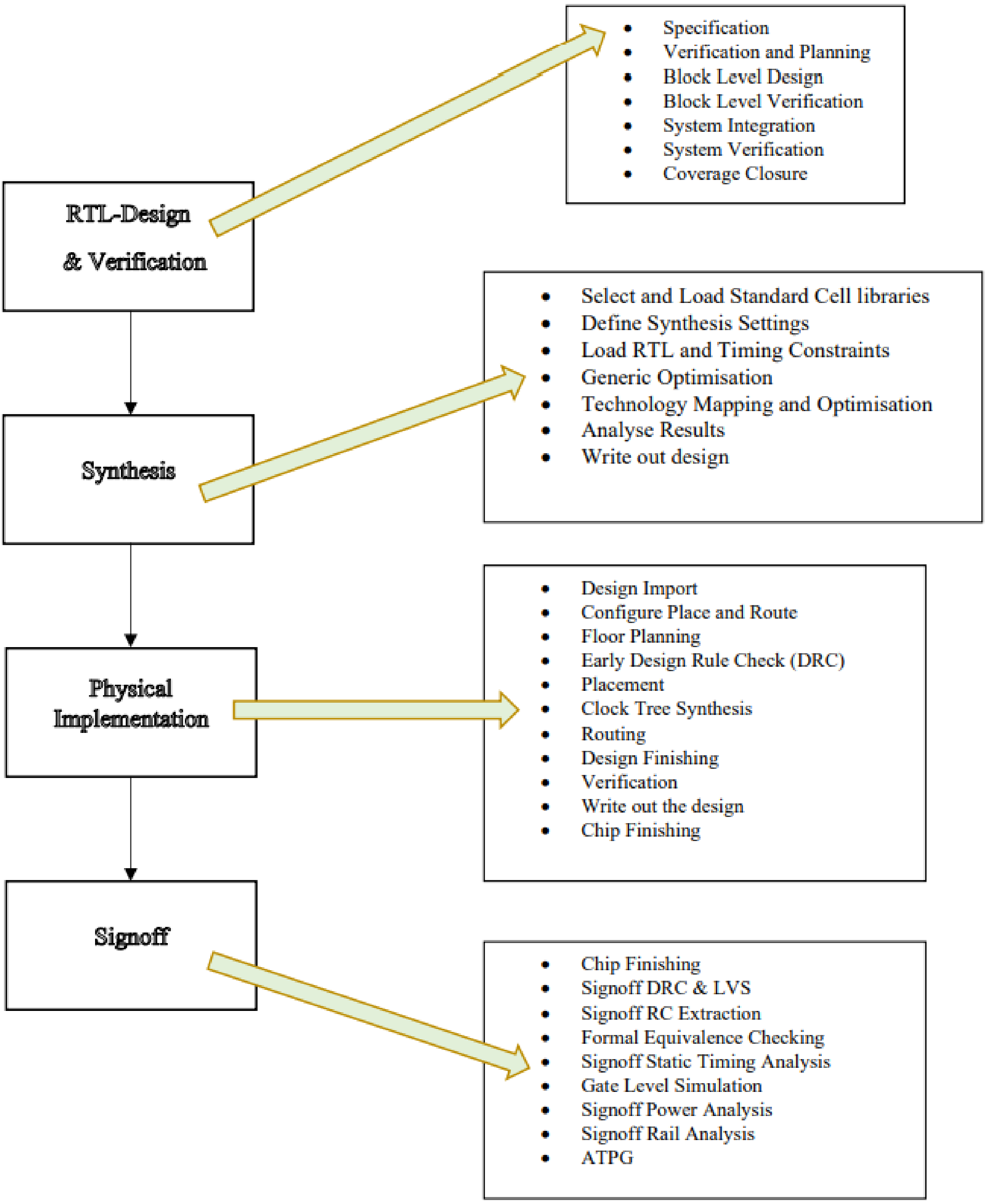


Figure 94: ASIC Design and Implementation Flow.

4.3.0 DSPR - Design and Functional Verification

The MDART system architecture enables the implementation of a complete data acquisition system with gigabit performance. MDART is an integrated digital signal processing chip and consists of five 12-bit digital input ports to facilitate parallel interfacing to various external analogue-to-digital converters such as the AD922ARS 12-bit ADC. Each integrated MDART DSRP processor consists of three 64-byte SRAMs to optionally store pedestal values. The purpose of the SRAM memory is to calibrate the ADC channels so as to compensate for input readings. The preloaded pedestal values of the SRAM memory can be used to apply the zero compression algorithm via an integrated DSRP comparator and subtraction circuit. Figure 85 illustrates the internal architecture of a DSRP module. The data generated by each DSRP module consists of a 21-bit word containing a DSRP identification number, the channel address, and the captured 12-bit digital code. The subtractors were implemented using the carry lookahead approach to minimise delay compared to standard subtractors. This takes advantage of the fact that at each bit position it can be determined whether a carry is either propagated or generated at a particular bit. Each DSRP module implements a fault-tolerant circuit design by implementing a triple-modular redundancy (TMR) strategy, which is a type of modular *N*-tuple redundancy, where *N* is an odd number.

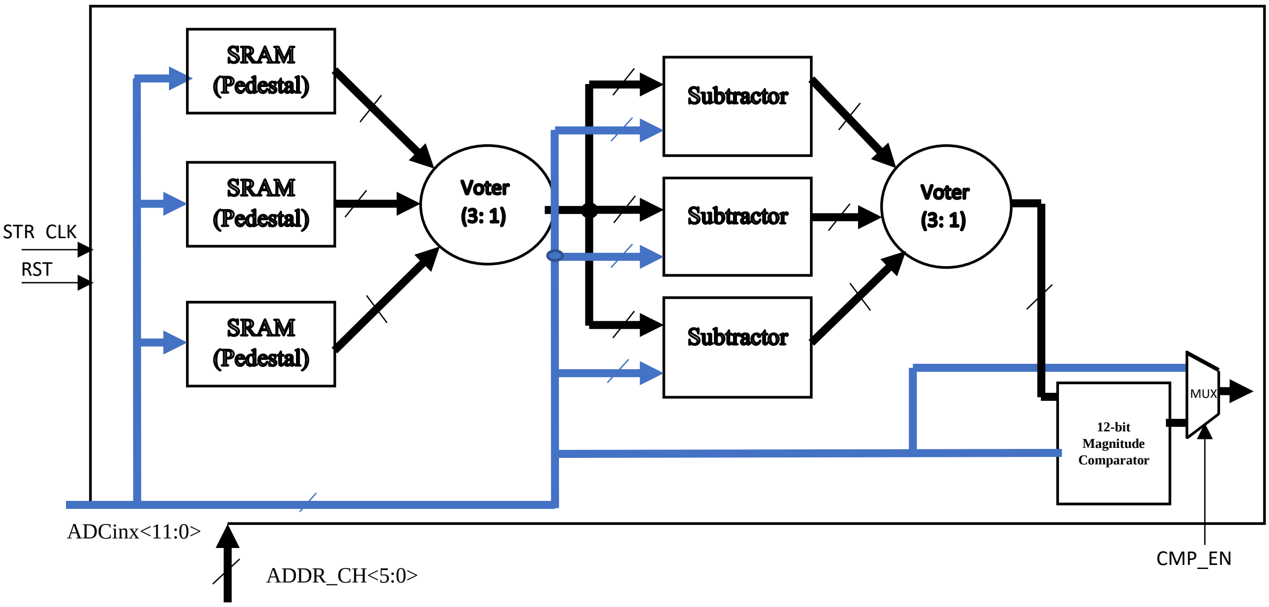


Figure 95: Illustration of one DSPR module.

Radiation exposure can cause interference from single events. Therefore, the use of fault-tolerant circuits is essential in physical experiments used for error correction and detection using a tuning circuit [61], [62]. A TMR tuning strategy [63] has been implemented in each DSRP module, for both the SRAM and the look-ahead carry generator subtractor circuitry, enabling fault detection and recovery. With the TMR strategy implemented, each SRAM or subtractor module operates independently and in parallel. The reliability of the selector circuit is very important to obtain the final output. At a low order (N < 7), the voter circuit is more efficient.

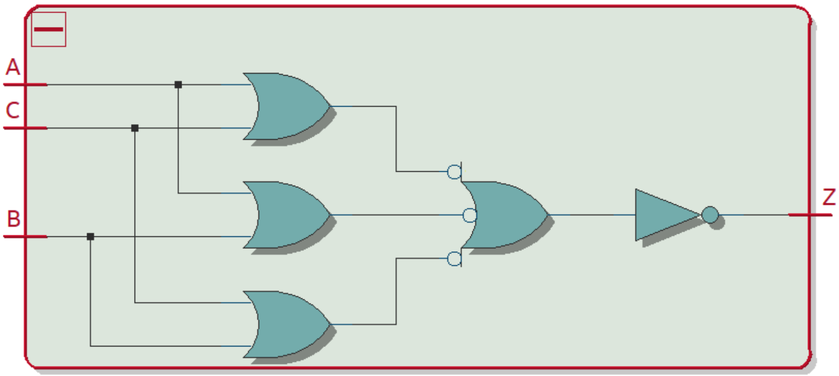


Figure 96: NOR implementation of Voter circuit.

All possible combinations of entries in a majority vote-based arbitrator are presented in Table 24. The corresponding Boolean expression is given in Equation 2, below.

$$Z = \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC \qquad \ldots (2),$$

Then using Karnaugh map Equation (2) simplifies to

$$Z = BC + AC + AB \qquad \ldots (3),$$

Equation (3) represents a combinational circuit of AND/OR logic operations, which can be structured using NOR gates as shown in Figure 86.

A	B	C	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Table 22: Truth table for an arbitrator based on majority voting.

The MDART digital controller manages zero suppression, pedestal subtraction, data format, local storage in memory FIFO, and data transfer to the DAQ via 1 Gbit/s LVDS transmitter links.

The MDART controller can be operated in two different modes, i.e., normal or calibration mode. In calibration mode, the SRAM memory is programmed with pedestal values using either the ADCinx < 11:0 > or TH _BUS_Din < 11:0 > digital input ports, as shown in Figure 86. Each digital pedestal value represents the channel noise. Each DSRP module can handle up to 64 channels in groups of 16, 32, 48, or 64 channels, which should be calibrated before normal readout operation is activated. In normal operation, each external ADC module, such as the AD922ARS, is clocked to sample each analogue input channel and shop the 12-bit ADC digital codeword in each of the four FIFO memories, with zero suppression either enabled or disabled. In addition, the CRC function can be either enabled or disabled in normal mode depending on the application requirements.

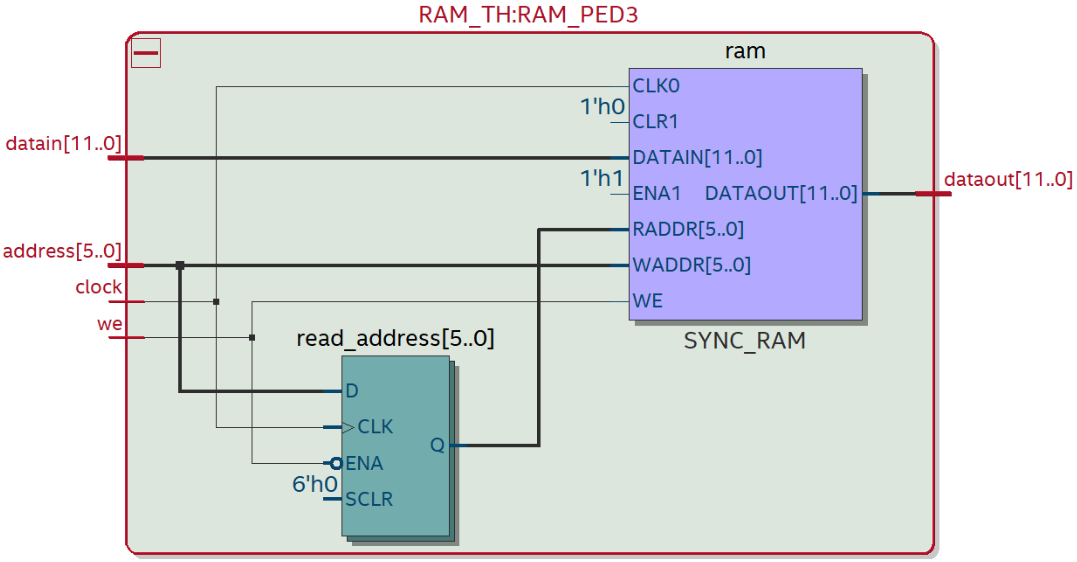


Figure 97: SRAM memory for pedestal subtraction.

The MDART device contains four FIFO memories with 64 words x 21-bits. Each FIFO memory is loaded with the digitized amplitude information (12-bits) and the address of the selected channels (6-bits), and a 3-bit DSRP identification number. Data will be written to a FIFO module on the clock rising edge and when it's control lines *datain_val* and *datain_rdy* inputs are high. Data can again be read from a FIFO module on the rising clock-edge when the control outputs *dataout_val* and *dataout_rdy* are both high. Additionally, a state machine monitors the state of both *dataout_val* and *dataout_rdy* outputs so to indicate if FIFO is either full or empty. The depth of the asynchronous FIFO is a positive value of 16. In addition, read and write FIFO pointers use the Grey coding scheme, where a complete sequence is always a power of 2. The reason for this is the single bit change property required for the Grey encoding scheme.

The following fixed 5-bit Grey encoded sequence is adopted. This makes it possible that only one-bit changes with every clock edge cycle.

"00000" > "00001" > "00011" > "00010" > "00110" > "00111" > "00101" > "00100" > "01100" > "01101" > "01111" > "01110" > "01010" > "01011" > "01001" > "01000" > "11000" > "11001" > "11011" > "11010" > "11110" > "11111" > "11101" > "11100” > "10100" > "10101" > "10111" > "10110" > "10010" > "10011" > "10001" > "10000"

A read pointer and a write pointer are used to access the asynchronous FIFO data storage memory locations. As shown in Figure 98, the read pointer is controlled by the read clock *clk* and the read enable pin *ENA1*, while the write pointer is controlled by the write enable (*WE*) and the *CLK0* clock pins. One major implementation problem with asynchronous FIFO is the comparison between the read and write pointers. Therefore, the Grey coding strategy is used for the synchronisation between the read and write pointers, which operate in different clock domains. Further, grey-encoding is used to reduce the possibility of a metastability problem, improving synchronisation between the read and write clock domains so that the full and empty states of the FIFO are correctly detected. Additionally, grey-encoding is useful to transfer the 27-bit stream of data from one clock domain to another. Since it is not possible to increment the Grey code, each Grey code is converted to a binary value whose value is incremented and then converted back to a Grey code.

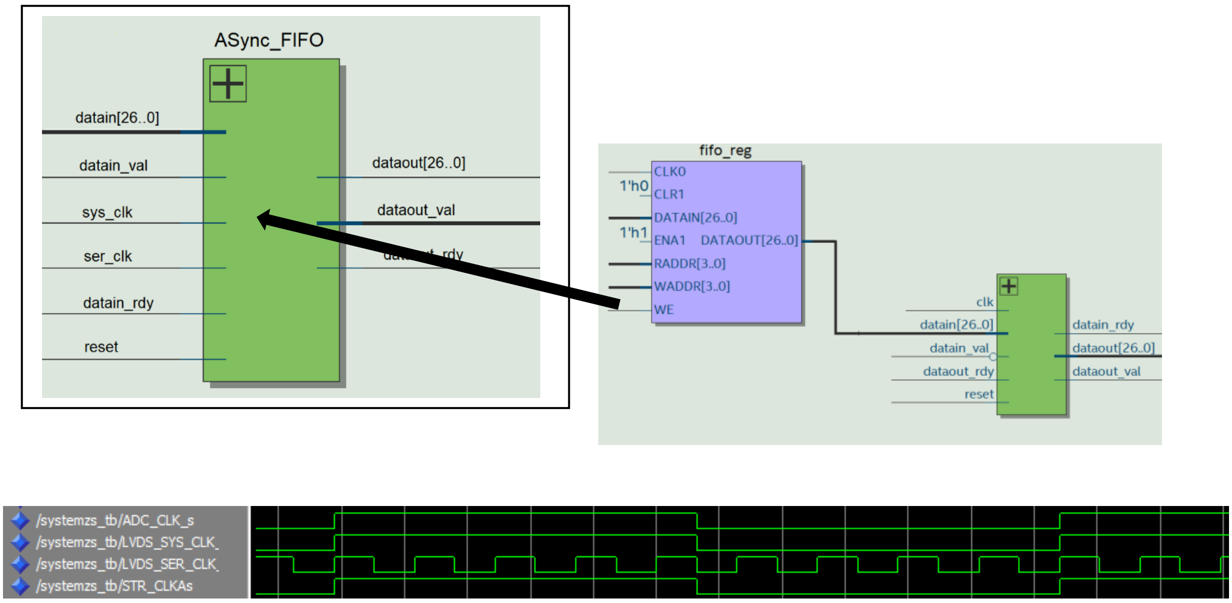


Figure 98: (a) Top Figure - Interface for Asynchronous (Async) FIFO memory (b) Bottom Figure - LVDS_SYS_CLK and LVDS_SER_CLK signals (5ns/div).

Four LVDS transmitter modules are used to independently serialize 27-bit parallel input data lines into three fully differential independent data lanes *lvds_tx0_D<2...0>*, *lvds_tx1_D <2...0>*, and *lvds_tx2_D<2...0>*. Each LVDS transmitter requires two separate clocks, serial clock *ser_clk* and system clock *sys_clk* for correct operation. The system clock *sys_clk* is synchronous with the input data, whereas the frequency of the serial clock *ser_clk* is *r* times higher than that of the system clock. Ratio *r* is given by:

$$\boldsymbol{r} = \frac{\boldsymbol{data\ width}}{\boldsymbol{number\ of\ serial\ data\ lanes}} \qquad \dots (4)$$

Therefore, for a data width (*dw*) of 27- bits and 3 serial lanes, then the *ser_clk* should be 9 times higher than the *sys_clk* as shown in Figure 98 (b). Serialization and transmission of data is done after performing reset operation and on the rising clock edges of LVDS_SYS_CLK_s (*sys_clk*) and LVDS_SER_CLK_s (*ser_clk)*. Setting LVDS_SER_CLK to 156.25 MHz, and LVDS_SYS_CLK_s at 17.36 MHz, a data transmission rate of up to 0.46 Gbps per serial transmitter lane is achieved. In addition, the integrated asynchronous LVDS FIFO detects the required data rate in and out of the LVDS transmitter during operation. Error flags are set if the ratio between the system and serial clocks is not set correctly or if the asynchronous FIFO overruns or underruns with or without data. The LVDS transmitter can be configured with up to eight lanes. Considering the availability of the CPFP 160 package for the MDART device, the MDART LVDS transmitter has been configured with a 3-lane option. The timing diagram shown in Figure 99 illustrates the operation of the LVDS and FIFO modules, where the 27-bit parallel data from the FIFO interfaces *dataout<26:0>* is serialized and transmitted via the 3-lane LVDS output data lines *tx_out<3:0>*. Additionally, as can be observed the system clock is 9 times lower than the serial transmitter clock.

The FPGA implementation of the serial transmitter occupies a total of:

- 432 Memory block bits,
- 52 Adaptive Logic Modules, and
- 116 registers.

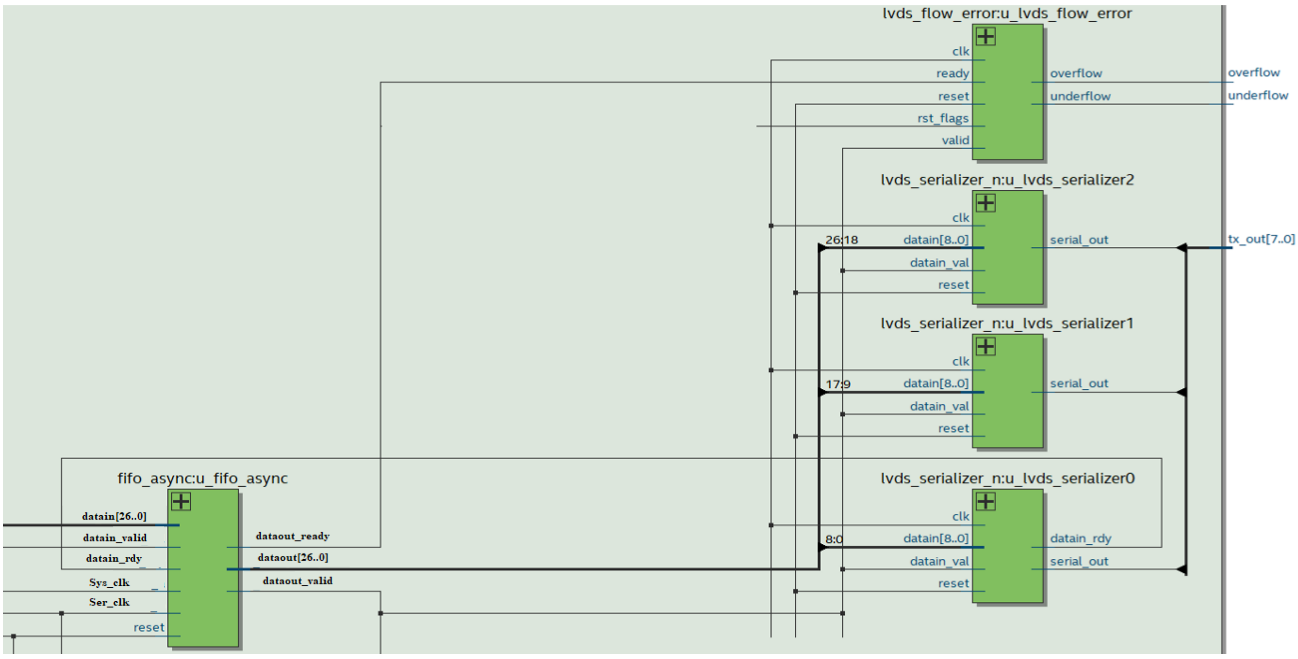


Figure 99: System interfaces between Asynchronous FIFO and four data serializers with and r ratio of 9.

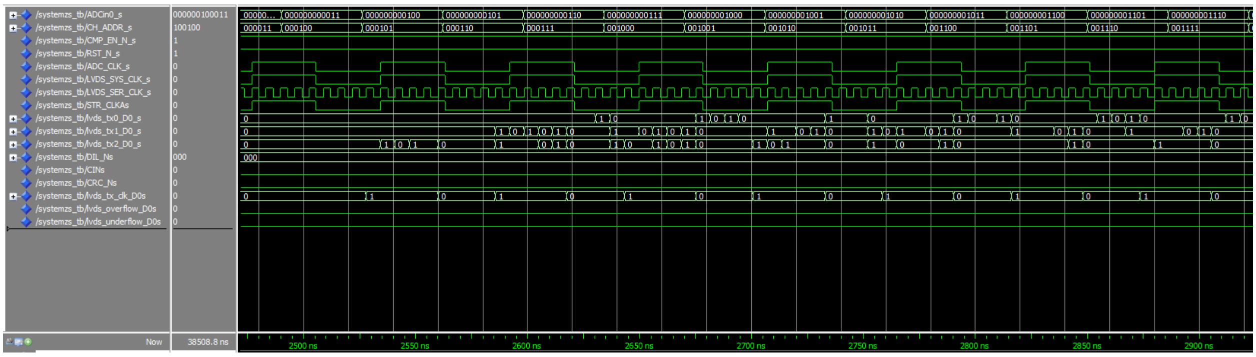

/systemzs_tb/ADCin0_s 000000100011
/systemzs_tb/CH_ADDR_s 100100
/systemzs_tb/CMP_EN_N_s 1
/systemzs_tb/RST_N_s 1
/systemzs_tb/ADC_CLK_s 0
/systemzs_tb/LVDS_SYS_CLK_s 0
/systemzs_tb/LVDS_SER_CLK_s 0
/systemzs_tb/STR_CLKAs 0
/systemzs_tb/lvds_tx0_D0_s 0
/systemzs_tb/lvds_tx1_D0_s 0
/systemzs_tb/lvds_tx2_D0_s 0
/systemzs_tb/DIL_Ns 000
/systemzs_tb/CINs 0
/systemzs_tb/CRC_Ns 0
/systemzs_tb/lvds_tx_clk_D0s 0
/systemzs_tb/lvds_overflow_D0s 0
/systemzs_tb/lvds_underflow_D0s 0
Now
38508.8 ns
2500 ns
2550 ns
2600 ns
2650 ns
2700 ns
2750 ns
2800 ns
2850 ns
2900 ns

Figure 100: Timing diagram showing LVDS lane D0, MDART clocks and the error flags lvds_overflow_D0s, and lvds_underflow_D0s.

MDART ASIC implements CRC for error detection. For the selection of a good CRC polynomial, one must take into consideration not only the size of the CRC, but also the data word length. Generally, CRCs detect various types of errors including for example single-bit, two-bit, and three-bit errors, depending on the type of selected polynomial generator. Mostly, CRC calculations can be implemented with linear-feedback shift registers (LFSRs). The LFSRs method allows the subtraction without carry to be performed using XOR function and division operation through shifting. Every bit is serially shifted one bit at a time, and then the frame a data frame is examined. CRC computation done via the serial LFSR method is suitable for serial-input bit streams, however high-speed communication systems may have data words of 8, 16 or 32-bit word lengths. As described in [28] there are various techniques that can be used to compute parallel CRC, examples include Table-Based algorithms for pipelined and F-matrix based parallel CRC generation. Parallel CRC computation can be implemented using recursive mathematics, where manual calculations can be done, but it could lead to a tedious process and could also be error prone especially when processing multi-bit wide data paths such as 21-bits in length. Therefore, after performing the necessary manual calculations as described in chapter 4 of this work, using the F-matrix method, the necessary VHDL CRC function was implemented.

The available Easics CRC tool was used to verify the implemented CRC function [65] and confirm the correct CRC hardware implementation, as shown in Figure 102. As shown in Figure 101, the Probability of undetected errors (P_{ud}) for a 21-bit data word is 10^{-15} with a Hamming Distance (HD) of 2 i.e., undetected errors can be observed with at least 2-bit inversions [66]. CRC5 performance degrades for data words with 10-bit length and lower. A 21-bit CRC5 checker hardware module implements the polynomial $x^5 + x^2 + 1$, thus, enabling error detection that may occur during data transmission. The use of this CRC5 polynomial is suitable to protect data wards of length 11-bits and longer as explained in [29]. Taking into consideration the trade-off between maximum number of undetected errors, word length and hardware requirements, then CRC5 has been adopted. The 5-bit encoded CRC checksum value is then appended at the end with the 27-bit message, so to be transmitted via LVDS link, and later decoded by the receiver. Using the same initialisation CRC value at the receiver, then if the received CRC checksum value matches the receiver's calculated one for the received data word, then there is a very small probability of data being corrupted. However, in this case, and in practice the probability can be considered as zero and result accepted.

The VHDL implementation for CRC5 makes use of eight Adaptive Logic Modules (ALMs), and five registers. Before the CRC value has been calculated, the CRC register needs to be initialised with a known CRC value, which in our case is binary value "10011". Such value is required both at the transmitter and receiver so to get the same initialisation conditions. As shown in Figure 102, CRC5 is represented by five flip-flops connected as a Parallel-In-Parallel-Out (PIPO) register, known as CRC register.

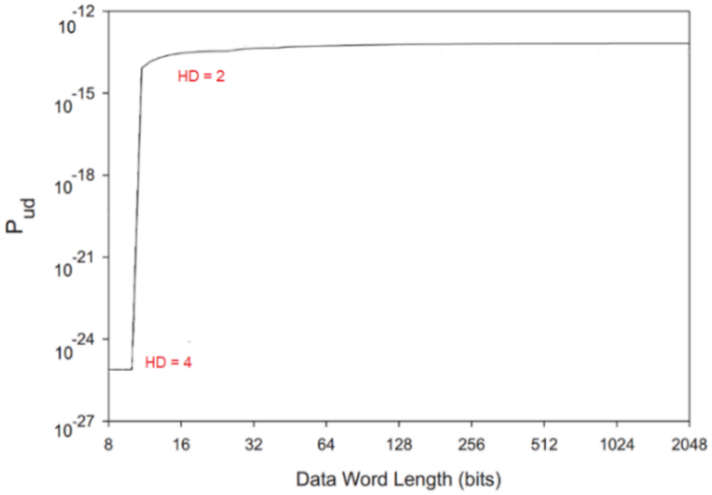


Figure 101: Performance of 5-bit CRC using polynomial x ^{5} + x^{2} + 1 [29].

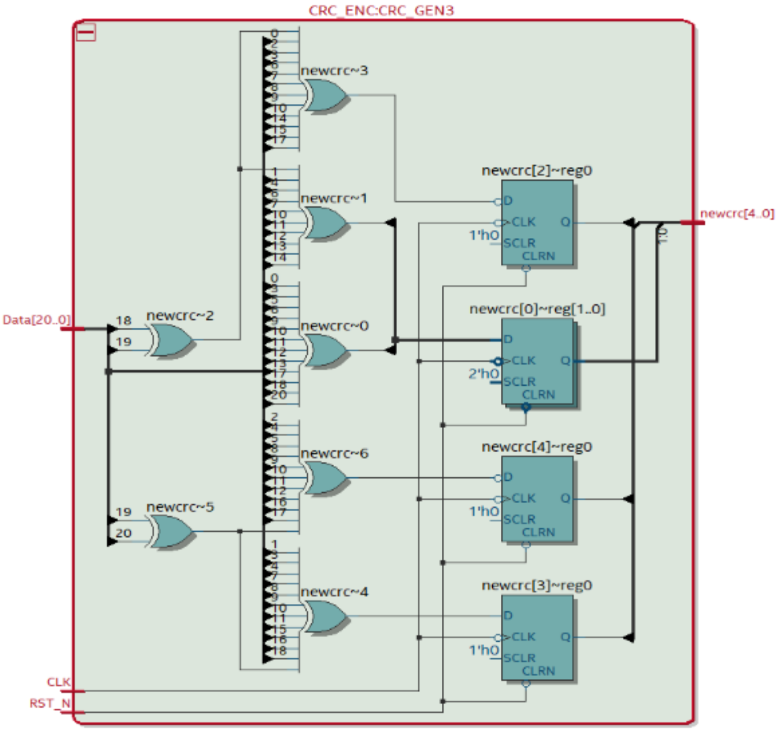


Figure 102: Parallel CRC-5 architecture using polynomial x ^{5} + x^{2} + 1.

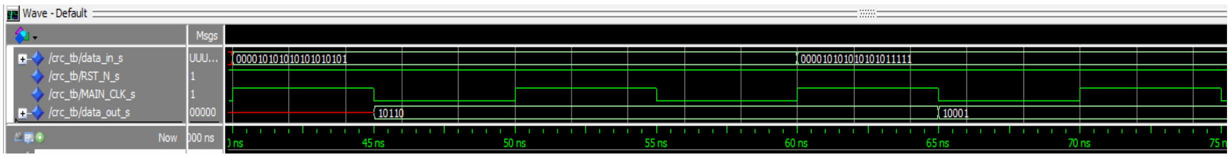


Figure 103: Timing diagram showing CRC5 output for two 21-bit values (2ns/division).

4.2 Measurement Tests for the proposed MDART Architecture on the Cyclone V FPGA

Before proceeding with the design and implementation of the MDART ASIC chip, the VHDL-based developed architecture was tested on a Cyclone V GX FPGA device. As described in the following sections, waveform measurements were made using the Quartus SignalTap Logic Analyzer. The required system clocks were generated by a 3-port PLLCLK module. An external 50MHz REF _CLK signal is used as the clock source for the internal PLLCLK module. Other control signals such as Carry In CIN, Comparator Enable CMP_EN_N, RST_N, FCODE [3:0] were set by external switches on the FPGA based test fixture. Figure 104 shows an overall view of the test setup used to perform FPGA-based measurements. The 4-bit FCODE [3:0] control nibble can be set to various combinations as explained in Table 23.

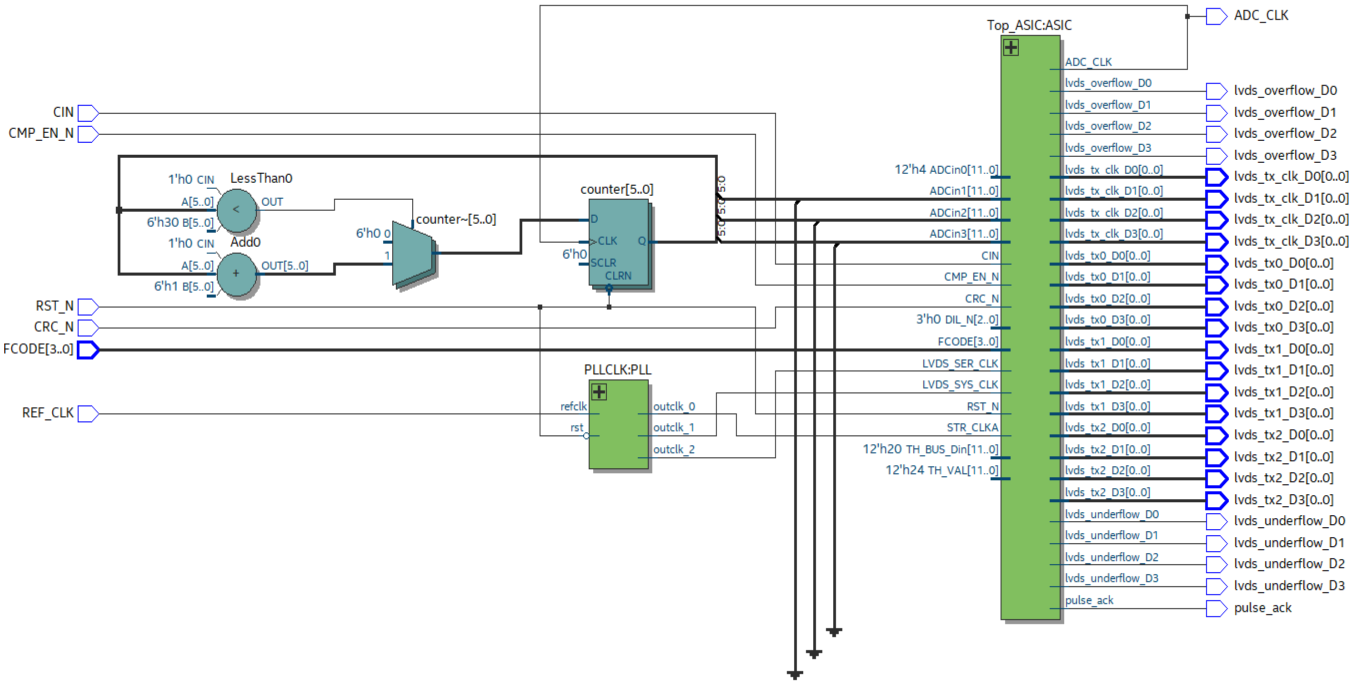


Figure 104: Adopting the RTL view with the Quartus environment to perform FPGA-based measurements.

FCODE [3:0] value	Description
"1100"	Event Readout, zero suppression enabled.
"1111"	Reset FIFO Memory Buffers.
"1000"	Event Readout, zero suppression disabled.
"1011"	Write Pedestal values to internal RAM via input port TH_BUS_Din<11:0>.
"1010"	Write Pedestal values to internal RAM via input ports ADCin0<11:0>, ADCin1<11:0>, ADCin2<11:0>, ADCin3<11:0>.
"0xxx"	Idle State.

Table 23. Description of the various DSRP Function Codes nibble settings FCODE [3:0].

4.2.1 FPGA waveform measurements for Event readout, and Zero suppression

Each DSPR module can process up to 48 analogue input channels using the GASSIPLEX signal conditioning circuitry described in Chapter 2 and as requested by the ALICE collaboration. Event Readout processing is performed in two steps. First, the pedestal and noise of each channel are measured and stored in the DSRP internal memory buffer. Second, the normal event readout process can be initiated in which the unwanted zeros are suppressed using a subtractor circuitry before the acquired data is transmitted serially via a 3-lane Low-Voltage fully (LVDS) differential channels. Zero suppression can be enabled with a FCODE<3:0> value of "1100" or deactivated with an FCODE<3:0> value of "1000". Channel Pedestal and noise values can be configured either through the TH_BUS_Din<11:0> port or ADCin0<11:0> input ports using FCODE<3:0> values of "1011" and "1010" respectively.

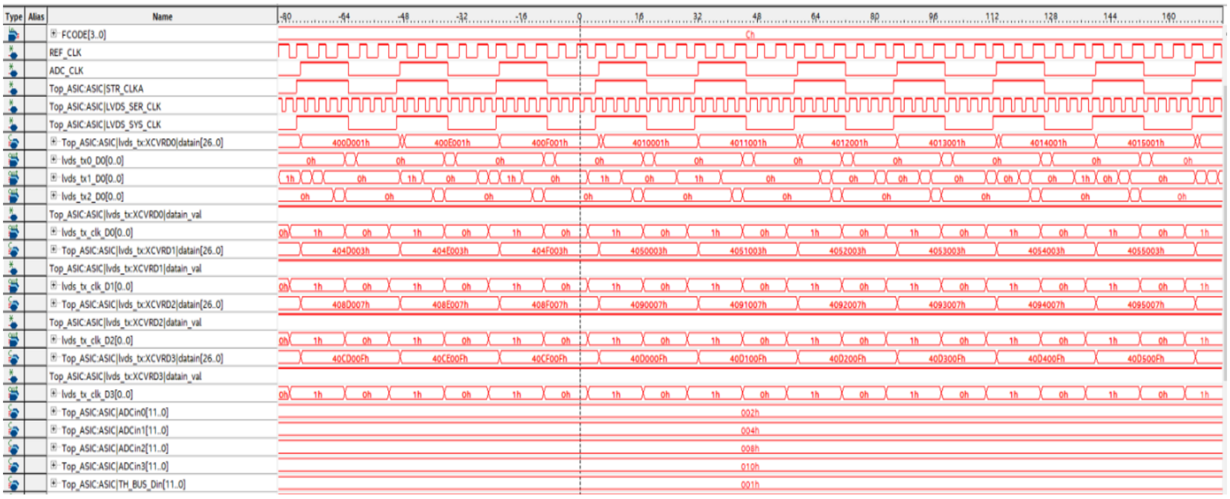
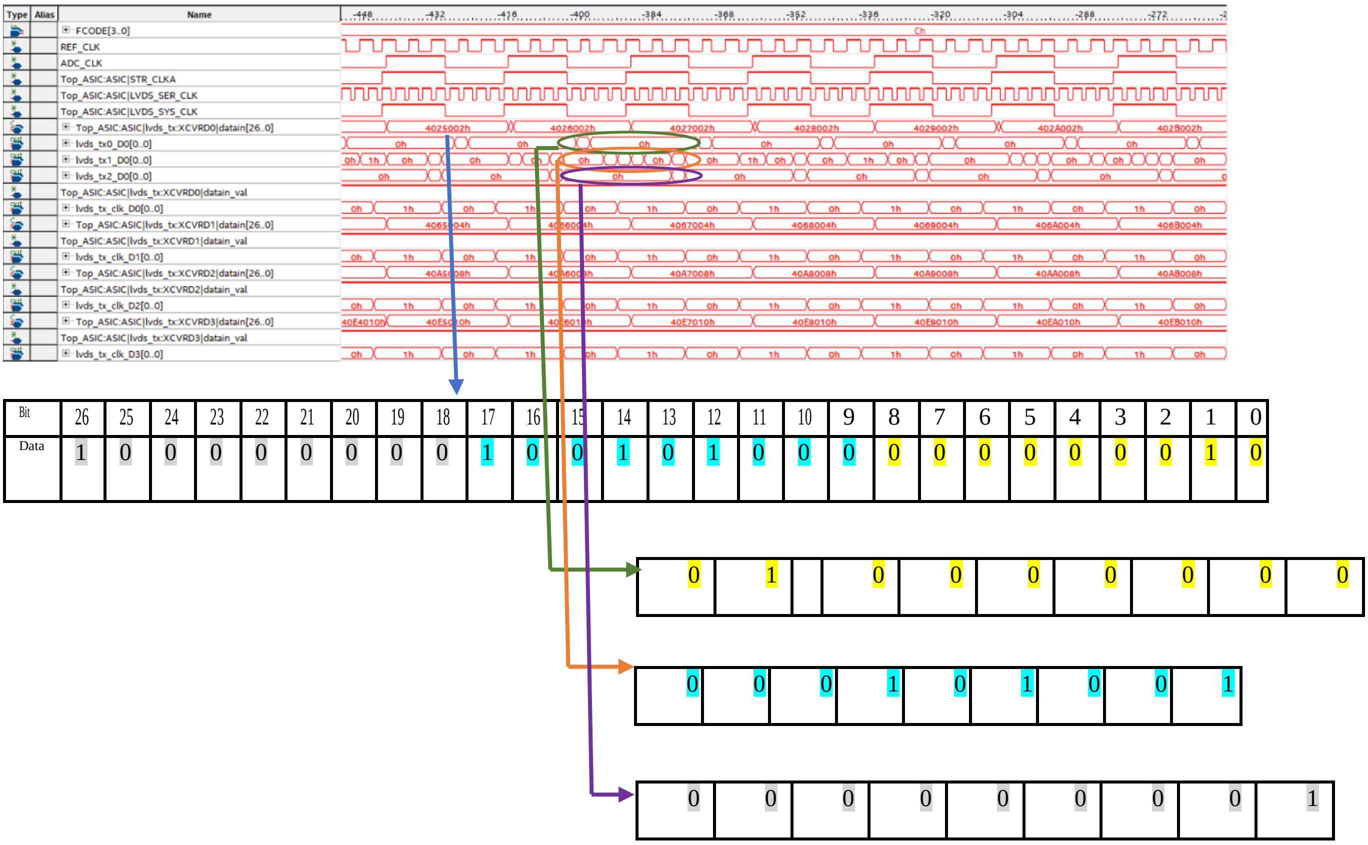

Type
Alias
Name
-80 -64 -48 -32 -16 0 16 32 48 64 80 96 112 128 144 160
FCODE[3..0]
REF_CLK
ADC_CLK
Top_ASIC:ASIC|STR_CLKA
Top_ASIC:ASIC|LVDS_SER_CLK
Top_ASIC:ASIC|LVDS_SYS_CLK
Top_ASIC:ASIC|lvds_tx:XCVRD0|datain[26..0]
lvds_tx0_D0[0..0]
lvds_tx1_D0[0..0]
lvds_tx2_D0[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD0|datain_val
lvds_tx_clk_D0[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD1|datain[26..0]
Top_ASIC:ASIC|lvds_tx:XCVRD1|datain_val
lvds_tx_clk_D1[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD2|datain[26..0]
Top_ASIC:ASIC|lvds_tx:XCVRD2|datain_val
lvds_tx_clk_D2[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD3|datain[26..0]
Top_ASIC:ASIC|lvds_tx:XCVRD3|datain_val
lvds_tx_clk_D3[0..0]
Top_ASIC:ASIC|ADCin0[11..0]
Top_ASIC:ASIC|ADCin1[11..0]
Top_ASIC:ASIC|ADCin2[11..0]
Top_ASIC:ASIC|ADCin3[11..0]
Top_ASIC:ASIC|TH_BUS_Din[11..0]
Ch
400D001h 400E001h 400F001h 4010001h 4011001h 4012001h 4013001h 4014001h 4015001h
0h 1h
404D003h 404E003h 404F003h 4050003h 4051003h 4052003h 4053003h 4054003h 4055003h
408D007h 408E007h 408F007h 4090007h 4091007h 4092007h 4093007h 4094007h 4095007h
40CD00Fh 40CE00Fh 40CF00Fh 40D000Fh 40D100Fh 40D200Fh 40D300Fh 40D400Fh 40D500Fh
002h
004h
008h
010h
001h

Figure 105: Timing diagram for FCODE “1100”, and pedestals set via port TH_BUS_Din<11:0> to a value of 0x0001.

Bit	26:20	19:17	16:12	11:0
Description	ACK<26>/CRC<24:20>	3-bit DSPR Identification value	Channel Address	12-bit ADC Digital Codeword

Table 24. Illustration of 27-bit datain[26:0] frame, transmitted via 3 LVDS lanes.

Figure 105 illustrates a 27-bit data frame datain[26:0] segmented as described in Table 24. Prior to reading events with zero suppression enabled via the FCODE < 3:0 > "1100", the channel memory buffers were loaded with a pedestal and noise value of "0x0000001" via the port TH_BUS_Din[11:0] with the FCODE < 3:0 > "1011". The CIN bit is set low. As can be seen from the signals XCVRD0, XCVRD1, XCVRD2, XCVRD3, each transmitted 12-bit ADC digital codeword has been decreased by the pedestal value compared to the respective digital port values ADCin0, ADCin1, ADCin2 and ADCin3. It can also be observed that LVDS_SYS_ CLK corresponds to nine LVDS_SER_ CLK clock cycles according to the design configuration.



Bit	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data	1	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0

0	1		0	0	0	0	0	0	0

0	0	0	1	0	1	0	0	1

0	0	0	0	0	0	0	0	1

Figure 106: Comparison between data frame datain[26:0], and respective lvds_txx_Dx[0:0] serial lanes.

Figure 106 shows the relationship between the parallel 27-bit input data<26:0> bus and the 3-lane LVDS serializer transmitter links lvds_tx0_Dx[0:0], lvds_tx1_Dx[0:0], and lvds_tx2_Dx[0:0] with respect to the serial clock lvds_tx_clk_D0[0:0]. The LVDS serializer requires two clocks for its operation, namely LVDS_SER_ CLK and LVDS_SYS_ CLK. The parallel LVDS data input is serialized in 3 groups of 9 bits each, so the frequency of LVDS_SER_ CLK should be 9 times higher than that of LVDS_SYS_ CLK clock.

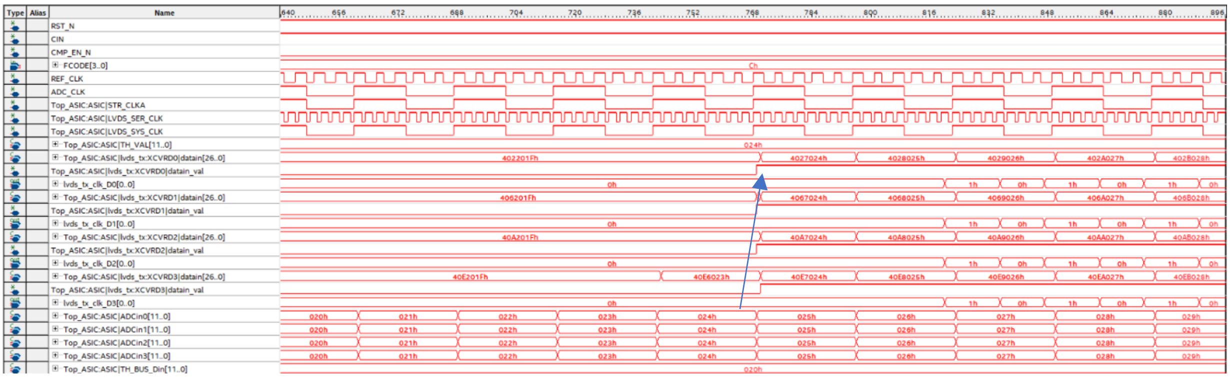

Type
Alias
Name
RST_N
CIN
CMP_EN_N
FCODE[3..0]
REF_CLK
ADC_CLK
Top_ASIC:ASIC|STR_CLKA
Top_ASIC:ASIC|LVDS_SER_CLK
Top_ASIC:ASIC|LVDS_SYS_CLK
Top_ASIC:ASIC|TH_VAL[11..0]
Top_ASIC:ASIC|lvds_tx:XCVRD0|datain[26..0]
Top_ASIC:ASIC|lvds_tx:XCVRD0|datain_val
lvds_tx_clk_D0[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD1|datain[26..0]
Top_ASIC:ASIC|lvds_tx:XCVRD1|datain_val
lvds_tx_clk_D1[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD2|datain[26..0]
Top_ASIC:ASIC|lvds_tx:XCVRD2|datain_val
lvds_tx_clk_D2[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD3|datain[26..0]
Top_ASIC:ASIC|lvds_tx:XCVRD3|datain_val
lvds_tx_clk_D3[0..0]
Top_ASIC:ASIC|ADCin0[11..0]
Top_ASIC:ASIC|ADCin1[11..0]
Top_ASIC:ASIC|ADCin2[11..0]
Top_ASIC:ASIC|ADCin3[11..0]
Top_ASIC:ASIC|TH_BUS_Din[11..0]
Ch
024h
020h
021h
022h
023h
024h
025h
026h
027h
028h
029h

Figure 107: Digital codewords greater than a set threshold 0x24 on port TH_VAL<11:0> are transmitted, on setting CMP_EN.

In this test bench, LVDS_SYS_ CLK is set to 10MHz, while LVDS_SER_ CLK is 90 MHz. The least significant bit (LSB) is transmitted first. The function of the input CMP_EN_N is for data suppression. Only values greater than a specified threshold at TH _VAL < 11:0 >, are transmitted over the LVDS ser connections, suppressing unwanted analogue codewords or noisy channels. The internal DSPR FIFOs are only activated if the ADCin<*x*><11:0> data is greater than 0x24, which is signalled by the datain_val control bit, where *x* represents a value from 0 to 3.

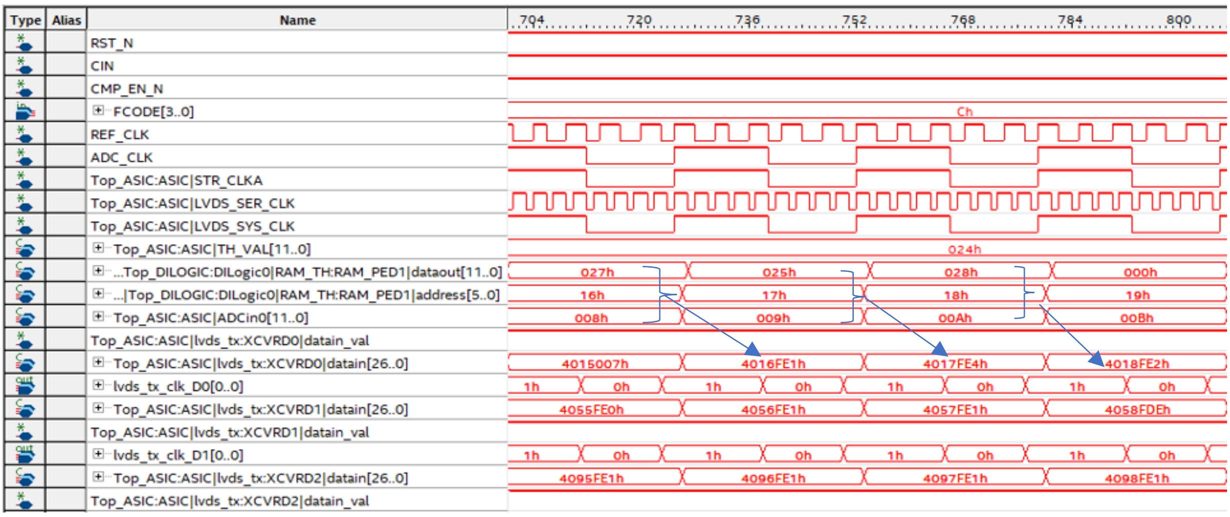

Type
Alias
Name
704
720
736
752
768
784
800
RST_N
CIN
CMP_EN_N
FCODE[3..0]
REF_CLK
ADC_CLK
Top_ASIC:ASIC|STR_CLKA
Top_ASIC:ASIC|LVDS_SER_CLK
Top_ASIC:ASIC|LVDS_SYS_CLK
Top_ASIC:ASIC|TH_VAL[11..0]
...Top_DILOGIC:DILogic0|RAM_TH:RAM_PED1|dataout[11..0]
...|Top_DILOGIC:DILogic0|RAM_TH:RAM_PED1|address[5..0]
Top_ASIC:ASIC|ADCin0[11..0]
Top_ASIC:ASIC|lvds_tx:XCVRD0|datain_val
Top_ASIC:ASIC|lvds_tx:XCVRD0|datain[26..0]
lvds_tx_clk_D0[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD1|datain[26..0]
Top_ASIC:ASIC|lvds_tx:XCVRD1|datain_val
lvds_tx_clk_D1[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD2|datain[26..0]
Top_ASIC:ASIC|lvds_tx:XCVRD2|datain_val
Ch
024h
027h
025h
028h
000h
16h
17h
18h
19h
008h
009h
00Ah
00Bh
4015007h
4016FE1h
4017FE4h
4018FE2h
1h
0h
4055FE0h
4056FE1h
4057FE1h
4058FDEh
4095FE1h
4096FE1h
4097FE1h
4098FE1h

Figure 108: Pedestal suppression measurement result.

As shown in Figure 108, when using the FCODE <3:0> = "1100", the internal pedestal RAM data bus output is subtracted from the ADCin0 <11:0>, subtracting the unwanted pedestal and noise values. Example: analogue channel address 16, pedestal value is equal to 0x27, and ADCin0 < 11:0 > is 0x08, so the result of the pedestal subtraction is 0xFE1, which is transmitted through the LVDS serializer as the full value of 0x4016FE1. A similar process is observed for analogue channels 17 and 18.

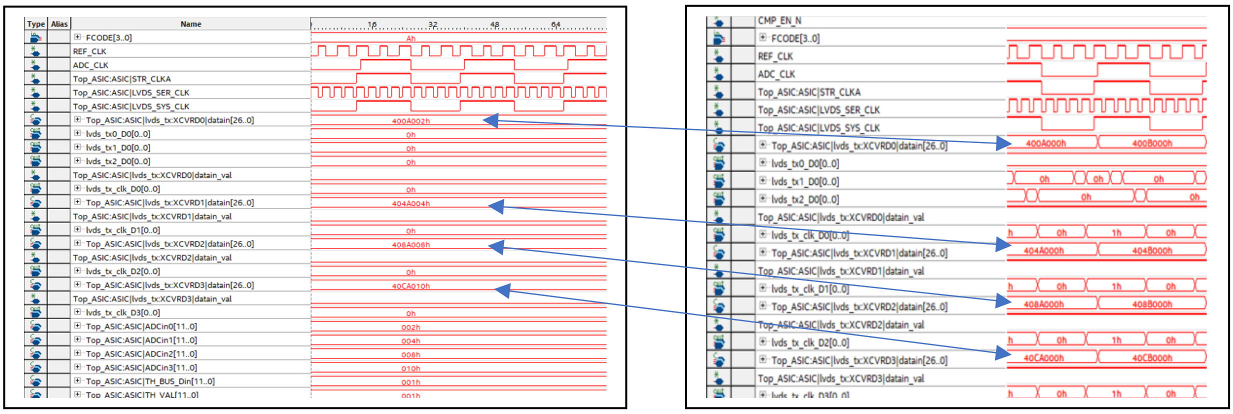

Type
Alias
Name
FCODE[3..0]
REF_CLK
ADC_CLK
Top_ASIC:ASIC|STR_CLKA
Top_ASIC:ASIC|LVDS_SER_CLK
Top_ASIC:ASIC|LVDS_SYS_CLK
Top_ASIC:ASIC|lvds_tx:XCVRD0|datain[26..0]
lvds_tx0_D0[0..0]
lvds_tx1_D0[0..0]
lvds_tx2_D0[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD0|datain_val
lvds_tx_clk_D0[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD1|datain[26..0]
Top_ASIC:ASIC|lvds_tx:XCVRD1|datain_val
lvds_tx_clk_D1[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD2|datain[26..0]
Top_ASIC:ASIC|lvds_tx:XCVRD2|datain_val
lvds_tx_clk_D2[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD3|datain[26..0]
Top_ASIC:ASIC|lvds_tx:XCVRD3|datain_val
lvds_tx_clk_D3[0..0]
Top_ASIC:ASIC|ADCin0[11..0]
Top_ASIC:ASIC|ADCin1[11..0]
Top_ASIC:ASIC|ADCin2[11..0]
Top_ASIC:ASIC|ADCin3[11..0]
Top_ASIC:ASIC|TH_BUS_Din[11..0]
Top_ASIC:ASIC|TH_VAL[11..0]
Ah
400A002h
404A004h
408A008h
40CA010h
002h
004h
008h
010h
001h
CMP_EN_N
400A000h
400B000h
404A000h
404B000h
408A000h
408B000h
40CA000h
40CB000h
0h
1h

Figure 109: a) (Left) Apply FCODE<3:0> = “1010”, followed by (b) FCODE<3:0> = “1100”.

Writing of pedestal and noise values to the internal RAM can also be done through the input ports ADCin0 < 11:0 >, ADCin1 < 11:0 >, ADCin2 < 11:0 >, ADCin3 < 11:0 > using the FCODE < 3:0 > = "1010" as shown in Figure 109. Event readout and suppression can then be performed using FCODE < 3:0 > = "1100" as shown in Figure 109(b).

4.2.2 Design and Verification of Cyclic Redundancy Check

Cyclic Redundancy Check (CRC) technique is used as an error detection technique for the implemented LVDS communication protocol. There are several approaches to implement CRC, and as described in [37], [30] [31] and [38], the parallel CRC implementation method is hardware efficient and suitable for high-speed implementation of VHDL FPGA. The parallel implementation of CRC-5 is used in this work. The degree of the polynomial generator is represented by m and the number of bits to be processed in parallel is represented by w. For the MDART system, m = 5, while w = 21. The design procedure for the parallel system CRC realization as explained in [38] has been adopted using the following *F-matrix*, where p_i represents the coefficients of the generator polynomial $x^5 + x^2 + 1$. With a data width length of w = 21, and m = 5, then the following 21 matrices were manually calculated and derived.

$$F = \begin{bmatrix} p_{m-1} & 1 & 0 & \cdots & 0 \\ p_{m-2} & 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ p_1 & 0 & 0 & \cdots & 1 \\ p_0 & 0 & 0 & \cdots & 0 \end{bmatrix}$$

and

$$F^{i} = \left[F^{i-1} \oplus \begin{bmatrix} p_{m-1} \\ \cdots \\ p_{1} \\ p_{0} \end{bmatrix} \middle| the\ first\ m-1\ columns\ of\ F^{i-1} \right] where\ i\ ranges\ from\ 2\ to\ w\,.$$

$$F^{1}=\begin{bmatrix} 0&1&0&0&0 \\ 0&0&1&0&0 \\ 1&0&0&1&0 \\ 0&0&0&0&1 \\ 1&0&0&0&0 \end{bmatrix} \quad F^{2}=\begin{bmatrix} 0&0&1&0&0 \\ 1&0&0&1&0 \\ 0&1&0&0&1 \\ 1&0&0&0&0 \\ 0&1&0&0&0 \end{bmatrix} \quad F^{3}=\begin{bmatrix} 1&0&0&1&0 \\ 0&1&0&0&1 \\ 1&0&1&0&0 \\ 0&1&0&0&0 \\ 0&0&1&0&0 \end{bmatrix} \quad F^{4}=\begin{bmatrix} 0&1&0&0&1 \\ 1&0&1&0&0 \\ 1&1&0&1&0 \\ 0&0&1&0&0 \\ 1&0&0&1&0 \end{bmatrix} \quad F^{5}=\begin{bmatrix} 1&0&1&0&0 \\ 1&1&0&1&0 \\ 0&1&1&0&1 \\ 1&0&0&1&0 \\ 0&1&0&0&1 \end{bmatrix}$$

$$F^{6}=\begin{bmatrix} 1&1&0&1&0 \\ 0&1&1&0&1 \\ 0&0&1&1&0 \\ 0&1&0&0&1 \\ 1&0&1&0&0 \end{bmatrix} \quad F^{7}=\begin{bmatrix} 0&1&1&0&1 \\ 0&0&1&1&0 \\ 1&0&0&1&1 \\ 1&0&1&0&0 \\ 1&1&0&1&0 \end{bmatrix} \quad F^{8}=\begin{bmatrix} 0&0&1&1&0 \\ 1&0&0&1&1 \\ 1&1&0&0&1 \\ 1&1&0&1&0 \\ 0&1&1&0&1 \end{bmatrix} \quad F^{9}=\begin{bmatrix} 1&0&0&1&1 \\ 1&1&0&0&1 \\ 1&1&1&0&0 \\ 0&1&1&0&1 \\ 0&0&1& &0 \end{bmatrix} F^{10}=\begin{bmatrix} 1&1&0&0&1 \\ 1&1&1&0&0 \\ 1&1&1&1&0 \\ 0&0&1&1&0 \\ 1&0&0&1&1 \end{bmatrix}$$

$$F^{11}=\begin{bmatrix} 1&1&1&0&0 \\ 1&1&1&1&0 \\ 1&1&1&1&1 \\ 1&0&0&1&1 \\ 1&1&0&0&1 \end{bmatrix} F^{12}=\begin{bmatrix} 1&1&1&1&0 \\ 1&1&1&1&1 \\ 0&1&1&1&1 \\ 1&1&0&0&1 \\ 1&1&1&0&0 \end{bmatrix} F^{13}=\begin{bmatrix} 1&1&1&1&1 \\ 0&1&1&1&1 \\ 0&0&1&1&1 \\ 1&1&1&0&0 \\ 1&1&1&1&0 \end{bmatrix} F^{15}=\begin{bmatrix} 0&0&1&1&1 \\ 0&0&0&1&1 \\ 1&0&0&0&1 \\ 1&1&1&1&1 \\ 0&1&1&1&1 \end{bmatrix} F^{16}=\begin{bmatrix} 0&0&0&1&1 \\ 1&0&0&0&1 \\ 1&1&0&0&0 \\ 0&1&1&1&1 \\ 0&0&1&1&1 \end{bmatrix}$$

$$F^{17}=\begin{bmatrix} 1&0&0&0&1 \\ 1&1&0&0&0 \\ 0&1&1&0&0 \\ 0&0&1&1&1 \\ 0&0&0&1&1 \end{bmatrix} F^{18}=\begin{bmatrix} 1&1&0&0&0 \\ 0&1&1&0&0 \\ 1&0&1&1&0 \\ 0&0&0&1&1 \\ 1&0&0&1&1 \end{bmatrix} F^{19}=\begin{bmatrix} 0&1&1&0&0 \\ 1&0&1&1&0 \\ 1&1&0&1&1 \\ 1&0&0&0&1 \\ 1&1&0&0&0 \end{bmatrix} F^{20}=\begin{bmatrix} 1&0&1&1&0 \\ 1&1&0&1&1 \\ 1&1&1&0&1 \\ 1&1&0&0&0 \\ 0&1&1&0&0 \end{bmatrix} F^{21}=\begin{bmatrix} 1&1&0&1&1 \\ 1&1&1&0&1 \\ 0&1&1&1&0 \\ 0&1&1&0&0 \\ 1&0&1&1&0 \end{bmatrix}$$

Since the system is time-invariant this means that the state equation of the system can be represented by,

$$X^{'} = F^{w} \otimes X \oplus D$$

where X' and X represent the next and present system states. D represents the 21-bit input data width. As shown in Table 25, the i-th row and j-th column of the F^{W} matrix represent enable switches for data inputs. Matrix product is represented by $\otimes$, while matrix addition is performed using XOR operation $\oplus$.

Therefore, taking in consideration the F-matrices shown above, and with reference to the following Matrix Table 25, X' can be expanded for the implementation of a CRC-5 generator circuit as described below.

Bit	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X'_4	1	1	0	1	1	0	0	0	1	1	1	1	1	0	0	1	1	0	1	0	0
X'_3	1	1	1	0	1	1	0	0	0	1	1	1	1	1	0	0	1	1	0	1	0
X'_2	0	1	1	1	0	1	1	0	0	0	1	1	1	1	1	0	0	1	1	0	1
X'_1	0	1	1	0	0	0	1	1	1	1	1	0	0	1	1	0	1	0	0	1	0
X'_0	1	0	1	1	0	0	0	1	1	1	1	1	0	0	1	1	0	1	0	0	1

Table 25: Matrix table for CRC-5 with w =21.

$\boldsymbol{X'_0}$ = *d(20)* ⊕ *d(18)* ⊕ *d(17)* ⊕ *d(13)* ⊕ *d(12)* ⊕ *d(11)* ⊕ *d(10)* ⊕ *d(9)* ⊕ *d(6)* ⊕ *d(5)* ⊕ *d(3)* ⊕ *d(0)* ⊕ *X(1)* ⊕ *X(2)* ⊕ *X(4),*

$\boldsymbol{X'_1}$ = *d(19)* ⊕ *d(18)* ⊕ *d(14)* ⊕ *d(13)* ⊕ *d(12)* ⊕ *d(11)* ⊕ *d(10)* ⊕ *d(7)* ⊕ *d(6)* ⊕ *d(4)* ⊕ *d(1)* ⊕ *X(2)* ⊕ *X(3),*

$\boldsymbol{X'_2}$ = *d(19)* ⊕ *d(18)* ⊕ *d(17)* ⊕ *d(15)* ⊕ *d(14)* ⊕ *d(10)* ⊕ *d(9)* ⊕ *d(8)* ⊕ *d(7)* ⊕ *d(6)* ⊕ *d(3)* ⊕ *d(2)* ⊕ *d(0)* ⊕ *X(1)* ⊕ *X(2)* ⊕ *X(3),*

$\boldsymbol{X'_3}$ = *d(20)* ⊕ *d(19)* ⊕ *d(18)* ⊕ *d(16)* ⊕ *d(15)* ⊕ *d(11)* ⊕ *d(10)* ⊕ *d(9)* ⊕ *d(8)* ⊕ *d(7)* ⊕ *d(4)* ⊕ *d(3)* ⊕ *d(1)* ⊕ *X(0)* ⊕ *X(2)* ⊕ *X(3)* ⊕ *X(4),*

$\boldsymbol{X'_4}$ = *d(20)* ⊕ *d(19)* ⊕ *d(17)* ⊕ *d(16)* ⊕ *d(12)* ⊕ *d(11)* ⊕ *d(10)* ⊕ *d(9)* ⊕ *d(8)* ⊕ *d(5)* ⊕ *d(4)* ⊕ *d(2)* ⊕ *X(0)* ⊕ *X(1)* ⊕ *X(3)* ⊕ *X(4).*

The above XOR-based Boolean expressions have been implemented using VHDL, contributing to the realisation of a parallel CRC-5 generator circuit with a 21-bit data message as input (see Figure 110). The CRC-5 generator circuit is initialised with a value of "10011", which is also applied to the receiver circuit. The generated 5-bit word CRC is appended to the transmitted data word. The receiver consists of a similar CRC generator circuit to compare the received checksum with the one generated in the receiver.

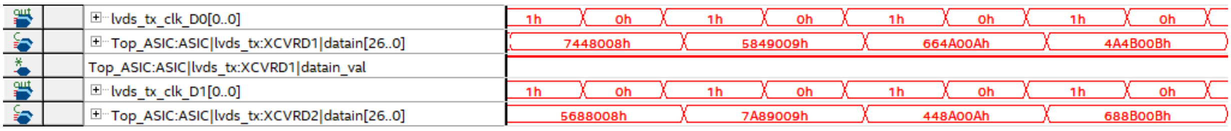

lvds_tx_clk_D0[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD1|datain[26..0]
Top_ASIC:ASIC|lvds_tx:XCVRD1|datain_val
lvds_tx_clk_D1[0..0]
Top_ASIC:ASIC|lvds_tx:XCVRD2|datain[26..0]
1h
0h
1h
0h
1h
0h
1h
0h
7448008h
5849009h
664A00Ah
4A4B00Bh
5688008h
7A89009h
448A00Ah
688B00Bh

Figure 110: CRC -5 word appended to a 21-bit data input message and transmitted via SERDES using a 27-bit data frame.

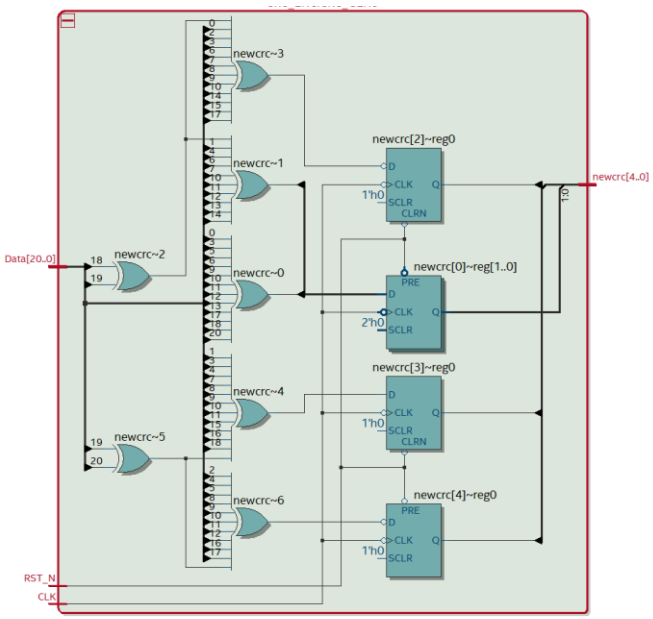


Figure 111: RTL view of CRC-5 generator circuit for a 21-bit input data message.

The implemented FPGA-based CRC generator circuit requires only 8 Look-up Tables, and 5 flip-flop based registers, and requires only one clock cycle to generate the required CRC word.

4.3.1 Overall Estimated Simulated Power Consumption

The functional verification described till now has been performed for a Cyclone V GX device, which is the same FPGA device currently being used in CPV FEE system. Using an average logic clocked toggle rate of 3.8 million transitions per second, the overall system early thermal power estimation for the MDART system architecture, is 344.13 mW, which meets the required power budget of 940mW. Further, 592 ALMs, 822 registers and 10,994 memory bits have been used.

4.4 ASIC VHDL Synthesis

The synthesis stage follows the VHDL/Verilog stage of functional verification and simulation, in which the behavioural VHDL file was extracted and converted to a structural file using X-FAB XH018 0.18µ CMOS standard cell libraries [69]. The physical design is a circuit representation (or netlist) that is converted into a geometric representation, the layout. This layout is created by converting each logical component (cells, macros, gates, transistors) into a geometric representation (specific shapes in multiple layers) that perform the intended logical function of the corresponding component. The interconnections between the various components are represented as geometric patterns, typically lines in multiple layers. The physical design of a chip begins with the partitioning process, which considers many factors such as the size of the blocks, the number of blocks, and the number of interconnections between the blocks. This is followed by floor space planning, which deals with the selection of suitable layout alternatives for each block as well as for the entire chip. Appropriate placement is then performed to find a minimum area layout for the blocks that allows the interconnections between the blocks to be completed while meeting the required performance specifications. Clock tree synthesis (CTS) is then performed. Clock tree synthesis (CTS) was performed using the Concurrent CTS and timing-optimization engine (CCOpt) available in the Innovus system implementation tool. The CCOpt algorithms focus on optimising the entire logic chain rather than just the critical paths. The purpose of clock tree synthesis is to create a balanced clock tree, optionally with useful skew for timing closure and post- CTS optimization.

Clock trees are created in two stages:

- Stage 1 builds a load balanced Design Rule Check (DRC) clean clock tree,
- Stage 2 performs clock tree optimisation, be performing cell sizing, relocation, and buffer insertion so to try and achieve the necessary clock targets,

After CTS process signal routing process is done so to complete the interconnections between blocks according to a specific netlist. The routing space not occupied by the blocks is partitioned into rectangular regions known as switchboxes or channels. The objective of the router is to complete all circuit connections using the minimum possible wire length via the channel and switch boxes. Routing is performed in two stages, referred to as the Global Routing and Detailed Routing phases. Finally, the physical layout of a chip must not only meet the geometric requirements but also meet the timing constraints or closure.

In other words, the VHDL program is therefore synthesized into a digital logic circuit consisting of a collection of cells, which behaves in the same manner. The structural file consists of structural instantiations of cells from one or more X-FAB XH018 0.18µ CMOS technology libraries. This generated synthesised structural file is the Verilog equivalent of a schematic circuit, which is later used by back-end tools such as Cadence Innovus so to place the cell libraries on the chip and accordingly route the necessary connections. The synthesis tool used for synthesizing MDART device is Cadence Genus Synthesis Solution, which optimises the timing paths so to meet the required Synopsys Design Constraints (SDC) and elaborates VHDL behavioural design into a netlist in terms of generic logic cells.

The scripted list of behavioural VHDL files and X-FAB XH018 0.18µ CMOS Technology cell library files used to generate the Verilog Synthesis Model for the complete MDART chip is shown in Figure 112 below. Additionally, the used setup and hold times are given. The digital logic and IO liberty library files contain a description of the X-FAB xh018 LPMOS digital logic and I/O Pad Cell libraries. Both libraries can operate at a single supply voltage of 3.3V ± 10%. This IO library contains IO cells, which have been used to drive signals on and off the ASIC chip. Theses libraries have been selected because MDART chip is expected to operate at a Logic Process Voltage Temperature (PVT) worst case condition of not more than 85^{o}C and at a supply Voltage of 3.0 V. Additionally, both the digital and IO libraries can be interfaced together since they are characterised to operate at the selected core voltage of 3.3 V.

```
set_attribute library [list D_CELLS_3V_LPMOS_slow_3_00V_85C.lib
IO_CELLS_C3V_LPMOS_UPF_slow_3_00V_3_00V_85C.lib ]
set_attribute information_level 6
set myFiles [ list PCK_CRC5_D21.vhd Top_ASIC.vhd Top_DILOGIC.vhd fifo_async.vhd
CRC_ENC.vhd fifo_sync.vhd pipeline_reg.vhd voter.vhd lvds_tx.vhd lvds_flow_error.vhd
lvds_serializer.vhd lvds_serializer_n.vhd lvds_obuf.vhd lvds_oclk.vhd comp_N.vhd
FADD_CLA.vhd MUX.vhd RAM_TH.vhd subtractor.vhd]

create_clock -name STR_CLKA -period 100.0 -waveform {0.0 50.0} [get_ports STR_CLKA]

set_input_transition -max 2.0 [get_ports STR_CLKA]
set_input_transition -min 0.1 [get_ports STR_CLKA]
set_input_delay -clock STR_CLKA  -add_delay 10.0 [get_ports  ADCin*]
set_input_delay -clock STR_CLKA  -add_delay 10.0 [get_ports  TH_BUS_Din]
set_clock_uncertainty -setup 10.0  [get_clocks STR_CLKA]
set_clock_uncertainty -hold  0.6  [get_clocks STR_CLKA]
set_clock_latency  3.5 [get_clocks STR_CLKA]
set_driving_cell -lib_cell ICPC [all_inputs]
set_load 0.4 [all_outputs]
```

Figure 112: Configuration Settings for Genus Synthesis tool.

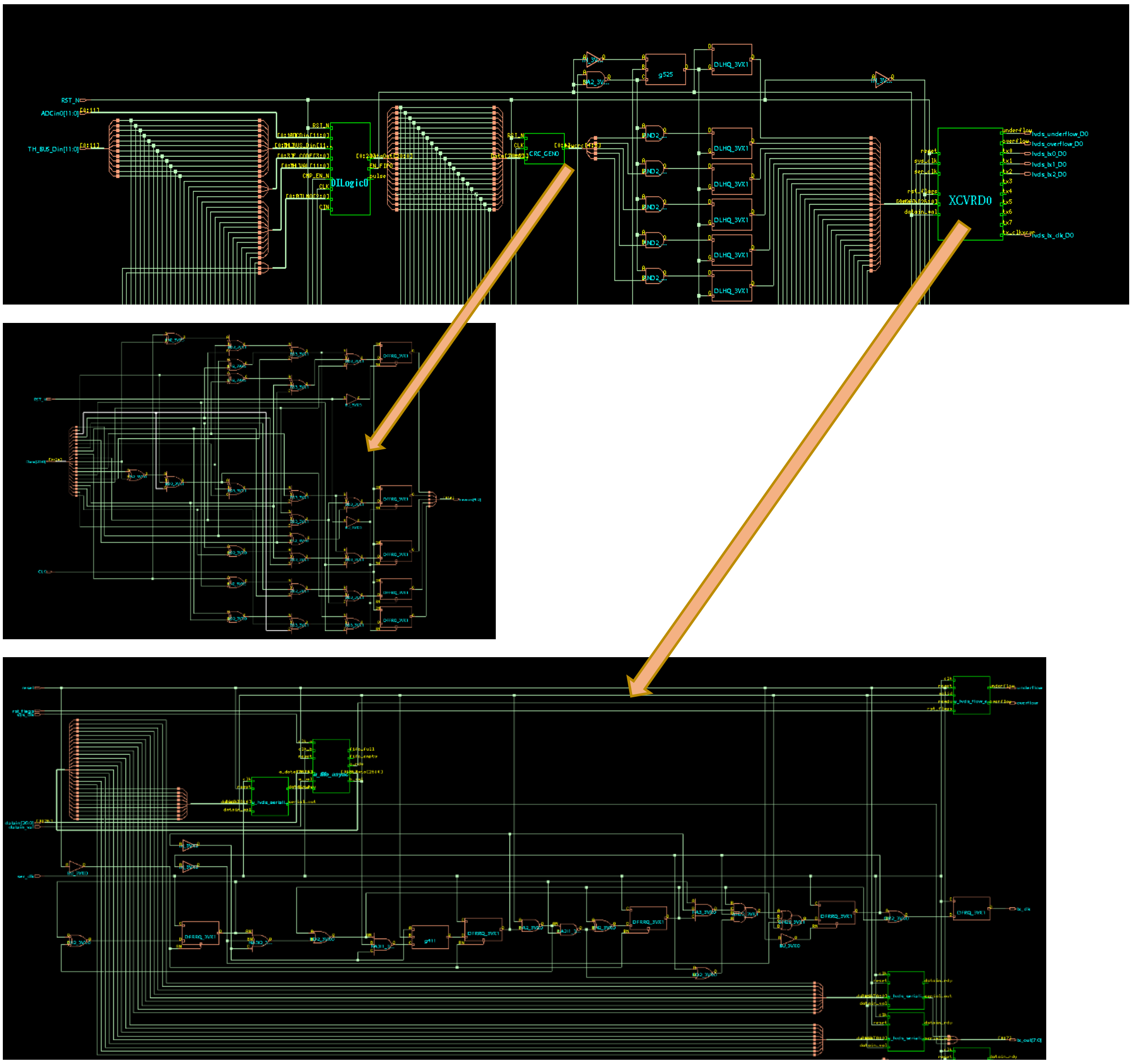


Figure 113: Schematic View of Synthesized Verilog File for one DSRP (DiLogic0), CRC and LVDS transmitter module.

4.5 ASIC Layout and Package Considerations

The physical implementation for the proposed MDART architecture has been implemented in Cadence using the X-FAB XH018 0.18μ CMOS E-FLASH (MET3, MET4, METMID, METTHK) technology and Innovus System Implementation tool. The complete chip layout is shown in Figure 105 and occupies a total area of *3.125 x 3.125 mm^2*. As highlighted in Figure 115 the MDART ASIC chip consists of four LVDS transmitters, four DSRPs and four CRC modules with an integrated digital controller for system operation. All modules have managed to fit on a silicon base are of 10 mm^2, thus limiting the cost to not more than Eur 15,250 without the need of extra pins or extra area. Package CERQUAD FP 160 (CQFP), has been selected, of which 144 out of 160 pins have been used for power, ground, digital input, and output pins as described below:

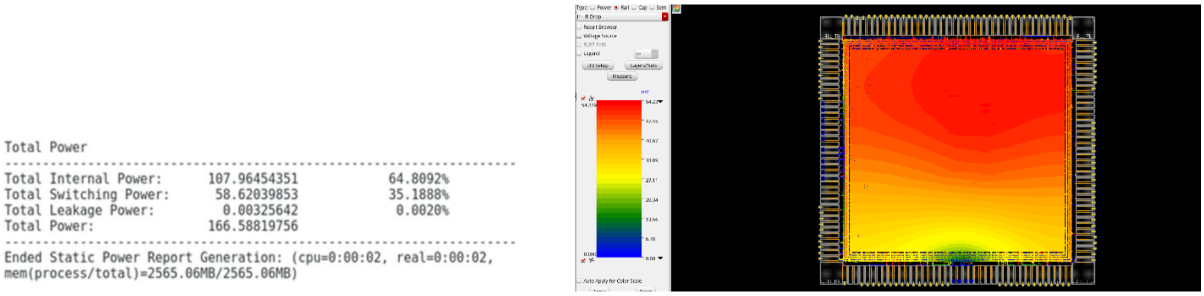
- 12 ground pins (Three GNDR, Three GNDO and Six GNDI)
- 13 pad supply pins (4 VDDR, 4 VDDO, 5 VDD33), to power the digital I/O driver circuits
- 5 VDDI Supply pins to isolate and power the supply core,
- 4 DataIn <3:0> ports, where each port consists of 12 digital inputs
- 2 digital input ports TH_BUS_Din<11:0> and TH_VAL<11:0> with 12 pins with for pedestal configuration
- 3 digital input clock pins STR_CLKA, LVDS_SYS_CLK and LVDS_SER_CLK
- 4 digital output Gigabit transmitter lanes, where each lane consists of three pins
- 5 digital output pins for monitoring serial transmitter overflow operation (one reserved)
- 5 digital output pins for monitoring serial transmitter underflow operation (one reserved)
- 4 digital output transmitter clock pins for serial data synchronisation
- 1 digital output for data synchronisation and acknowledgment (pulse_ack)
- 1 digital input pin labelled CRC_N
- 3 digital inputs for Reset and optionally enable pedestal subtraction operation (CMP_EN, CIN_N)
- 1 digital output pin to optionally clock external ADC modules.
- 3 digital input pins DIL_N<2:0> to optionally configure the number of analogue input words or channels per DSPRs module
- 4 Digital input pins to configure MDART mode of operation

X-FAB XH018 standard digital I/O libraries are available for multi supply voltage chips, and with I/O supply voltage and chip core supply voltage at different levels. Such supply voltage levels can vary from a minimum of 1.8 V and a maximum of 3.3 V±10%. The Non-Inverting CMOS Input Buffer pad called ICPC has been used for digital inputs. The BT4PC tri-state output buffer with a drive strength capability of 4mA at 3.3V, has been used to drive output pins. The maximum driving current capability for the VDDO core power supply pins is 37 mA, at a supply maximum supply voltage of 3.6V [32]. According to the below static power report, when MDART device is clocked at 200 MHz, the total power consumption is 166 mW, hence, the amount of power pads the ASIC design requires has been calculated as follows:

$$Current\ (m\ A) = \frac{Power\ (W)}{Volatge(V)} = \frac{0.166\ W}{3.3\ V} = 50\ .5\ mA$$

Considering a 10% excess margin => (0.0505 x 0.1) + 0.0505 = 55.5 mA.

Therefore, the total number of required pads to power the core is two, while the current required to simultaneously drive all the 26 digital output pins is 0.104 A (26 x 4 mA). Therefore, based on the above calculations 4 groups of power supplies have been used. These four groups are linked to four VDDR, four VDDO, and five VDD33, power supply pins have been used to power the digital I/O driver and buffer circuits. Five VDDI supply pins were used to isolate and power the supply core.



```
Total Power
----------------------------------------------------------------------
Total Internal Power:      107.96454351           64.8092%
Total Switching Power:      58.62039853           35.1888%
Total Leakage Power:         0.00325642            0.0020%
Total Power:               166.58819756
----------------------------------------------------------------------
Ended Static Power Report Generation: (cpu=0:00:02, real=0:00:02,
mem(process/total)=2565.06MB/2565.06MB)
```

Figure 114: Basic static power consumption (166 mW) at clock frequency of 200 MHz. On the right there is a plot of IR drop across MDART ASIC, with the top region contributing to the highest IR drop of 50.23 mV.

The total simulated switching power consumption for the MDART chip when operating at a supply voltage of 3.3V and clocked at a frequency of 50 MHz is 58mW. This is equivalent to approximately at least four-fold decrease in power requirements needed for processing sequentially 192 analogue channels using four CPV DILOGIC chip, each consuming approximately 60 mW, when clocked at a maximum possible frequency of 10 MHz and operating at a typical supply voltage of 5 V. As shown in Figure 114 the highest reported IR drop is estimated to be 50.23 mV, which does not contribute to deterioration in timings.

Figure 116 shows an example of the reported positive Worst Negative Slack (WNS), a positive slack of 5.315 ns hold time, setup time of 68.7 ns, and with a Total Negative slack (TNS) of 0ns, thus meeting the design timing closure requirements at post-route stage, thus giving enough margin to operate the integrated DSRP and the required external ADC modules at typical clock frequency of 10 MHz.

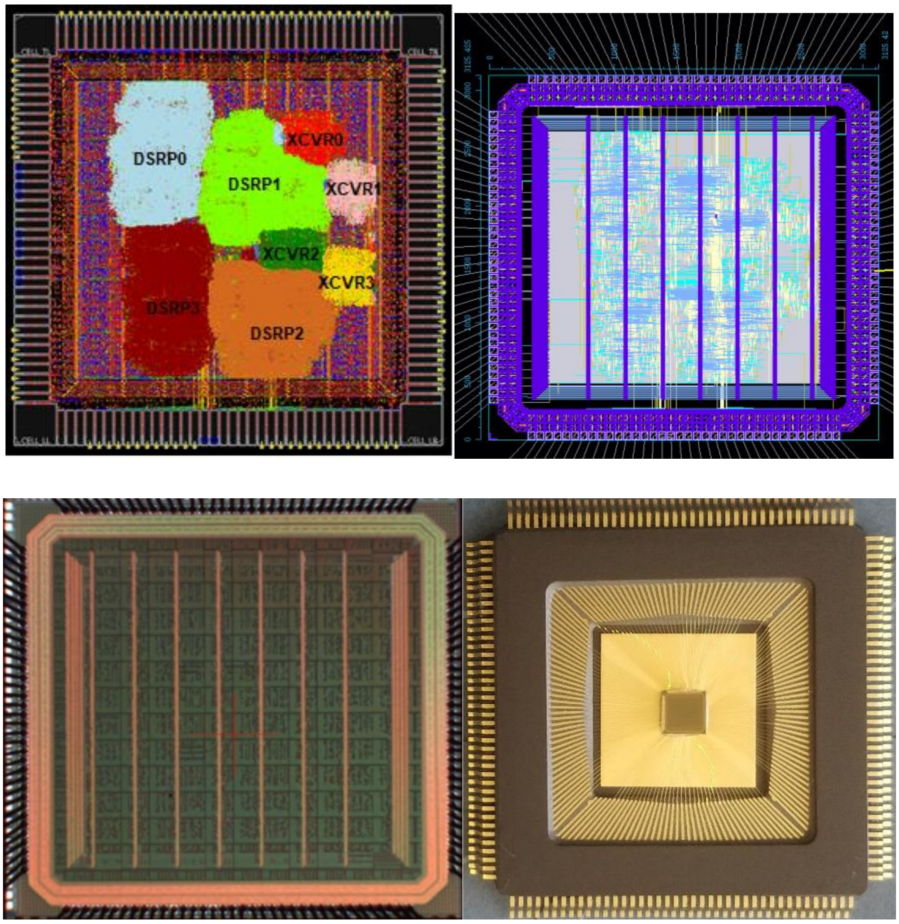
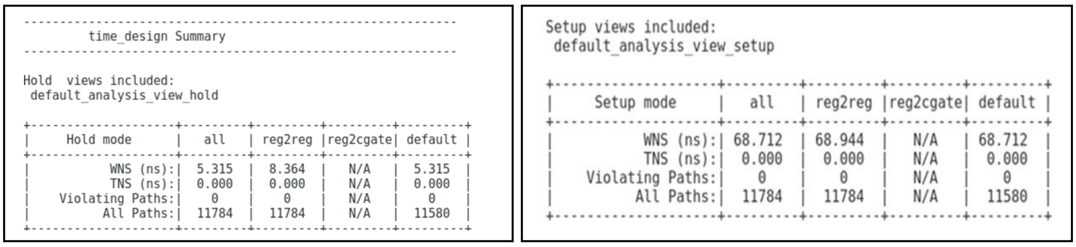


Figure 115: Chip Layout for proposed MDART device with dimensions (3.125 x 3.125) mm^2.

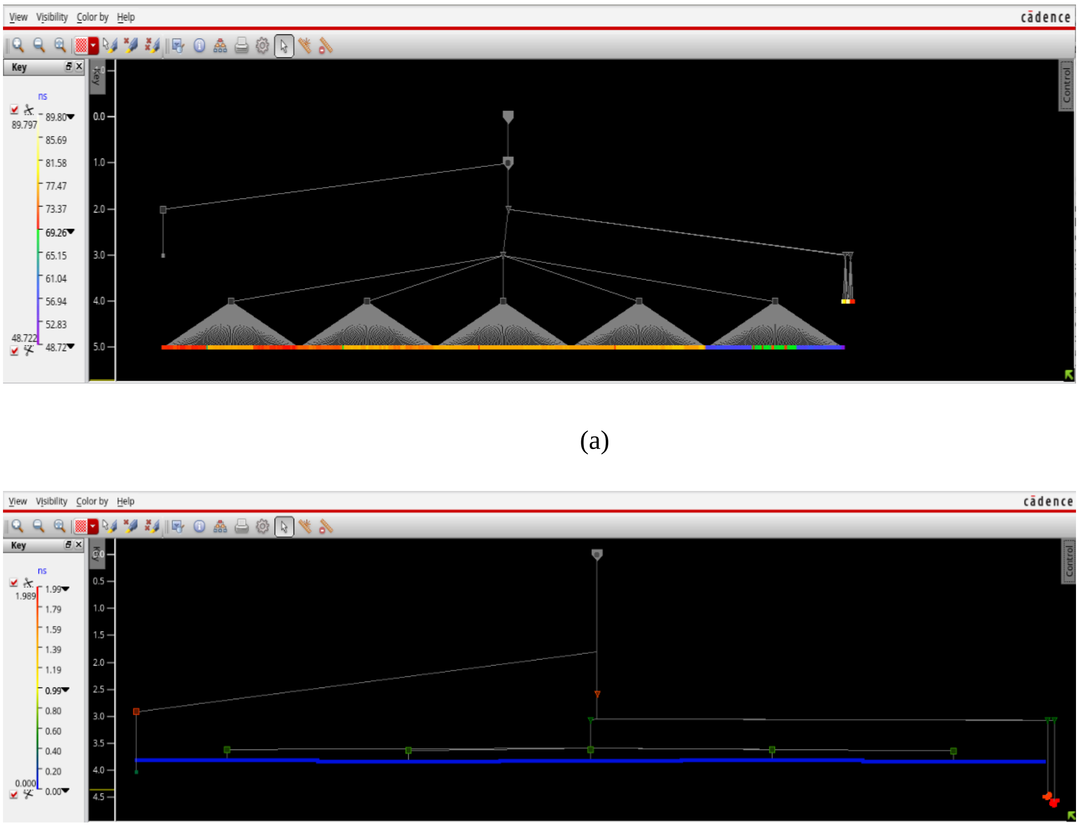


```
---------------------------------------------------------
        time_design Summary
---------------------------------------------------------

Hold  views included:
 default_analysis_view_hold

+--------------------+---------+---------+---------+---------+
|     Hold mode      |   all   | reg2reg |reg2cgate| default |
+--------------------+---------+---------+---------+---------+
|           WNS (ns):|  5.315  |  8.364  |   N/A   |  5.315  |
|           TNS (ns):|  0.000  |  0.000  |   N/A   |  0.000  |
|    Violating Paths:|    0    |    0    |   N/A   |    0    |
|          All Paths:|  11784  |  11784  |   N/A   |  11580  |
+--------------------+---------+---------+---------+---------+
```

```
Setup views included:
 default_analysis_view_setup

+--------------------+---------+---------+---------+---------+
|     Setup mode     |   all   | reg2reg |reg2cgate| default |
+--------------------+---------+---------+---------+---------+
|           WNS (ns):| 68.712  | 68.944  |   N/A   | 68.712  |
|           TNS (ns):|  0.000  |  0.000  |   N/A   |  0.000  |
|    Violating Paths:|    0    |    0    |   N/A   |    0    |
|          All Paths:|  11784  |  11784  |   N/A   |  11580  |
+--------------------+---------+---------+---------+---------+
```

Figure 116: Setup and hold time views, after performing Post-CTS and Post Route Timing analysis

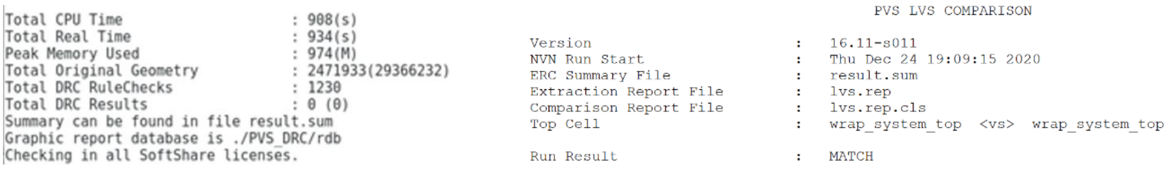
During CTS a maximum skew target of 800 ps has been set. CTS layers of buffers will be inserted by CCOpt engine so to provide a reasonable drive and interconnect load at each stage with minimum skew and delay. The selected CTS Buffers BU_3VX1, BU_3VX2, BU_3VX3, BU_3VX4, BU_3VX6, BU_3VX8, BU_3VX16, DLY1_3VX1, DLY2_3VX1, DLY4_3VX1, DLY8_3VX1, and inverter CTS buffer cells IN_3VX0, IN_3VX1, IN_3VX2, IN_3VX3, IN_3VX4, IN_3VX6, IN_3VX8, IN_3VX16 have been used. As shown in Figure 107, the generated clock tree for the MDART device has three levels, consisting of a root cell (Level 0), distribution cell (Level 1, 2 and 3), and Leaf cells.



(a)

(b)

Figure 117: (a) Clock Tree Synthesis with a (a) setup slack time of 75ns, and (b) transition time of 0.25 ns.



```
Total CPU Time                    : 908(s)
Total Real Time                   : 934(s)
Peak Memory Used                  : 974(M)
Total Original Geometry           : 2471933(29366232)
Total DRC RuleChecks              : 1230
Total DRC Results                 : 0 (0)
Summary can be found in file result.sum
Graphic report database is ./PVS_DRC/rdb
Checking in all SoftShare licenses.
```

```
                                    PVS LVS COMPARISON

Version                         :    16.11-s011
NVN Run Start                   :    Thu Dec 24 19:09:15 2020
ERC Summary File                :    result.sum
Extraction Report File          :    lvs.rep
Comparison Report File          :    lvs.rep.cls
Top Cell                        :    wrap_system_top  <vs>  wrap_system_top

Run Result                      :    MATCH
```

Figure 118: (a) DRC Result Report (left), and (b) Layout versus Schematic (LVS) comparison match report (Right).

As reported in Figure 108 both the Design Rule Check (DRC) and Layout versus Schematic (LVS) comparison passed and matched.

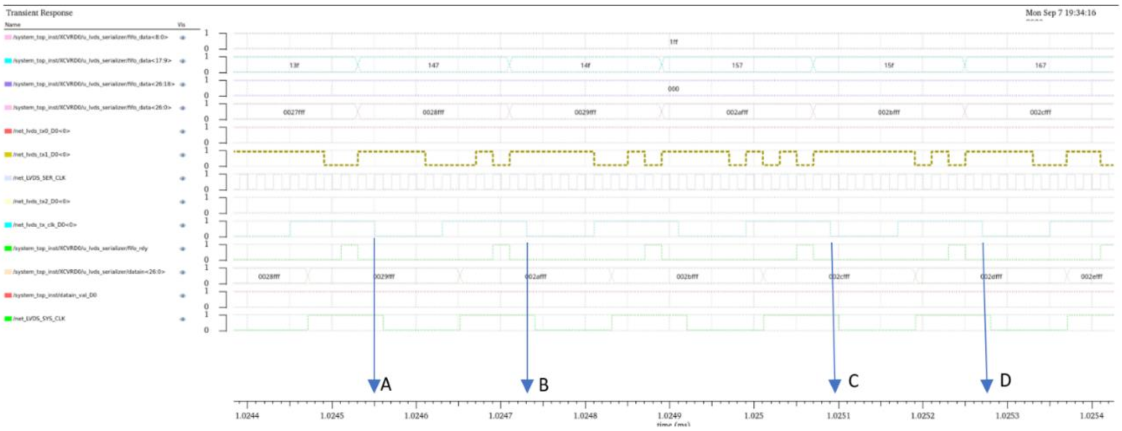


Figure 119: (a) Timing diagram data Serializer output after post-Synthesis phase (10µs/division).

Before proceeding to Sign-off stage, post-Synthesis simulation is done. Figure 119 illustrates an instance of the post-Synthesis simulation results, where the transmitted value between lvds_tx_clk clock cycles indicated by points “A” and “B”, equal to “741”, matches the correct sequence “147” inputted to transmitter SERDES module via FIFO with the Most Significant Bit (MSB) transmitted first. Again, one can notice the value outputted between points “C” and “D” equal to “F51” which matches the correct sequence “15F”. A Standard Delay Format (SDF) file is generated at the end of post layout simulation. The Kyocera CQFP160 package with a body footprint of 28 ×28 mm^2 was chosen for packing the ASIC device. This package was selected due to the large number of input and output pins as described in Table 26. The complete bonding diagram is shown in Figure 120.

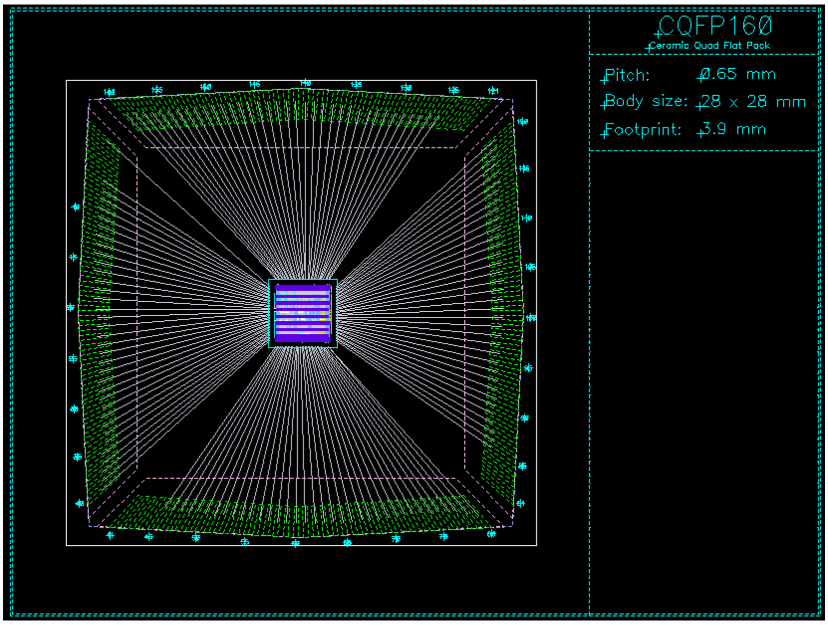


Figure 120: ASIC CQFP160 lead package and die bonding diagram.

Port/Pin Name	Pin Number	Pin/Port Functionality
lvds_underflow_D4	1	Reserved
lvds_overflow_D4	2	Reserved
VDDR	3	3.3V VDDR Supply pin
Datain3[11:0]	4-15	Digital Input Port
VDDI	16-17	3.3V VDDI Supply pin
VDDR	18-19	3.3V VDDR Supply pin
VDDO	20-21	3.3V VDDO Supply pin
DataIn2[10]	22	DSRP2 Digital Input
DataIn2[11]	23	DSRP2 Digital Input
DataIn2[9:0]	24-33	DSRP2 Digital Input
DataIn2 [11]	34	DSRP2 Digital Input
DataIn1 [10]	35	DSRP1 Digital Input
DIL_N_2	36	Digital Input for channel configuration
GNDO	37	Ground
lvds_underflow_D3	38	Serializer TX3 underflow digital output
lvds_overflow_D3	39	Serializer TX3 overflow digital output
lvds_underflow_D2	40	Serializer TX2 underflow digital output
lvds_overflow_D2	41	Serializer TX2 overflow digital output
lvds_underflow_D1	42	Serializer TX1 underflow digital output
lvds_overflow_D1	43	Serializer TX1 overflow digital output
CMP_EN_N	44	Comparator Enable. Digital Input
RST_N	45	Reset pin. Digital Input
lvds_underflow_D0	46	Serializer TX1 underflow digital output
lvds_overflow_D0	47	Serializer TX1 overflow digital output
CRC_N	48	CRC Enable. Digital Input
LVDS_SYS_CLK	49	Digital Input for 10 MHz clock (max)
LVDS_SER_CLK	50	Digital Input for 160 MHz clock (max)

lvds_tx_clk_D[3:0]	51-54	Digital output Transmitter clocks
lvdss_tx2_D3	55	Serializer TX3 Digital output lane 2
lvdss_tx1_D3	56	Serializer TX3 Digital output lane 1
lvdss_tx0_D3	57	Serializer TX3 Digital output lane 0
lvdss_tx2_D2	58	Serializer TX2 Digital output lane 2
lvdss_tx1_D2	59	Serializer TX2 Digital output lane 1
lvdss_tx0_D2	60	Serializer TX2 Digital output lane 0
lvdss_tx2_D1	61	Serializer TX1 Digital output lane 2
lvdss_tx1_D1	62	Serializer TX1 Digital output lane 1
lvdss_tx0_D1	63	Serializer TX1 Digital output lane 0
lvdss_tx2_D0	64	Serializer TX0 Digital output lane 2
lvdss_tx1_D0	65	Serializer TX0 Digital output lane 1
lvdss_tx0_D0	66	Serializer TX0 Digital output lane 0
VDDO	67	3.3V VDDO Supply pin
VDD	68-69	3.3V VDD Supply pin
ADC_CLK	70	Digital Output Clock 10 MHz (max)
CIN_N	71	Digital input Carry in, for Pedestal Subtraction
VDDR	72	3.3V VDDR Supply pin
GNDR	73	Ground Pin
DIL_N_1	74	Digital Input for channel configuration
Datain0[5:0]	75-80	Digital Input Port
TH_VAL [11:0]	81-92	Digital Input Port for pedestal configuration
TH_BUS_Din [11:0]	93-104	Digital Input Port for pedestal configuration
FCODE [3:0]	105-108	4-bit digital input port to select functional code
Datain1[9:0]	109-118	Digital Input Port
pulse_ack	119	Digital Output pin for testing purpose.
GNDI	120-123	Core ground pins
VDD	124-126	3.3 V Supply pins

VDDI	127-129	3.3 V Core Supply pins
VDDO	130	3.3 V VDDO Supply pin
GNDI	131-132	Core ground pins
GNDO	133-134	GNDO ground pin
GNDR	135-136	GNDR ground pin
STR_CLKA	137	Digital input clock 10 MHz (max)
Datain0[11:6]	138-143	Digital Input Port
DIL_N [0]	144	Digital Input for channel configuration

Table 26: ASIC Pin List.

4.6 Conclusion

This chapter provides a description of the first three phases of the digital design flow MDART ASIC. The phases of RTL verification, synthesis, and physical (place and route) implementation are explained. These first three phases, followed by LVS and DRC procedures, should be performed before the sign-off phase. Design validation was performed through both RTL and post-layout simulations. The performance verification results confirmed that the implementation of the ASIC design met expectations in terms of speed, flexibility, and portability of the implemented RTL code. In terms of metastability, no violations were found in the recorded setup and hold times. In addition, by using appropriate clock tree synthesis tools, various clock tree buffers, inverters, and clock gating cells were included in the design to minimize the clock insertion delay and implement a balanced clock tree. Finally, no schematic rule violations were reported, according to LVS. In addition, the DRC has confirmed that the layout design complies with the XFAB design rules.

CHAPTER 5 - ASIC Testing

To test the fabricated ASIC chip, the printed circuit board (PCB) shown in Figure 121 was developed. It is a 6-layer PCB with an internal ground plane and an internal power supply plane. The use of such power supply and ground planes ensures short return current paths and thus also minimises assembly inductance. The remaining layers were used for signal routing. Most components were placed on the top layer, while the decoupling capacitive components were placed on the bottom layer. The use of decoupling components ensures that any return currents are diverted directly to the power supply via decoupling components. A 32-pin connector is used to supply the required FCODE and other control signals from an interfaced Cyclone V FPGA device. Clock generation can be provided either by an external FPGA device or by two built-in crystal oscillators running at 10 and 90 MHz. The upper components, including the MDART ASIC chip, were soldered manually. The developed test fixture allows simultaneous processing of at least 192 analogue channels using the four AD9220ARZ 12-bit devices ADC and the GASSIPLEX signal conditioning circuitry currently used in both HMPID and CPV detectors. The electronics can be easily powered via an external 5V connector. Three independent LT3080EMS8E voltage regulators were used to convert the 5V supply voltage into three 3.3V voltage sources. Each voltage regulator supplied power to the MDART ASIC chip core, the input and output drivers, and other electronic components on the chip, such as the DS90LV031ATM LVDS Integrated Circuits (IC). The DS90LV031ATM external differential line driver ICs were included because XFAB technology does not include libraries for differential line driver cells. The LVDS high-speed differential lines were placed between the power supply planes, power supply indicator LEDs were added, and decoupling capacitors were soldered on both planes and near each IC supply pin to prevent unwanted noise from the power supply plane or inductive and capacitive coupling from other adjacent electronics boards. The final PCB has dimensions of 98 mm × 98 mm. The 3D rendering views of the PCB from top and bottom are shown in Figure 111. Appendix B shows the PCB layout in Altium and the schematic along with an illustration of the final PCB design with soldered components and a list of components that were used on the PCB to verify the function of the ASIC IC. Since the complete chip could not be fully characterised, several sign-off post-layout simulations were performed using SDF files along with the generated netlist to verify that the developed ASIC chip functions as expected.

The SDF file contains accurate timing information, such as circuit interconnect delays, and timing constraints. Using the SDF file, along with the generated netlist, Signoff Static Timing Analysis (STA) can be done so to verify that a fabricated ASIC design meets both the target and timing requirements.

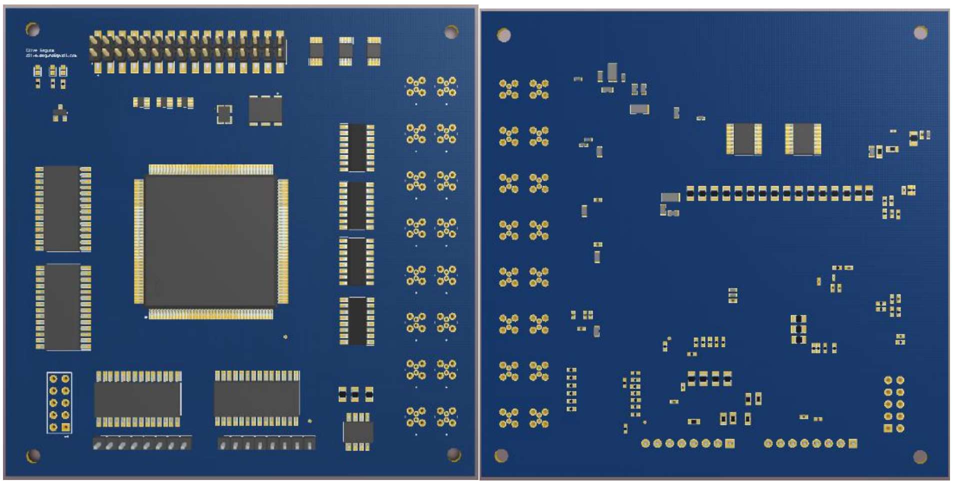


Figure 121: 3D rendered view of the developed Top (left) and Bottom (Right) Printed Circuit Board layers for Testing ASIC device.

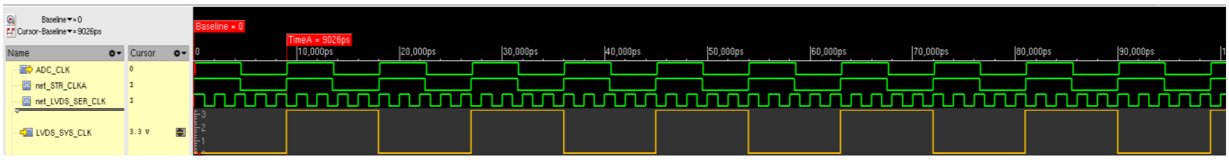


Figure 122: Signoff STA, ADC_CLK, LVDS_SER_CLK (500 MHz) and LVDS_SYS_CLK (55.5 MHz) scaled to (2 ns/division).

The serial data transmitter requires two separate clocks for its operation, LVDS_SER_ CLK and the system clock LVDS_SYS_ CLK. As shown in Figure 122, the LVDS_SER_ CLK is an integer multiple of nine with respect to the system clock LVDS_SYS_ CLK. Phase alignment between the two clocks is not required due to the built-in asynchronous FIFO implementation. The STR _CLKA clock signal controls the DSPR modules and external ADC devices via the ADC_CLK pin.

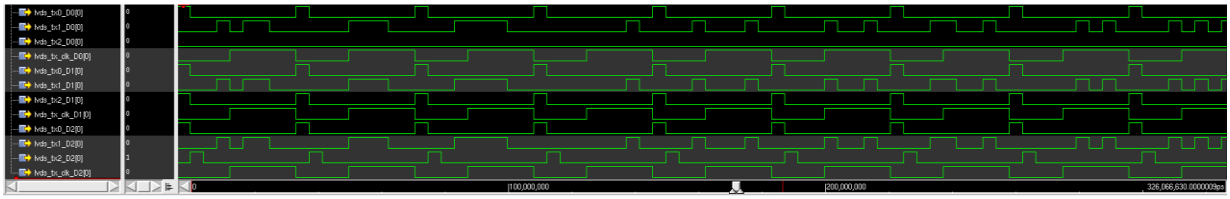


Figure 123: Static Timing Analysis for FCODE = “1000”, data Serializer output (10ns/division), at lvds_tx_clk frequency of 500 MHz, (10ns/division).

The transmitter serializes the parallel 27-bit data input into three separate data lanes lvds_tx0_D<2:0>, lvds_tx1_D<2:0>, lvds_tx2_D<2:0>. Each lane serializes the 9-bits data input with reference to a synchronous lvds_tx_clk. As shown in Figures 123 and 124, lane 1, i.e., lvds_tx0_D0, transmits the first 9 bits of the digital codeword ADC, followed by lane 2 lvds_tx1_D0, which transmits the channel address, and lane 3 lvds_tx2_D0, which transmits the DSPR identification code. The LVDS_SER_ CLK is an integer multiple of nine compared to the synchronous transmitter clock lvds_tx_clk.

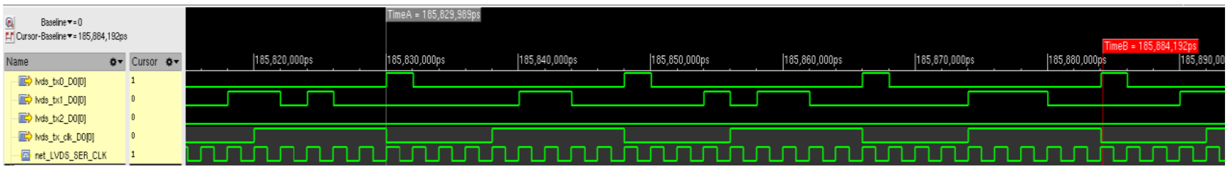


Figure 124: Static Timing Analysis – Relationship between LVDS_SER_CLK (500 MHz), lvds_tx_clk clock, and the 3-lane transmitter data pins (10ns/division).

In Figure 125 one can observe that data transfer is halted while the pedestal thresholds are written to the volatile memory of the DSRP. The FCODE<3:0> port allows setting the DSRP mode of operation.

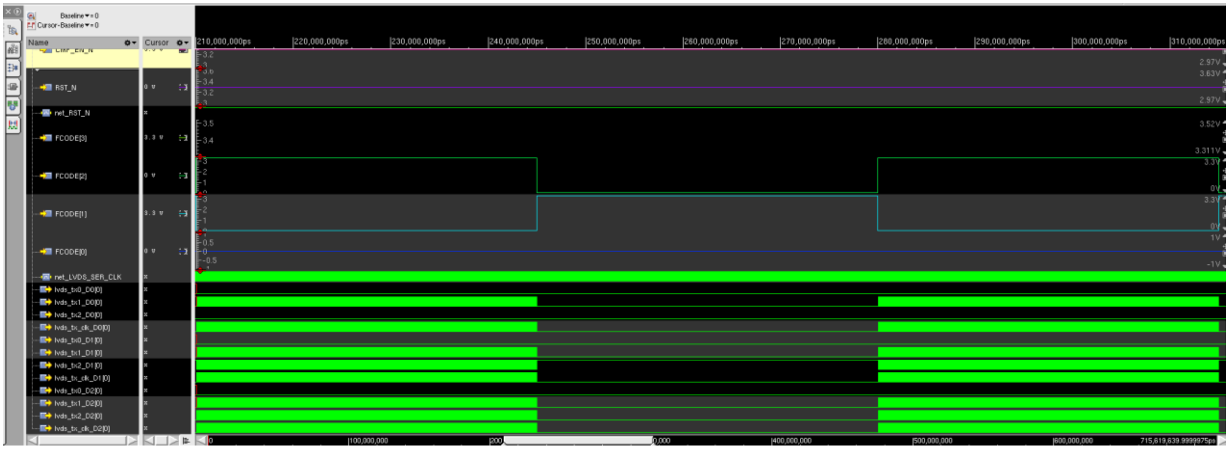


Figure 125: Static Timing Analysis – FCODE value of “1010” followed by “1100” using same an increment digital code as per channel number for pedestal suppression(10ns/division).

CRC option can be enabled or disabled through the active low CRC_N input pin. The computed 5-bit CRC checksum value for the first 21-bits is transmitted serially through lane 3 of each transmitter. This can be observed from the static timing analysis results shown in the following Figures 116 and 117, where for example, taking a CRC initialization value of $(19)_{10}$, and a data value of 0x0, a CRC checksum of $(20)_{10}$ is transmitted on the 3rd lane lvds_tx2_D<2:0>. Again, for data value of 0x02, a CRC checksum of $(14)_{10}$, while for a data value of 0x01a CRC checksum of $(16)_{10}$ is generated.

The 27-bit transmitter data input port is multiplexed into three different lanes *lvds_ tx<t>D<n>*, where *n* represents the transmitter number and *t* represents the lane number for each transmitter. The values of *n* from 0 to 3, while *t* represents values from 0 to 2. Each DSRP processor is assigned an identification code. The serial transmitter lvds_ tx0_D0 is internally wired to DSRP0 and is assigned the identification code 0, while the serial transmitters D1, D2 and D3 are assigned the identification codes 1, 2 and 3 respectively. These identification codes are appended to the transmitted data frame in bits < 19:17 >. In addition, transmitter lanes *lvds_ tx<t>_D0* and *lvds_ tx<t>_D1* contain the transmitter data input bits < 16:0 >, which contain the 12-bit digital codeword and the respective ADC channel address, shown in Figure 128.

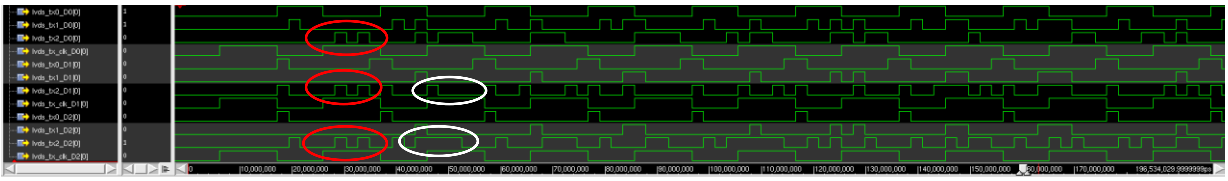


Figure 126: Static Timing Analysis – FCODE value of "1000" with CRC enabled. Various checksum values transmitted through lane 3.

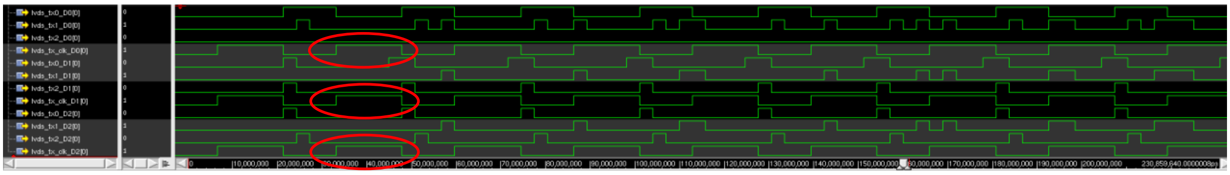


Figure 127: Static Timing Analysis – FCODE value of "1000" with CRC disabled. No checksum value transmitted through lane.

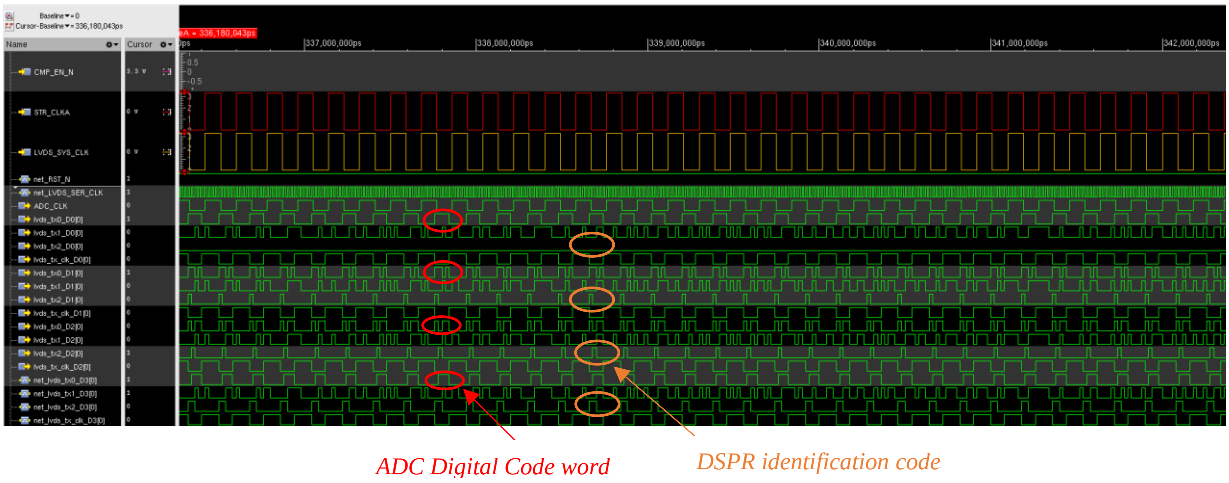


Figure 128: Serial data transmission of ADC digital code word and DSPR identification code bits <19:17> with values 0,1, 2, 3.

5.1 Conclusion

In this chapter, test results were presented for the implemented MDART ASIC IC fabricated using X-FAB XH018 technology. A 6-layer PCB was designed to test the ASIC, with the appropriate input and output connectors for the FPGA configuration, LVDS driver lines, and four 12-bit ADCs. A minor design flaw prevented full characterization of the ASIC IC. Therefore, signoff timing analysis measurements were performed after the ASIC device was laid out to show that the IC performed as expected in various operating modes. Compared to the DILOGIC device used in both ALICE HMPID and CPV particle detectors at CERN, the implemented ASIC device allows the simultaneous processing of at least 192 analogue channels, a fourfold increase in the number of channels. In addition, unlike the DILOGIC processor, MDART applies TMR strategies for error detection and correction, with integrated CRC circuitry. The maximum DILOGIC operating frequency is 10 MHz, while signal processing operations on MDART can be performed at a clock rate of at least 10 MHz. The LVDS transmitter integrated into MDART enables data rates of at least 100 MHz and up to 0.5 Gbps per LVDS channel, contributing to at least a tenfold increase in data rates. Compared to the DILOGIC IC, the MDART device can operate at a lower supply voltage of 3.3 V instead of 5.0 V, contributing to a 16% reduction in static power consumption. In addition, the number of discrete SMD components required to operate the DILOGIC with an FPGA device is about a hundred times higher compared to the MDART device, dramatically halving the PCB manufacturing cost.

CHAPTER 6 - Conclusion and Novelty Contribution

This work is based on a literature review of various front-end electronic readout architectures currently used in various particle physics colliders such as the ALICE experiment in CERN. In addition, it relates to the full development of an industrial electronic system that has been installed and commissioned and is currently in use in the ALICE experiment. Key requirements for such experiments include the use of electronic readout systems that can reliably acquire, process, and transmit large amounts of physical data at high speeds (e.g., one petabyte of collision data per second) using low-power, small form factor electronic systems. In addition, luminosity requirements are high because complex particle collisions overlap, requiring more sophisticated reconstruction and analysis. Such requirements place high demands on the computer and electronic data acquisition system. To eliminate such bottlenecks and reliably achieve the required bandwidth and data volume, it is necessary to develop special custom electronic systems that can be easily integrated with minimal cost and power consumption. This is because there is no off-the-shelf solution for a particular particle detector. In addition, the development and upgrade of such electronic systems is time-limited and occurs during an estimated two-year shutdown period. In addition, the front-end readout electronics for such particle detectors use a large number of FPGAs and digital signal processing devices to acquire, filter, and process data from a large number of analogue channels. For example, the Veto charged particle detector, located in the ALICE experiment, uses over 500 custom-designed digital signal processors and 24 FPGA-based devices to simultaneously process 23,040 analogue channels, and transmit data serially over high-speed links. The hardware and firmware design for such data acquisition systems presents several challenges in terms of radiation tolerance, power distribution for the digital circuits, space requirements, and power consumption. The only possible solutions to the high-speed decoupling problems are to slow down the rise time, reduce current transients, reduce the inductance in series with the capacitor, and use multiple capacitors. The first two approaches are contrary to technical progress. Reducing the inductance in series with the decoupling capacitors, while desirable, does not solve the high-speed decoupling problems. In addition, the use of many FPGAs, CPLDs, and digital signal processors in such systems contributes to the challenges of implementing complex PCB multi-layer stack up designs, component grouping and placement, routing traces and mechanisms for thermal management, long traces, vias, loops, and returns. In addition, conventional decoupling techniques use a large number of capacitors spaced far apart, (over 100’s with different values of 1 µF, 0.1 µF, 0.01 µF, 0.001 µF, 100 pF etc…) together with use of power and ground planes for high-speed traces. Improper PCB design can lead to performance degradation, product failure, and noncompliance with EMC regulations. In addition, designers must consider not only traditional requirements, but also aspects related to maximum operating frequency, power consumption, PCB form factor, operating conditions, signal flow, logic levels, type of environment, and electromagnetic energy coupling during high-speed switching of components such as FPGAs, CPLDs, transceiver serializers, processors, and clocks. The manufacturing and assembly costs for such FPGA circuit boards are very expensive and time-consuming, making it difficult and risky to develop custom electronics needed for such high-speed and high-volume data acquisition systems. At the time of writing, no literature was found that clearly explains or offers techniques on how to mitigate such challenges and costs in PCB design. The publications reviewed merely follow traditional approaches. In addition, the solutions offered by many high-tech companies cannot be easily integrated or used in particle detector electronics. The main reasons are the chip interfaces or architectures used, the limited number of analogue input or digital output channels, compatibility issues with pins, the need for firmware design and validation according to the supplier's specifications, the lack of integrated data compression or fault tolerance solutions that can lead to data corruption or even electronics failures in single events, and the limited number of high-speed channels available.

The main objectives and contributions of this thesis relate to the implementation of a completely new electronic front-end readout system currently used in the ALICE CPV physics experiment. With reference to the measurement results presented in Chapter 3, it was demonstrated that an actual readout time of at least ~25 us (40 kHz) was achieved, thus meeting the targets for the CPV detector in the ALICE experiment. This new electronic system FEE RO contributed to a performance improvement in the data transfer rate between FEE and the DAQ computer system by a factor of almost ten compared to the current High Momentum Particle Identification (HMPID) readout detector and the previous CPV electronics. The newly developed upgrade provides significantly improved electronic performance. This improvement in event readout rate is mainly due to the simultaneous readout and processing of 23,040 analogue channels, the newly developed firmware architecture, and the implemented gigabit transceiver interconnect speeds between DAQ and readout electronics of about 3 Gb/s. The measurement results with a detector occupancy of 15% per column show an event readout rate of up to 37 kHz. So, there is enough margin for the actual Run 3 target occupancy of 1%. Future upgrades may be required to achieve the target readout rate of more than 50 kHz. This includes replacing the 700nm DIL5 card technology with a low-cost ASIC module that will result in better system performance in terms of power consumption and throughput. Furthermore, by replacing and integrating the FPGA column control logic into the same custom structured ASIC device, higher speed, lower space or area requirements, and lower maintenance costs can be achieved for the CPV FEE. Therefore, another contribution of this work relates to the development of a new ASIC chip architecture for digital signal processing that can mitigate most of the challenges described above. An evaluation was performed to understand the novelty and scientific contribution of this work compared to other unique front-end electronic readout architectures currently used for particle detectors, such as the Charged-Particle-Veto (CPV) and High-Momentum Particle Identification Detectors (HMPID) used in the ALICE experiment at CERN. This newly developed application-specific integrated circuit is suitable for simultaneous readout, real-time measurement and processing of digital data in a multi-channel data acquisition system. High-speed multichannel digitizers are useful for large-scale experiments in high-energy physics, astrophysics, nuclear physics, and plasma physics. The developed application-specific integrated circuit enables simultaneous continuous readout and processing of 192 analogue 12-bit channels with a data transfer rate of 0.5 Gbps per track. The ASIC device consists of 12 independent lanes that contribute to a total bandwidth of approximately 6 Gbps via low-voltage differential signalling transmitter drivers. Unlike the various vendor-defined high-speed digitizers currently on the market, the developed microelectronic circuit in the 160-pin ceramic low-profile package includes the implementation of an integrated fault-tolerant and recoverable triple-modular redundancy voting circuit, the use of a zero-suppression compression algorithm, and the implementation of the cyclic redundancy check technique. The integration of these features reduces development time and allows the developed integrated circuit to be used in a radiation physics environment where a single event can cause corruption of event data or even failure of the electronics. The implemented system architecture reduces maintenance costs and improves system performance by a factor of ten when compared, for example, with the various electronic systems for data acquisition currently used in the A Large Ion Collider experiment at CERN. In addition, the developed XFAB 180 nm 6-layer parallel readout integrated circuit architecture can be easily interfaced with other various vendor-specific analogue-to-digital converter modules currently available on the market, further facilitating the upgrade process of data acquisition systems. The results of this evaluation showed that the use of this unique ASIC device in such front-end electronic readout detector systems resulted in a 95% reduction in the use of high-cost components (e.g., FPGAs, digital signal processors, clock circuits, decoupling components, etc.), a tenfold increase in readout rate i.e., from 4 kHz to at least 40 kHz, and a lower operating voltage from 5.0 V to 3.3 V. The reason for these improvements is mainly due to the development of a new digital signal processing ASIC chip architecture that can handle 192 analogue channels simultaneously. In addition, the developed ASIC architecture includes the integration and implementation of data compression, error detection and correction modules that help reduce the time-consuming processes and resources required for FPGA firmware validation. In addition, by using the developed ASIC chip and reducing the electronic components required to implement such data acquisition systems, PCB designers can develop compact electronic products that comply with EMC regulations. These requirements are the current key requirements for high-speed electronic systems with fast rise and fall time signals and for mixed signal design.

In summary, based on the above assessment, this research has contributed to a newly adapted Front-end Read-out electronic system architecture for the CPV detector at CERN, which, compared to previous systems, has improved performance due to the:

- increase in event readout rate by more than 50 kHz at 1% detector occupancy, contributing to better detector luminosity levels. This novel electronics has been installed, commissioned and currently in use in the ALICE CPV experiment,
- ability to process multiple detector columns simultaneously, with each column containing 480 channels,
- integration of the functions of 5DIL and FPGA CPV FEE RO into an implemented and fabricated ASIC device called MDART,
- reduction in maintenance costs by using fewer discrete electronic components to simultaneously process 23,040 analogue tungsten lead channels,
- development of a new portable VHDL firmware architecture, for both CPV and HMPID detectors,
- development of VHDL firmware that emulates SIU behaviour in an FPGA to interface the CPV detector with the ALICE detector to replace the hardware required to use the TLK2501 gigabit transceiver.

In addition, this work has been published in seven peer-reviewed conference proceedings, one journal, and one best publication award, demonstrating the validity of this research.

References

[1] D. Rohr, G. Sergey and V. Lindenstruth, "Fast TPC Online Tracking on GPUs and Asynchronous

Data Processing in the ALICE HLT to facilitate Online Calibration,"

Journal of Physics Conference Series, vol. 664, no. 8, p. 082047, 2015.

[2] J. Santiard, "The ALICE HMPID on-detector front-end and readout electronics,"

in *Nucl. Instrum. Methods Phys. Res.*, 2004.

[3] S. Calleja, "HMPID RCB firmware upgrade for Run3," 2016.

[4] H. Witters, J. Santiard and P. Martinengo, "Dilogic2 a sparse data scan readout processor for

the HMPID detector of ALICE," in *Electronics for LHC experiments*, Cracow, 11–15 September 2000.

[5] G.Rubin and P.Vande Vyvre: “ALICE-DDL Interface Control Document”;

ALICE/96-43, Internal Note/DAQ, 1996.

[6] J. Alme et. al., "Radiation-Tolerant, SRAM-FPGA Based Trigger and Readout Electronics

for the ALICE Experiment,"

IEEE Transactions on Nuclear Science, vol. 55, no. 1, pp. 76-83, Feb. 2008.

[7] F. Carena et. al., "DDL, the ALICE data transmission protocol and its evolution from 2 to 6 Gb/s,"

Journal of Instrumentation, vol. 10, pp. C04008, 2015.

[8] G.Rubin and P.Vande Vyvre, "ALICE-DDL Interface Control Document," ALICE/96-43,

Internal Note/DAQ, 1996.

[9] A. A. Grillo, "Primer on Detectors and Electronics," Sematic Scholar, California, 2014.

[10] J. Troska et. al., "The Versatile Transceiver Proof of Concept," in *Proc. TWEPP. Conf.*, Geneva, 2009.

[11] F.Zhang et. al., "Point-to-point readout for the ALICE EMCal detector,"

Nuclear Instruments and Methods in Physics Research, vol. 735, pp. 157-162, 2014.

[12] A. Velure, "Upgrades of the ALICE TPC front-end electronics for LS1 and LS2,"

2014 19th IEEE-NPSS Real Time Conference, 2014, pp. 1-3, doi: 10.1109/RTC.2014.7097489.

[13] I. Kuon and J. Rose, "Measuring the Gap Between FPGAs and ASICs,"

in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*,

vol. 26, no. 2, pp. 203-215, Feb. 2007, doi: 10.1109/TCAD.2006.884574.

[14] A. Corporation, "Section I. HardCopy II device Datasheet Family," 2007. [Online].

Available: https://www.intel.com/content/dam/www/programmable/us/en/pdfs

/literature/hb/hrd/hc_h5v1_05.pdf. [Accessed 3 9 2018].

[15] J. Knopf, P. Fischer, C. Kreidl, "A 256 channel 8-Bit current digitizer ASIC for the Belle-II PXD,"*in Journal of Instrumentation*, vol. 6, pp. 1-6, 2011.

[16] D. Meier et al., "Development of an ASIC for charged particle counting with silicon radiation detectors," 2013 IEEE Nuclear Science Symposium and Medical Imaging Conference (2013 NSS/MIC), 2013, pp. 1-5, doi: 10.1109/NSSMIC.2013.6829764.

[17] G. G. Tambave and A. Velure, "Qualification of the ALICE SAMPA ASIC With a High-Speed Continuous DAQ System," in IEEE Transactions on Nuclear Science, vol. 64, no. 6, pp. 1461-1466, June 2017, doi: 10.1109/TNS.2017.2707339.

[18] R. E. Bosch, A. J. de Parga, B. Mota and L. Musa, "The ALTRO chip: a 16-channel A/D converter and digital processor for gas detectors," 2002 IEEE Nuclear Science Symposium Conference Record, 2002, pp. 566-570 vol.1, doi: 10.1109/NSSMIC.2002.1239377.

[19] F. Anghinolfi, P. Jarron, F. Krummenacher, E. Usenko and M. C. S. Williams, "NINO, an ultra-fast, low-power, front-end amplifier discriminator for the Time-Of-Flight detector in ALICE experiment," 2003 IEEE Nuclear Science Symposium. Conference Record (IEEE Cat. No.03CH37515), 2003, pp. 375-379 Vol.1, doi: 10.1109/NSSMIC.2003.1352067.

[20] M.J. Lee et al., "Design and performance of a custom ASIC digitizer for wire chamber readout in 65 nm CMOS technology," *Journal of Instrumentation*, vol. 10, pp. 1-21 2015, doi: 10.1088/1748-0221/10/06/P06007

[21] F. Zhang et al., "Point-to-point readout for the ALICE EMCal detector," *Nuclear Instruments and Methods in Physics Research*, vol. 735, pp. 157–162, 2014., doi: 10.1016/j.nima.2013.09.023

[22] A. Corporate, "Intel," 2 9 2018. [Online]. Available: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/wp/wp-01207-single-event-functional-interrupt.pdf. [Accessed 3 9 2018].

[23] I. Technologies, "Working Environments," IROC Support Center, 2016. [Online]. Available: https://www.iroctech.com/library/working-environments/. [Accessed 3 9 2018].

[24] C. Corporation, "Radiation Dose and Fluence in ALICE after LS2," 2018. [Online]. Available: https://alice-notes.web.cern.ch/node/405. [Accessed 3 9 2018].

[25] H. Engel, "The C-RORC PCIe card and its application in the ALICE and ATLAS experiments," in *Topical Workshop on Electronics for Particle Physics 2014*, Aix en Provence,, 2014.

[26] "ECUE FireFly Low Profile Micro Flyover System Cable Assembly," Samtec , 2021. [Online]. Available: https://www.samtec.com/products/ecue. [Accessed 15 12 202 1].

[27] X-FAB, "X-FAB Global Services GmbH," 7 December 2020. [Online]. Available: https://my.xfab.com/search/_action/open/. [Accessed 19 7 2021].

[28] V. P. Priya, "Design of Parallel CRC generation for High Speed Application," *International Journal of Engineering Science and Computing,* vol. 7, no. 8, pp. 14733 - 14736 , 2017.

[29] P. Koopman, "Cyclic redundancy code (CRC) polynomial selection for embedded networks," in *International Conference on Dependable Systems and Networks*, Florence, Italy, 2004.

[30] E. Stavinov, "A Practical Parallel CRC Generation Method," 5 1 2010. [Online]. Available: www.circuitcellar.com. [Accessed 3 1 2022].

[31] K. CH. Janakiram, "An Efficient Technique for Parallel CRC Generation," *International Journal Of Engineering And Computer Science,* vol. 3, no. 12, pp. 9761-9765, 2014.

[32] X.-F. S. F. 2017, "XH018 IO Library Documentation," August 2018. [Online]. Available: https://www.xfab.com/. [Accessed 29 7 2021].

[33] "Intel FPGA," 1 September 2018. [Online]. Available: https://www.intel.com/content/www/us/en/products/programmable/fpga/cyclone-v.html.

[34] J.Schukraft et al., "ALICE Technical Design Report of the High Momentum Particle Identification Detector," ALICE Collaboration, Geneva, 1998.

[35] The ALICE Technical Proposal; CERN/LHCC 95-71, ISBN 92-9083-07708.

[36] C. Soos, and G. Rubin,"ALICE-DDL Hardware Guide for the Front-end Designers," ALICE-INT-1998-21, CERN-ALICE-INT-1998-21, 1998.

[37] G.Rubin: “ALICE-DDL Physical and Signalling Interface Specification”;

ALICE/97-04, Internal Note/DAQ, 14.02.1997

[38] P. Antonioli, C. Baldanza, D. Falchieri, F. M. Giorgi, A. Mati and C. Tintori,

"Design and test of a GBTx based board for the upgrade of the ALICE TOF readout electronics," 2016 IEEE-NPSS Real Time Conference (RT), 2016, pp. 1-3, doi: 10.1109/RTC.2016.7543121.

[39] H. Soltveit et al., "The preamplifier shaper for the ALICE TPC detector,"

Nucl. Instrum. Methods Phys. Res., vol. 676, 2012, pp. 106-109,
doi: 10.1016/j.nima.2012.02.012.

[40] A. Collaboration, "The ALICE Transition Radiation Detector: Construction, operation, and performance," *Nuclear Inst. and Methods in Physics Research, A*, 2018, pp. 88-127, doi: 10.1016/j.nima.2017.09.028

[41] N. H. E.Weste, "CMOS VLSI Design A Circuits and Systems", Addison-Wesley, 2005.

[Online]. Available: http://swarm.cs.pub.ro/~mbarbulescu/SMPA/CMOS-VLSI-design.pdf

[42] P. Antonioli, A. Kluge, and W. Riegler,"Upgrade of the ALICE Read-Out and Trigger System," ALICE Collaboration, CERN-LHCC-2013-019, ALICE-TDR-015, Sep, 2013. [Online]
Available: https://cds.cern.ch/record/1603472. [Accessed 3 1 2021].

[43] J. Schambach et al., "A Radiation-Tolerant Readout System for the

ALICE Inner Tracking System Upgrade," 2018 IEEE Nuclear Science Symposium and Medical Imaging Conference Proceedings (NSS/MIC), 2018, pp. 1-6, doi: 10.1109/NSSMIC.2018.8824419.

[44] H. Witters, J.C. Santiard, and P. Martinengo,

"Dilogic-2: A Sparse Data Scan Readout Processor for the HMPID Detector of ALICE," Geneva, 2000.

[45] T. Instruments, "TLK2501," 2006. [Online]. Available: https://www.ti.com/product. [Accessed 7 7 2021].

[46] Intel, "Mouser Electronics," 23 1 2020. [Online].

Available: https://eu.mouser.com/datasheet/2/612/cfg_cf52012-1301744.pdf. [Accessed 7 7 2021].

[47] Intel, 3 2009. [Online]. Available: ttps://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/ug_pdn.pdf.

[Accessed 7 7 2021].

[48] T. Instruments, "Kaimeite Electronics," October 2018. [Online].

Available: https://kaimte.com/product/details/ti-texas-instruments/. [Accessed 2021 7 8].

[49] T. Instruments, "Texas Instruments," January 2008. [Online].

Available: https://www.ti.com/lit/ds/scas649b/scas649b.pdf. [Accessed 12 7 2021].

[50] "FOSCO," 2021. [Online].

Available: https://www.fiberoptics4sale.com/blogs/archive-posts/

95047174-what-is-ber-bit-error-ratio-and-bert-bit-error-ratio-tester. [Accessed 12 7 2021].

[51] J. Hottenroth and P. Webb, "Making custom measurements with high-speed digitizers -

In the FPGA or host processor," 2008 IEEE AUTOTESTCON, 2008, pp. 249-251,

doi: 10.1109/AUTEST.2008.4662620.

[52] S. Cometti and G. Mazza, "New ASICs for the Very Front-End Card

Upgrade of the CMS ECAL Barrel at the HL-LHC," 2019 15th Conference on Ph.D Research in Microelectronics and Electronics (PRIME), 2019, pp. 37-40, doi: 10.1109/PRIME.2019.8787778.

[53] A. A. Grillo, "Primer on Detectors and Electronics for Particle Physics Experiments,"

physics.ins-det, p. 30, 8 1 2019.

[54] S. Rooban, P. Divya, P. Preethi, and M. Divya, "A Fault Tolerant Han-Carlson adder using Tmr Technique," *International Journal of Recent Technology and Engineering (IJRTE),* vol. 8, pp. 2277-3878, Nov. 2019.

[55] F. L. Kastensmidt, L. Carro, and R. Reis, "Fault-tolerance techniques for SRAM-based FPGAs", Springer, vol. 32, 2006.

[56] M. H. Rahman, "A Fault Tolerant Voter Circuit for Triple Modular Redundant," *Journal of Electrical and Electronic Engineering,* vol. 5, pp. 165-166, 2017.

[57] e. NV, "easics," webdesign w247.be, 2020. [Online]. Available: https://www.easics.com/crctool/. [Accessed 27 7 2021].

[58] G. Campobello, G. Patane and M. Russo, "Parallel CRC realization," in IEEE Transactions on Computers, vol. 52, no. 10, pp. 1312-1319, Oct. 2003, doi: 10.1109/TC.2003.1234528.

[59] V. P. Priya, "Design of Parallel CRC generation for High Speed Application," *International Journal of Engineering Science and Computing,* vol. 7, no. 8, pp. 14733-14736, 2017.

[60] P. Moreira et al., “TTCrx Reference Manual,” CERN, Internal document, 2003.

[61] G. Rubin, J. Sulyán, N. Tóth, C. Soós, P. Van de Vyvre, I. Novák, P. Csató, D. Tarján, A. Telegdy, G. Vesztergombi, et al., “The ALICE detector data link,” tech. rep., CERN, 1999.

[62] "Europractice-IC - ASICS," [Online]. Available: https://europractice-ic.com/technologies/asics/. [Accessed 11 12 2021].

[63] K. Aamodt et al., "The ALICE experiment at the CERN LHC ," *Journal of Instrumentation,* vol. 3, pp. S08002 2008, doi: 10.1088/1748-0221/3/08/S08002.

[64] B. Abelev et al., "Upgrade of the ALICE Experiment: Letter Of Intent,"

Journal of Physics G: Nuclear and Particle Physics, vol. 41, pp. 087001 2014,

doi: 10.1088/0954-3899/41/8/087001

[65] B. Abelev et al., "Performance of the ALICE Experiment at the CERN LHC,"

International Journal of Modern Physics A, vol. 29, pp. 1430044 2014,

doi: 10.1142/S0217751X14300440

[66] J.-C. Santiard et al., "Gasplex : a low-noise analog signal processor for readout of gaseous detectors," in *6th Pisa Meeting on Advanced Detectors*, Italy, 1994.

[67] A. Breskin, "R&D results on a CsI-TTGEM based photodetector,"

Nuclear Instruments and Methods in Physics Research A, pp. 126-129, 2010.

[68] S. Cometti and G. Mazza, "The front-end data conversion and readout electronics for the CMS ECAL upgrade," *Journal of Instrumentation,* vol. 13, p. C03003, 2018, doi: 10.1088/1748-0221/13/03/c03003.

[69] S. Cometti and G. Mazza, "Design of the new front-end electronics for the readout of the upgraded CMS electromagnetic calorimeter for the HL-LHC," *in Topical Workshop on Electronics for Particle Physics*, 2018, doi: 10.22323/1.313.0106

[70] R. E. Bosch, A. J. de Parga, B. Mota and L. Musa,

"The ALTRO chip: a 16-channel A/D converter and digital processor for gas detectors,"

2002 IEEE Nuclear Science Symposium Conference Record, 2002, pp. 566-570 vol.1,

doi: 10.1109/NSSMIC.2002.1239377.

[71] J. Alme. et. al., "RCU2 - The ALICE TPC readout electronics consolidation for Run2,"

Journal of Instrumentation, vol. 8, no. 12, pp. C12032-C12032, 2013,

doi: 10.1088/1748-0221/8/12/c12032.

[72] J. Alme et. al., "The ALICE TPC, a large 3-dimensional tracking device with fast readout for ultra-high multiplicity events," *Nuclear Instruments and Methods in Physics Research Section A*, vol. 622, no. 1, pp. 316-367, 2010, doi: 10.1016/j.nima.2010.04.042.

[73] P. Moreira et. al., "The GBT project," *in Topical Workshop on Electronics for Particle Physics*, Vols. CERN-2009-006, pp. 342-346, 2009.

[74] J. Borga et. al., "The C-RORC PCIe card and its application in the ALICE and ATLAS experiments," in *International Journal of Modern Physics A*, vol. 10, pp. C02022 2015, doi:10.1088/1748-0221/10/02/C02022.

[75] XILINX, "Virtex-II Pro and Virtex-II Pro X Platfrom FPGAs: Complete Datasheet(DS083 (v5.0)," XILINX, 2011, Available: https://www.xilinx.com/support/documentation/data_sheets/ds083.pdf. [Accessed 10 12 2021].

[76] Macronix International, "Macronix MX29LV64OBT/BB 64M-BIT [8Mx8/4Mx16] single voltage," Macronix , 2005, Available: https://www.macronix.com/Lists/Datasheet/. [Accessed 10 12 2021].

[77] A. Szczepankiewicz, "Readout of the upgraded ALICE-ITS," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 824, no. 0168-9002, pp. 465-469, 2016, doi:10.1016/j.nima.2015.10.056.

[78] P.Buncic, M. Krzewicki, and P. Vande Vyvre, "Technical Design Report for the Upgrade of the Online-Offline Computing System," 2015, Available: https://cds.cern.ch/record/2011297. [Accessed 10 12 2021].

[79] F. Costa, "ITS Readout Electronic – Production Readiness Review," [Online]. Available: https://www.phenix.bnl.gov/WWW/publish/mxliu/sPHENIX

/ITS-Reviews/Electrical-PRR-04132018/WP10_PRR_RU_Overview_v1.pdf. [Accessed 3 1 2022].

[80] P. Boeschoten and Filippo Costa, "The ALICE O2 common driver for the C-RORC and CRU read-out cards," *Journal of Physics Conference Series* , vol. 1085, no. 3, pp. 1-5, 2017.

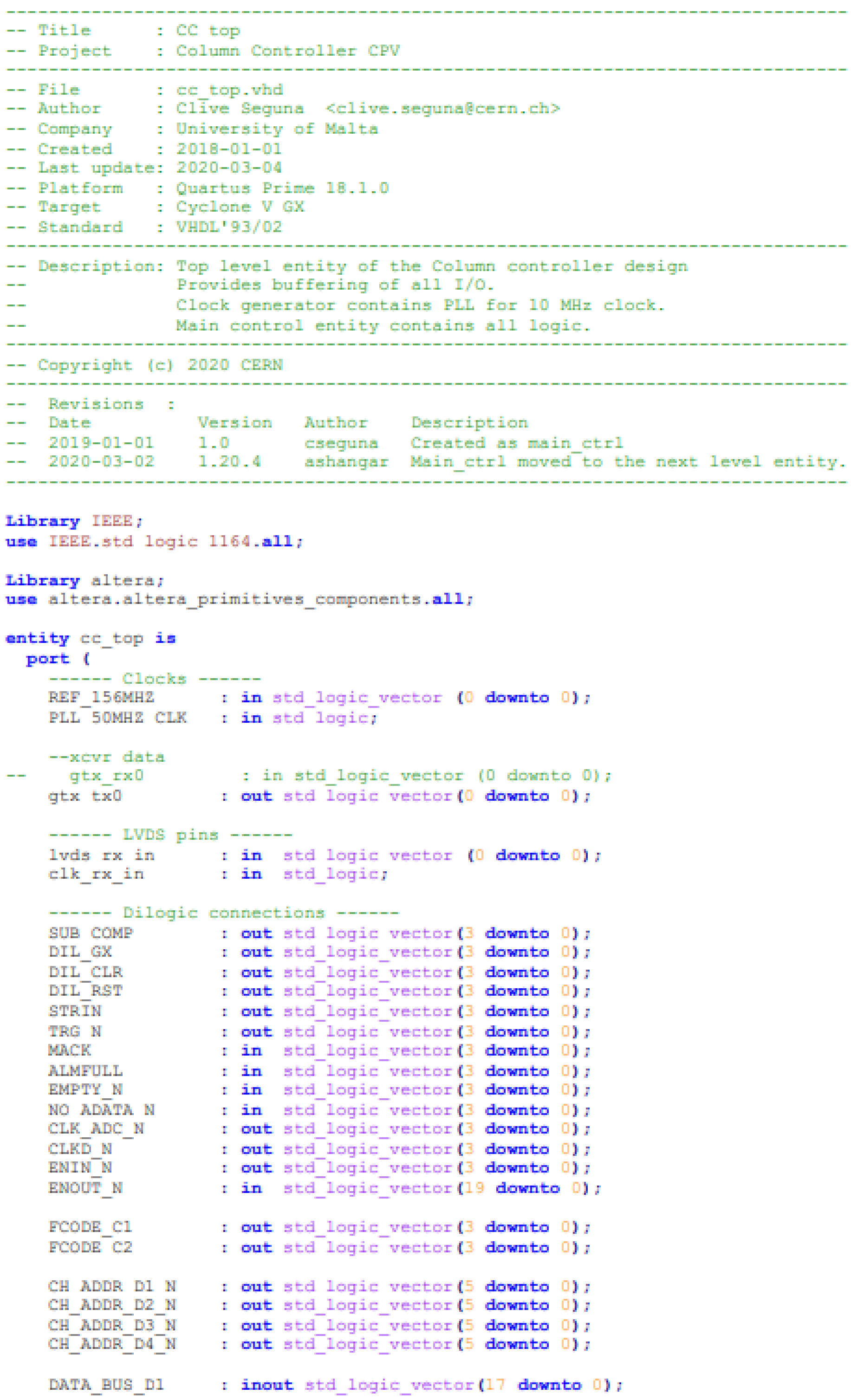
[81] P. Antonioli, C. Baldanza, D. Falchieri, F. M. Giorgi, A. Mati and C. Tintori,

"Design and test of a GBTx based board for the upgrade of the ALICE TOF readout electronics," 2016 IEEE-NPSS Real Time Conference (RT), 2016, pp. 1-3, doi: 10.1109/RTC.2016.7543121.

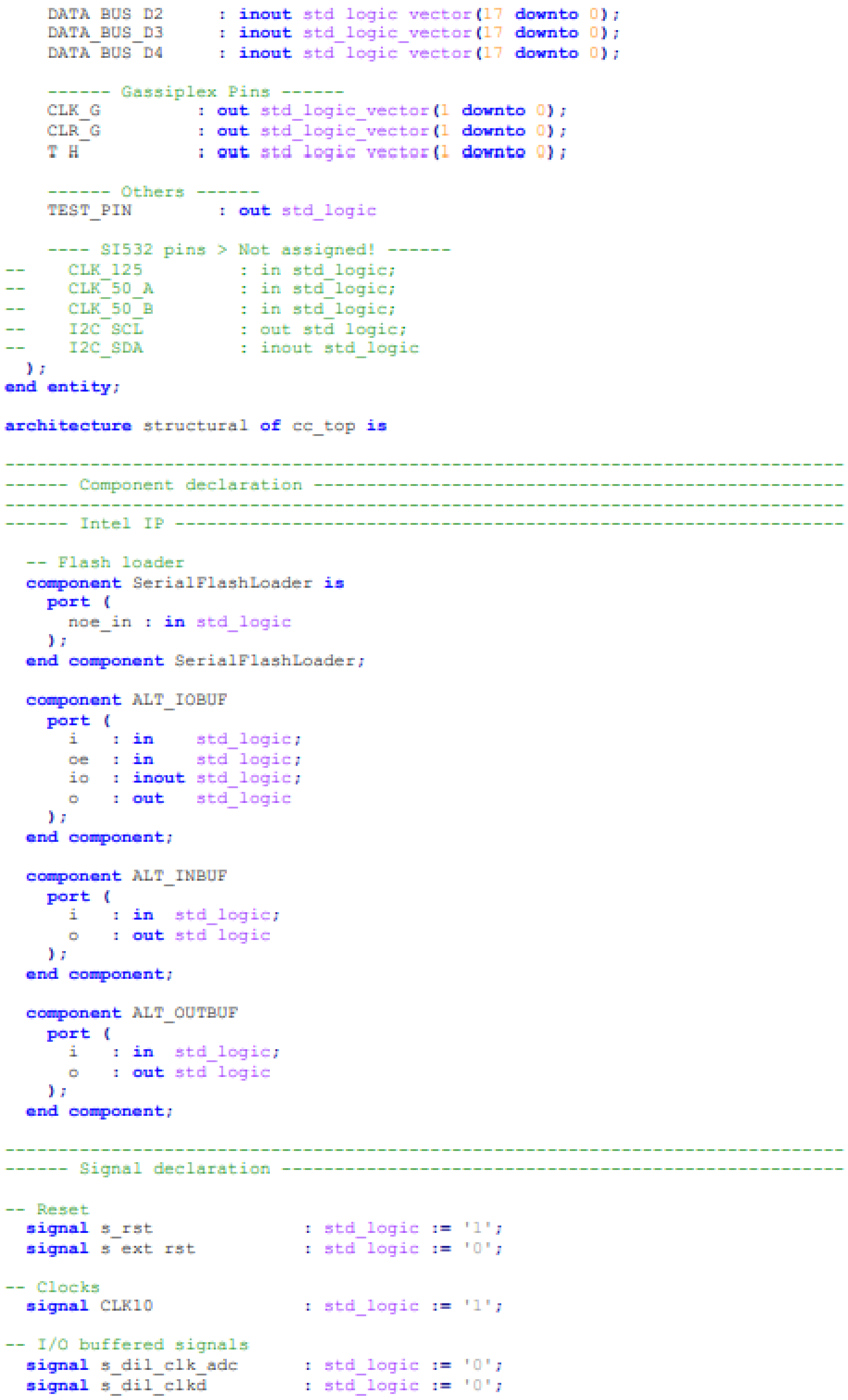
[82] Microsemi., "Igloo2 FPGA Family Website," 6 5 2017. [Online].

Available: https://www.microsemi.com/product-directory/fpgas/1688-igloo2. [Accessed 2022].

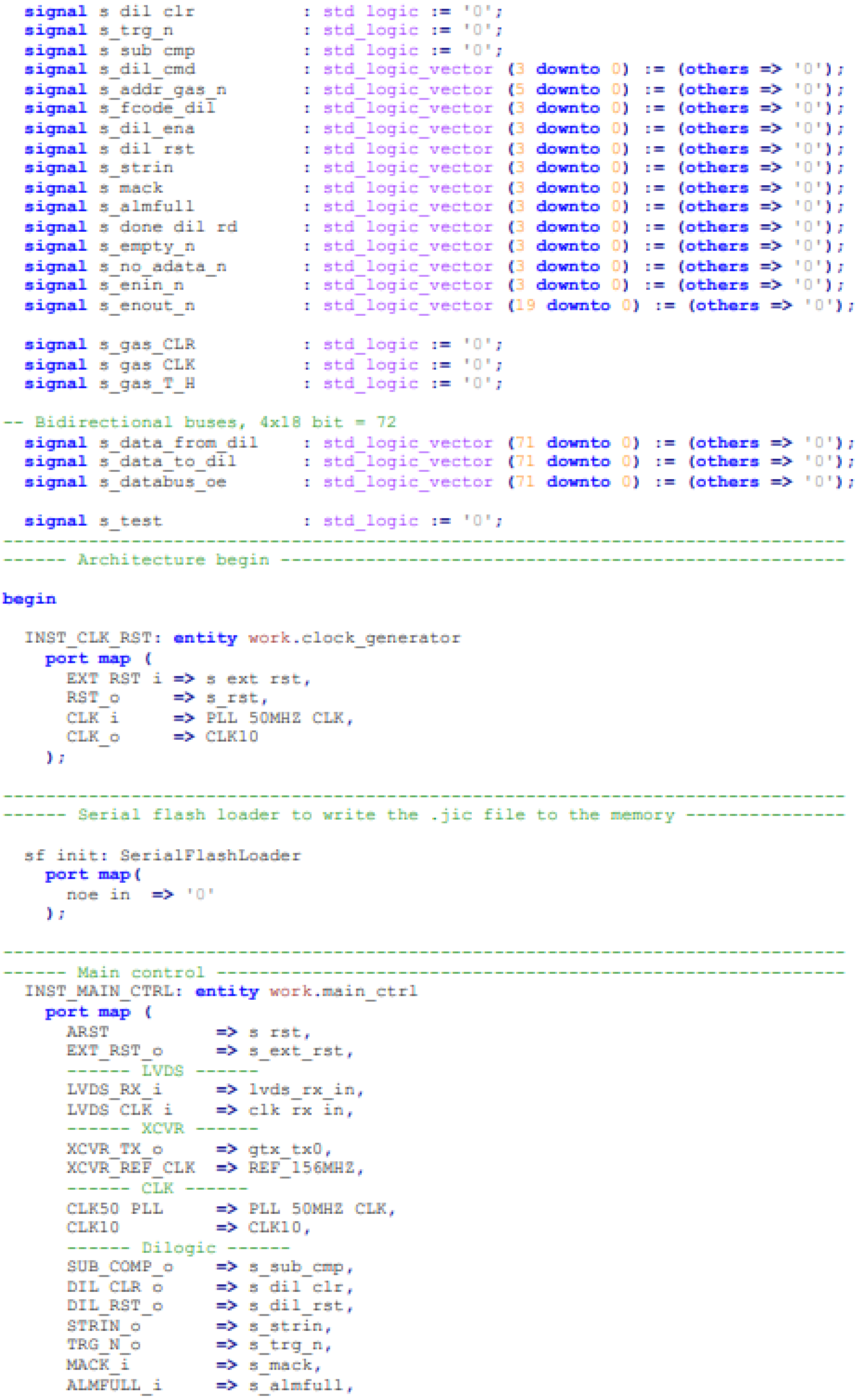
Appendix A.1 - Column Controller Firmware – cc_top.vhd



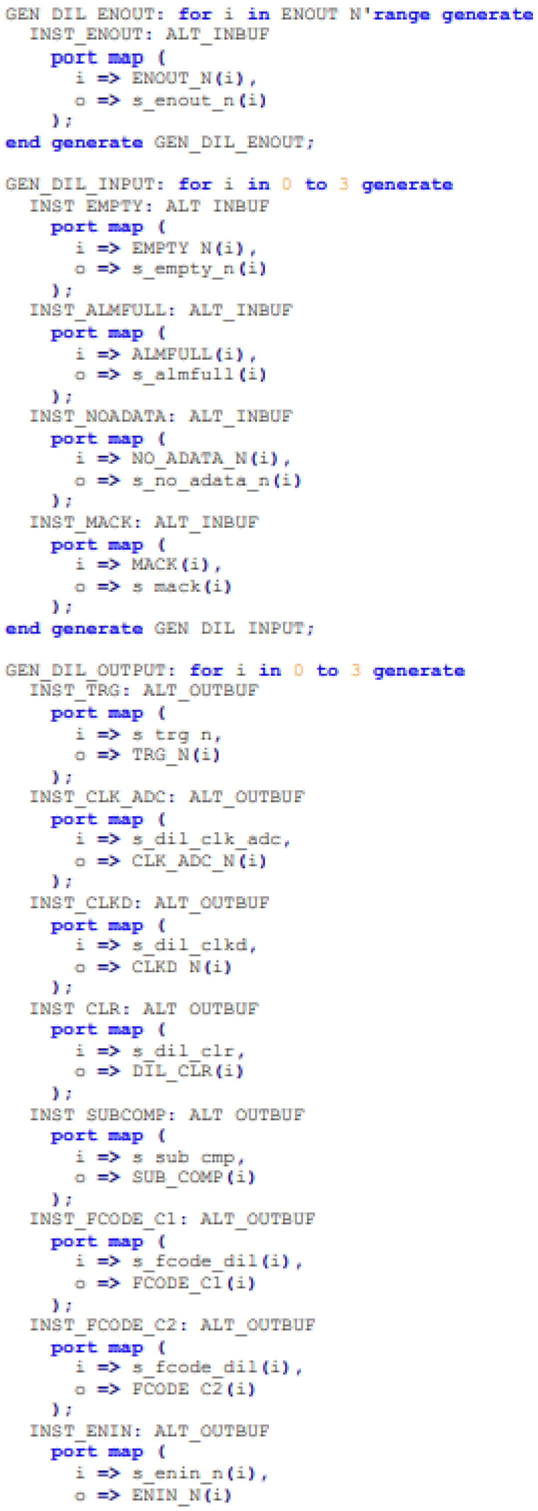
```
-------------------------------------------------------------------------------
-- Title      : CC top
-- Project    : Column Controller CPV
-------------------------------------------------------------------------------
-- File       : cc_top.vhd
-- Author     : Clive Seguna  <clive.seguna@cern.ch>
-- Company    : University of Malta
-- Created    : 2018-01-01
-- Last update: 2020-03-04
-- Platform   : Quartus Prime 18.1.0
-- Target     : Cyclone V GX
-- Standard   : VHDL'93/02
-------------------------------------------------------------------------------
-- Description: Top level entity of the Column controller design
--              Provides buffering of all I/O.
--              Clock generator contains PLL for 10 MHz clock.
--              Main control entity contains all logic.
-------------------------------------------------------------------------------
-- Copyright (c) 2020 CERN
-------------------------------------------------------------------------------
--  Revisions  :
--  Date           Version   Author    Description
--  2019-01-01     1.0       cseguna   Created as main_ctrl
--  2020-03-02     1.20.4    ashangar  Main_ctrl moved to the next level entity.
-------------------------------------------------------------------------------

Library IEEE;
use IEEE.std_logic_1164.all;

Library altera;
use altera.altera_primitives_components.all;

entity cc_top is
  port (
    ------ Clocks ------
    REF_156MHZ       : in std_logic_vector (0 downto 0);
    PLL_50MHZ_CLK    : in std_logic;

    --xcvr data
--    gtx_rx0           : in std_logic_vector (0 downto 0);
    gtx_tx0          : out std_logic_vector(0 downto 0);

    ------ LVDS pins ------
    lvds_rx_in       : in  std_logic_vector (0 downto 0);
    clk_rx_in        : in  std_logic;

    ------ Dilogic connections ------
    SUB_COMP         : out std_logic_vector(3 downto 0);
    DIL_GX           : out std_logic_vector(3 downto 0);
    DIL_CLR          : out std_logic_vector(3 downto 0);
    DIL_RST          : out std_logic_vector(3 downto 0);
    STRIN            : out std_logic_vector(3 downto 0);
    TRG_N            : out std_logic_vector(3 downto 0);
    MACK             : in  std_logic_vector(3 downto 0);
    ALMFULL          : in  std_logic_vector(3 downto 0);
    EMPTY_N          : in  std_logic_vector(3 downto 0);
    NO_ADATA_N       : in  std_logic_vector(3 downto 0);
    CLK_ADC_N        : out std_logic_vector(3 downto 0);
    CLKD_N           : out std_logic_vector(3 downto 0);
    ENIN_N           : out std_logic_vector(3 downto 0);
    ENOUT_N          : in  std_logic_vector(19 downto 0);

    FCODE_C1         : out std_logic_vector(3 downto 0);
    FCODE_C2         : out std_logic_vector(3 downto 0);

    CH_ADDR_D1_N     : out std_logic_vector(5 downto 0);
    CH_ADDR_D2_N     : out std_logic_vector(5 downto 0);
    CH_ADDR_D3_N     : out std_logic_vector(5 downto 0);
    CH_ADDR_D4_N     : out std_logic_vector(5 downto 0);

    DATA_BUS_D1      : inout std_logic_vector(17 downto 0);
```



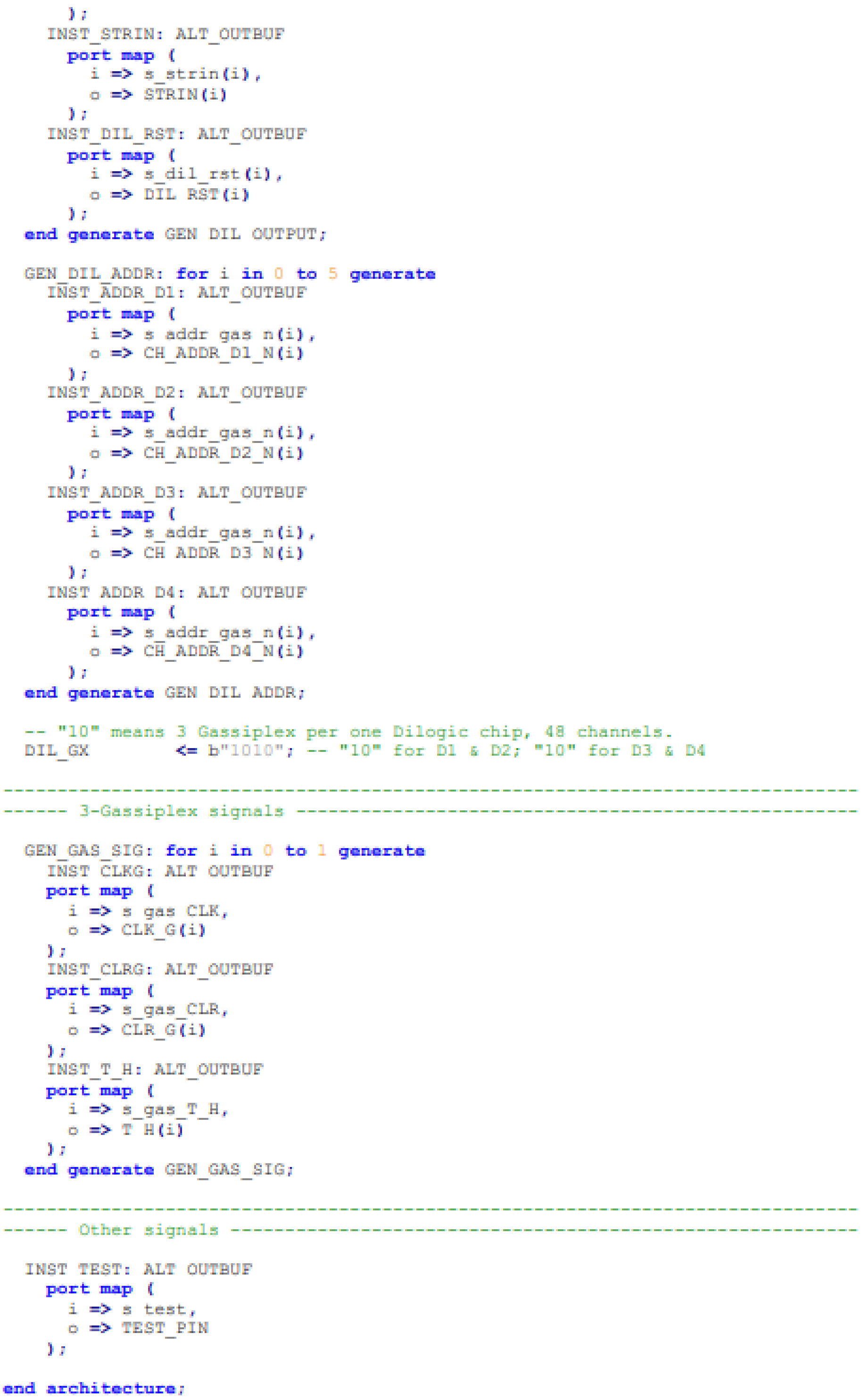
```vhdl
    DATA_BUS_D2      : inout std_logic_vector(17 downto 0);
    DATA_BUS_D3      : inout std_logic_vector(17 downto 0);
    DATA_BUS_D4      : inout std_logic_vector(17 downto 0);

    ------ Gassiplex Pins ------
    CLK_G          : out std_logic_vector(1 downto 0);
    CLR_G          : out std_logic_vector(1 downto 0);
    T_H            : out std_logic_vector(1 downto 0);

    ------ Others ------
    TEST_PIN         : out std_logic

    ---- SI532 pins > Not assigned! ------
--    CLK_125          : in std_logic;
--    CLK_50_A         : in std_logic;
--    CLK_50_B         : in std_logic;
--    I2C_SCL          : out std_logic;
--    I2C_SDA          : inout std_logic
  );
end entity;

architecture structural of cc_top is

--------------------------------------------------------------------------------
------ Component declaration ---------------------------------------------------
--------------------------------------------------------------------------------
------ Intel IP ----------------------------------------------------------------

  -- Flash loader
  component SerialFlashLoader is
    port (
      noe_in : in std_logic
    );
  end component SerialFlashLoader;

  component ALT_IOBUF
    port (
      i   : in    std_logic;
      oe  : in    std_logic;
      io  : inout std_logic;
      o   : out   std_logic
    );
  end component;

  component ALT_INBUF
    port (
      i   : in  std_logic;
      o   : out std_logic
    );
  end component;

  component ALT_OUTBUF
    port (
      i   : in  std_logic;
      o   : out std_logic
    );
  end component;

--------------------------------------------------------------------------------
------ Signal declaration ------------------------------------------------------

-- Reset
  signal s_rst               : std_logic := '1';
  signal s_ext_rst           : std_logic := '0';

-- Clocks
  signal CLK10               : std_logic := '1';

-- I/O buffered signals
  signal s_dil_clk_adc       : std_logic := '0';
  signal s_dil_clkd          : std_logic := '0';
```



```
  signal s_dil_clr           : std_logic := '0';
  signal s_trg_n             : std_logic := '0';
  signal s_sub_cmp           : std_logic := '0';
  signal s_dil_cmd           : std_logic_vector (3 downto 0) := (others => '0');
  signal s_addr_gas_n        : std_logic_vector (5 downto 0) := (others => '0');
  signal s_fcode_dil         : std_logic_vector (3 downto 0) := (others => '0');
  signal s_dil_ena           : std_logic_vector (3 downto 0) := (others => '0');
  signal s_dil_rst           : std_logic_vector (3 downto 0) := (others => '0');
  signal s_strin             : std_logic_vector (3 downto 0) := (others => '0');
  signal s_mack              : std_logic_vector (3 downto 0) := (others => '0');
  signal s_almfull           : std_logic_vector (3 downto 0) := (others => '0');
  signal s_done_dil_rd       : std_logic_vector (3 downto 0) := (others => '0');
  signal s_empty_n           : std_logic_vector (3 downto 0) := (others => '0');
  signal s_no_adata_n        : std_logic_vector (3 downto 0) := (others => '0');
  signal s_enin_n            : std_logic_vector (3 downto 0) := (others => '0');
  signal s_enout_n           : std_logic_vector (19 downto 0) := (others => '0');

  signal s_gas_CLR           : std_logic := '0';
  signal s_gas_CLK           : std_logic := '0';
  signal s_gas_T_H           : std_logic := '0';

-- Bidirectional buses, 4x18 bit = 72
  signal s_data_from_dil     : std_logic_vector (71 downto 0) := (others => '0');
  signal s_data_to_dil       : std_logic_vector (71 downto 0) := (others => '0');
  signal s_databus_oe        : std_logic_vector (71 downto 0) := (others => '0');

  signal s_test              : std_logic := '0';
--------------------------------------------------------------------------------
------ Architecture begin ------------------------------------------------------

begin

  INST_CLK_RST: entity work.clock_generator
    port map (
      EXT_RST_i => s_ext_rst,
      RST_o     => s_rst,
      CLK_i     => PLL_50MHZ_CLK,
      CLK_o     => CLK10
    );

--------------------------------------------------------------------------------
------ Serial flash loader to write the .jic file to the memory ---------------

  sf_init: SerialFlashLoader
    port map(
      noe_in  => '0'
    );

--------------------------------------------------------------------------------
------ Main control ------------------------------------------------------------
  INST_MAIN_CTRL: entity work.main_ctrl
    port map (
      ARST           => s_rst,
      EXT_RST_o      => s_ext_rst,
      ------ LVDS ------
      LVDS_RX_i      => lvds_rx_in,
      LVDS_CLK_i     => clk_rx_in,
      ------ XCVR ------
      XCVR_TX_o      => gtx_tx0,
      XCVR_REF_CLK   => REF_156MHZ,
      ------ CLK ------
      CLK50_PLL      => PLL_50MHZ_CLK,
      CLK10          => CLK10,
      ------ Dilogic ------
      SUB_COMP_o     => s_sub_cmp,
      DIL_CLR_o      => s_dil_clr,
      DIL_RST_o      => s_dil_rst,
      STRIN_o        => s_strin,
      TRG_N_o        => s_trg_n,
      MACK_i         => s_mack,
      ALMFULL_i      => s_almfull,
      EMPTY_N_i     => s_empty_n,
      NO_ADATA_N_i  => s_no_adata_n,
      CLK_ADC_N_o   => s_dil_clk_adc,
      CLKD_N_o      => s_dil_clkd,
      ENIN_N_o      => s_enin_n,
      ENOUT_N_i     => s_enout_n,
      FCODE_o       => s_fcode_dil,
      CH_ADDR_N_o   => s_addr_gas_n,
      DATABUS_o     => s_data_to_dil,
      DATABUS_i     => s_data_from_dil,
      DIL_ENA_o     => s_dil_ena,
      ------ Gassiplex ------
      CLK_G_o       => s_gas_CLK,
      CLR_G_o       => s_gas_CLR,
      T_H_o         => s_gas_T_H,
      ------ Others ------
      TEST_PIN_o    => s_test
    );

-------------------------------------------------------------------------------
------ Databus bidirectional buffering ----------------------------------------

  INST_DATABUS_D1: for j in DATA_BUS_D1'range generate
    INST_IOBUF_DATABUSi: ALT_IOBUF
      port map (
        i   => s_data_to_dil(j),
        oe  => s_databus_oe(j),
        io  => DATA_BUS_D1(j),
        o   => s_data_from_dil(j)
      );
  end generate INST_DATABUS_D1;

  INST_DATABUS_D2: for j in DATA_BUS_D1'range generate
    INST_IOBUF_DATABUSi: ALT_IOBUF
      port map (
        i   => s_data_to_dil(18 + j),
        oe  => s_databus_oe(18 + j),
        io  => DATA_BUS_D2(j),
        o   => s_data_from_dil(18 + j)
      );
  end generate INST_DATABUS_D2;

  INST_DATABUS_D3: for j in DATA_BUS_D1'range generate
    INST_IOBUF_DATABUSi: ALT_IOBUF
      port map (
        i   => s_data_to_dil(36 + j),
        oe  => s_databus_oe(36 + j),
        io  => DATA_BUS_D3(j),
        o   => s_data_from_dil(36 + j)
      );
  end generate INST_DATABUS_D3;

  INST_DATABUS_D4: for j in DATA_BUS_D1'range generate
    INST_IOBUF_DATABUSi: ALT_IOBUF
      port map (
        i   => s_data_to_dil(54 + j),
        oe  => s_databus_oe(54 + j),
        io  => DATA_BUS_D4(j),
        o   => s_data_from_dil(54 + j)
      );
  end generate INST_DATABUS_D4;

  INST_DIL_ENA: for i in 0 to 3 generate
    INST_DATABUS_ENA: for j in DATA_BUS_D1'range generate
      s_databus_oe(18*i+j) <= s_dil_ena(i);
    end generate INST_DATABUS_ENA;
  end generate INST_DIL_ENA;

-------------------------------------------------------------------------------
------ Input and output buffers -----------------------------------------------
```

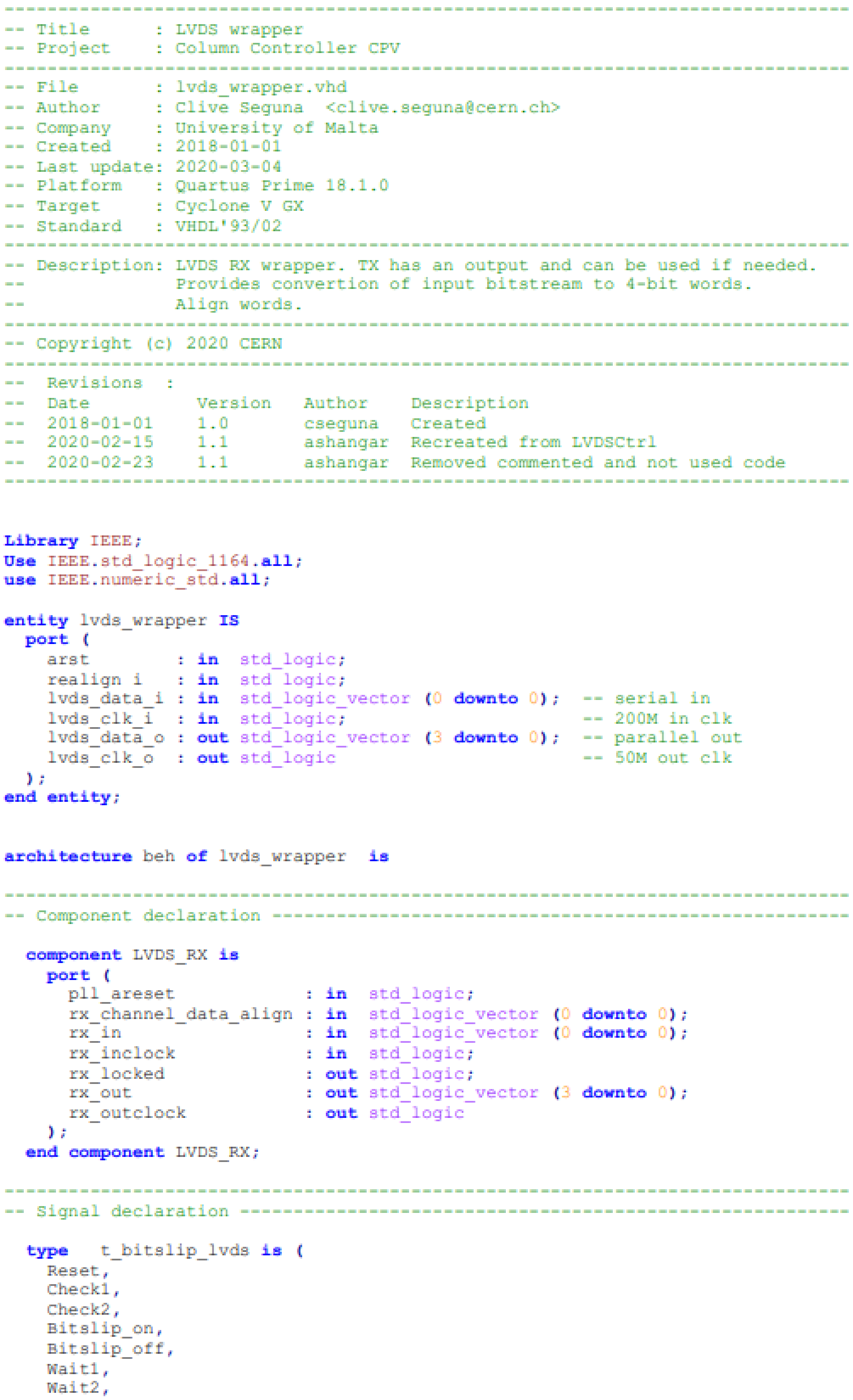


```
GEN_DIL_ENOUT: for i in ENOUT_N'range generate
  INST_ENOUT: ALT_INBUF
    port map (
      i => ENOUT_N(i),
      o => s_enout_n(i)
    );
end generate GEN_DIL_ENOUT;

GEN_DIL_INPUT: for i in 0 to 3 generate
  INST_EMPTY: ALT_INBUF
    port map (
      i => EMPTY_N(i),
      o => s_empty_n(i)
    );
  INST_ALMFULL: ALT_INBUF
    port map (
      i => ALMFULL(i),
      o => s_almfull(i)
    );
  INST_NOADATA: ALT_INBUF
    port map (
      i => NO_ADATA_N(i),
      o => s_no_adata_n(i)
    );
  INST_MACK: ALT_INBUF
    port map (
      i => MACK(i),
      o => s_mack(i)
    );
end generate GEN_DIL_INPUT;

GEN_DIL_OUTPUT: for i in 0 to 3 generate
  INST_TRG: ALT_OUTBUF
    port map (
      i => s_trg_n,
      o => TRG_N(i)
    );
  INST_CLK_ADC: ALT_OUTBUF
    port map (
      i => s_dil_clk_adc,
      o => CLK_ADC_N(i)
    );
  INST_CLKD: ALT_OUTBUF
    port map (
      i => s_dil_clkd,
      o => CLKD_N(i)
    );
  INST_CLR: ALT_OUTBUF
    port map (
      i => s_dil_clr,
      o => DIL_CLR(i)
    );
  INST_SUBCOMP: ALT_OUTBUF
    port map (
      i => s_sub_cmp,
      o => SUB_COMP(i)
    );
  INST_FCODE_C1: ALT_OUTBUF
    port map (
      i => s_fcode_dil(i),
      o => FCODE_C1(i)
    );
  INST_FCODE_C2: ALT_OUTBUF
    port map (
      i => s_fcode_dil(i),
      o => FCODE_C2(i)
    );
  INST_ENIN: ALT_OUTBUF
    port map (
      i => s_enin_n(i),
      o => ENIN_N(i)
```

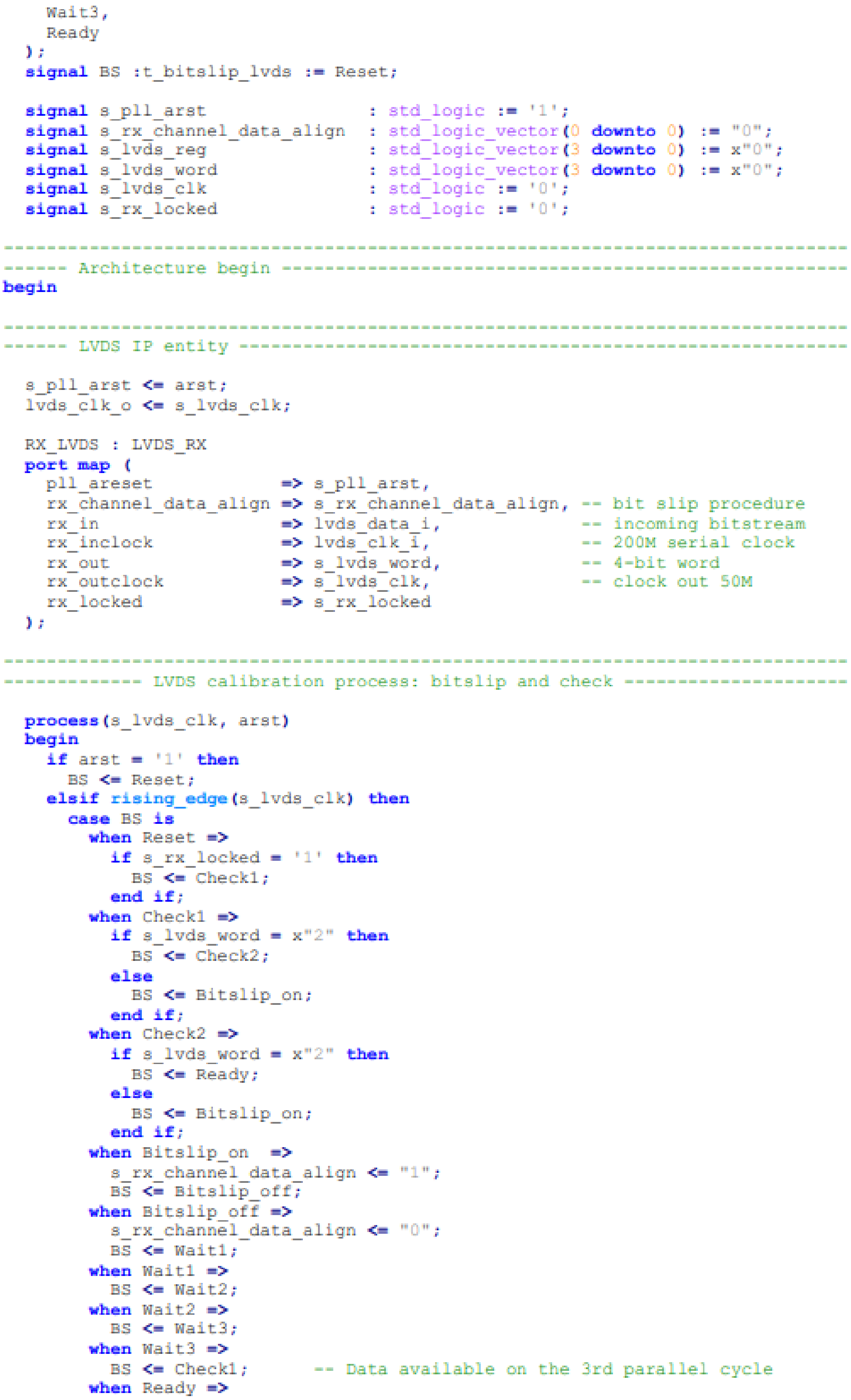


```
      );
    INST_STRIN: ALT_OUTBUF
      port map (
        i => s_strin(i),
        o => STRIN(i)
      );
    INST_DIL_RST: ALT_OUTBUF
      port map (
        i => s_dil_rst(i),
        o => DIL_RST(i)
      );
  end generate GEN_DIL_OUTPUT;

  GEN_DIL_ADDR: for i in 0 to 5 generate
    INST_ADDR_D1: ALT_OUTBUF
      port map (
        i => s_addr_gas_n(i),
        o => CH_ADDR_D1_N(i)
      );
    INST_ADDR_D2: ALT_OUTBUF
      port map (
        i => s_addr_gas_n(i),
        o => CH_ADDR_D2_N(i)
      );
    INST_ADDR_D3: ALT_OUTBUF
      port map (
        i => s_addr_gas_n(i),
        o => CH_ADDR_D3_N(i)
      );
    INST_ADDR_D4: ALT_OUTBUF
      port map (
        i => s_addr_gas_n(i),
        o => CH_ADDR_D4_N(i)
      );
  end generate GEN_DIL_ADDR;

  -- "10" means 3 Gassiplex per one Dilogic chip, 48 channels.
  DIL_GX         <= b"1010"; -- "10" for D1 & D2; "10" for D3 & D4

--------------------------------------------------------------------------------
------ 3-Gassiplex signals -----------------------------------------------------

  GEN_GAS_SIG: for i in 0 to 1 generate
    INST_CLKG: ALT_OUTBUF
    port map (
      i => s_gas_CLK,
      o => CLK_G(i)
    );
    INST_CLRG: ALT_OUTBUF
    port map (
      i => s_gas_CLR,
      o => CLR_G(i)
    );
    INST_T_H: ALT_OUTBUF
    port map (
      i => s_gas_T_H,
      o => T_H(i)
    );
  end generate GEN_GAS_SIG;

--------------------------------------------------------------------------------
------ Other signals -----------------------------------------------------------

  INST_TEST: ALT_OUTBUF
    port map (
      i => s_test,
      o => TEST_PIN
    );

end architecture;
```

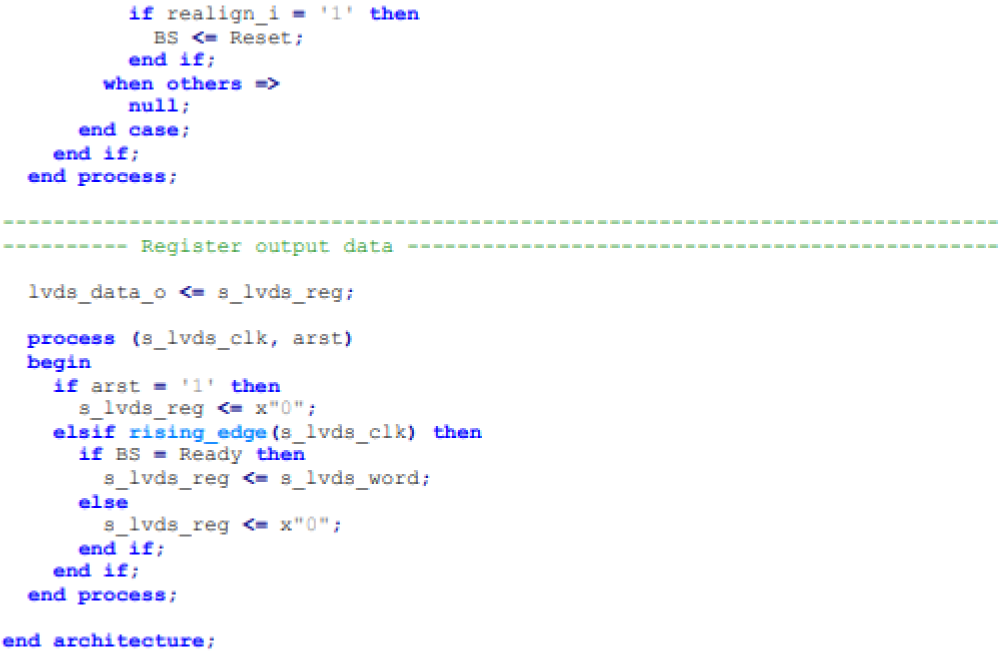
Appendix A.2 - Column Controller Firmware – lvds_wrapper.vhd



```
--------------------------------------------------------------------------------
-- Title      : LVDS wrapper
-- Project    : Column Controller CPV
--------------------------------------------------------------------------------
-- File       : lvds_wrapper.vhd
-- Author     : Clive Seguna  <clive.seguna@cern.ch>
-- Company    : University of Malta
-- Created    : 2018-01-01
-- Last update: 2020-03-04
-- Platform   : Quartus Prime 18.1.0
-- Target     : Cyclone V GX
-- Standard   : VHDL'93/02
--------------------------------------------------------------------------------
-- Description: LVDS RX wrapper. TX has an output and can be used if needed.
--              Provides convertion of input bitstream to 4-bit words.
--              Align words.
--------------------------------------------------------------------------------
-- Copyright (c) 2020 CERN
--------------------------------------------------------------------------------
--  Revisions  :
--  Date          Version   Author    Description
--  2018-01-01    1.0       cseguna   Created
--  2020-02-15    1.1       ashangar  Recreated from LVDSCtrl
--  2020-02-23    1.1       ashangar  Removed commented and not used code
--------------------------------------------------------------------------------


Library IEEE;
Use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity lvds_wrapper IS
  port (
    arst        : in  std_logic;
    realign_i   : in  std_logic;
    lvds_data_i : in  std_logic_vector (0 downto 0);   -- serial in
    lvds_clk_i  : in  std_logic;                       -- 200M in clk
    lvds_data_o : out std_logic_vector (3 downto 0);   -- parallel out
    lvds_clk_o  : out std_logic                        -- 50M out clk
  );
end entity;


architecture beh of lvds_wrapper  is

--------------------------------------------------------------------------------
-- Component declaration -------------------------------------------------------

  component LVDS_RX is
    port (
      pll_areset            : in  std_logic;
      rx_channel_data_align : in  std_logic_vector (0 downto 0);
      rx_in                 : in  std_logic_vector (0 downto 0);
      rx_inclock            : in  std_logic;
      rx_locked             : out std_logic;
      rx_out                : out std_logic_vector (3 downto 0);
      rx_outclock           : out std_logic
    );
  end component LVDS_RX;

--------------------------------------------------------------------------------
-- Signal declaration ----------------------------------------------------------

  type   t_bitslip_lvds is (
    Reset,
    Check1,
    Check2,
    Bitslip_on,
    Bitslip_off,
    Wait1,
    Wait2,
```

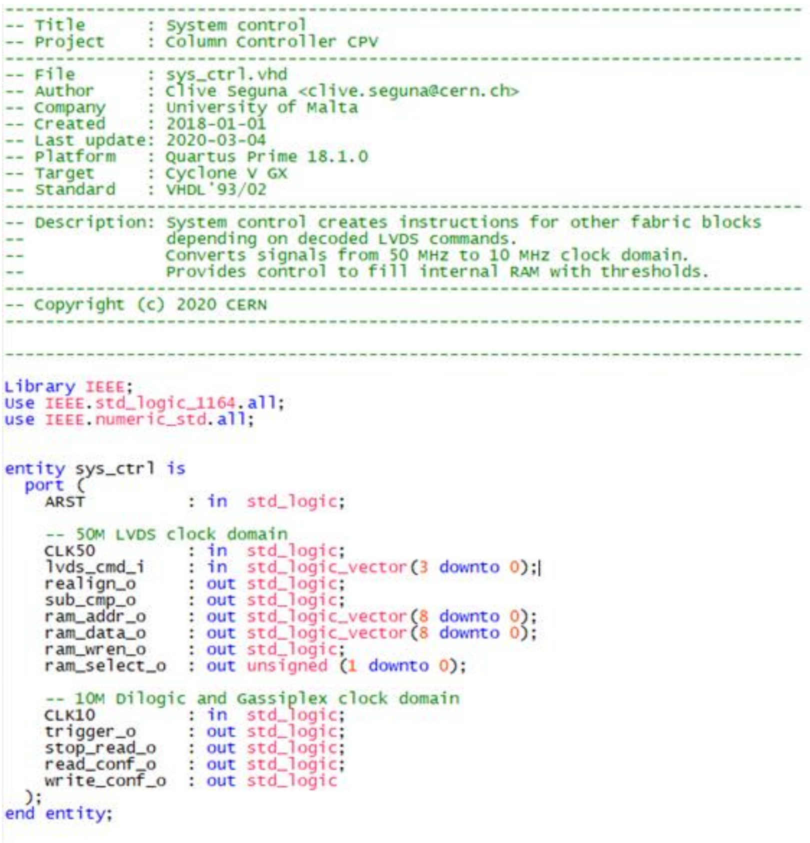


```
    Wait3,
    Ready
  );
  signal BS :t_bitslip_lvds := Reset;

  signal s_pll_arst                : std_logic := '1';
  signal s_rx_channel_data_align  : std_logic_vector(0 downto 0)  := "0";
  signal s_lvds_reg               : std_logic_vector(3 downto 0)  := x"0";
  signal s_lvds_word              : std_logic_vector(3 downto 0)  := x"0";
  signal s_lvds_clk               : std_logic := '0';
  signal s_rx_locked              : std_logic := '0';

--------------------------------------------------------------------------------
------ Architecture begin ------------------------------------------------------
begin

--------------------------------------------------------------------------------
------ LVDS IP entity ----------------------------------------------------------

  s_pll_arst <= arst;
  lvds_clk_o <= s_lvds_clk;

  RX_LVDS : LVDS_RX
  port map (
    pll_areset            => s_pll_arst,
    rx_channel_data_align => s_rx_channel_data_align, -- bit slip procedure
    rx_in                 => lvds_data_i,             -- incoming bitstream
    rx_inclock            => lvds_clk_i,              -- 200M serial clock
    rx_out                => s_lvds_word,             -- 4-bit word
    rx_outclock           => s_lvds_clk,              -- clock out 50M
    rx_locked             => s_rx_locked
  );

--------------------------------------------------------------------------------
------------- LVDS calibration process: bitslip and check ----------------------

  process(s_lvds_clk, arst)
  begin
    if arst = '1' then
      BS <= Reset;
    elsif rising_edge(s_lvds_clk) then
      case BS is
        when Reset =>
          if s_rx_locked = '1' then
            BS <= Check1;
          end if;
        when Check1 =>
          if s_lvds_word = x"2" then
            BS <= Check2;
          else
            BS <= Bitslip_on;
          end if;
        when Check2 =>
          if s_lvds_word = x"2" then
            BS <= Ready;
          else
            BS <= Bitslip_on;
          end if;
        when Bitslip_on  =>
          s_rx_channel_data_align <= "1";
          BS <= Bitslip_off;
        when Bitslip_off =>
          s_rx_channel_data_align <= "0";
          BS <= Wait1;
        when Wait1 =>
          BS <= Wait2;
        when Wait2 =>
          BS <= Wait3;
        when Wait3 =>
          BS <= Check1;      -- Data available on the 3rd parallel cycle
        when Ready =>
```

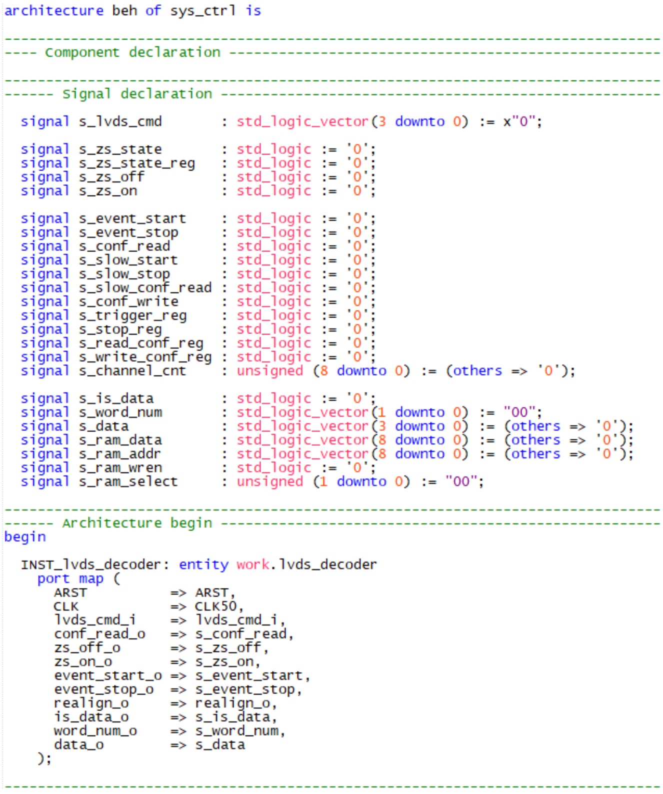


```vhdl
          if realign_i = '1' then
            BS <= Reset;
          end if;
        when others =>
          null;
      end case;
    end if;
  end process;

------------------------------------------------------------------------------
---------- Register output data ----------------------------------------------

  lvds_data_o <= s_lvds_reg;

  process (s_lvds_clk, arst)
  begin
    if arst = '1' then
      s_lvds_reg <= x"0";
    elsif rising_edge(s_lvds_clk) then
      if BS = Ready then
        s_lvds_reg <= s_lvds_word;
      else
        s_lvds_reg <= x"0";
      end if;
    end if;
  end process;

end architecture;
```

Appendix A.3 - Column Controller Firmware – sys_ctrl.vhd

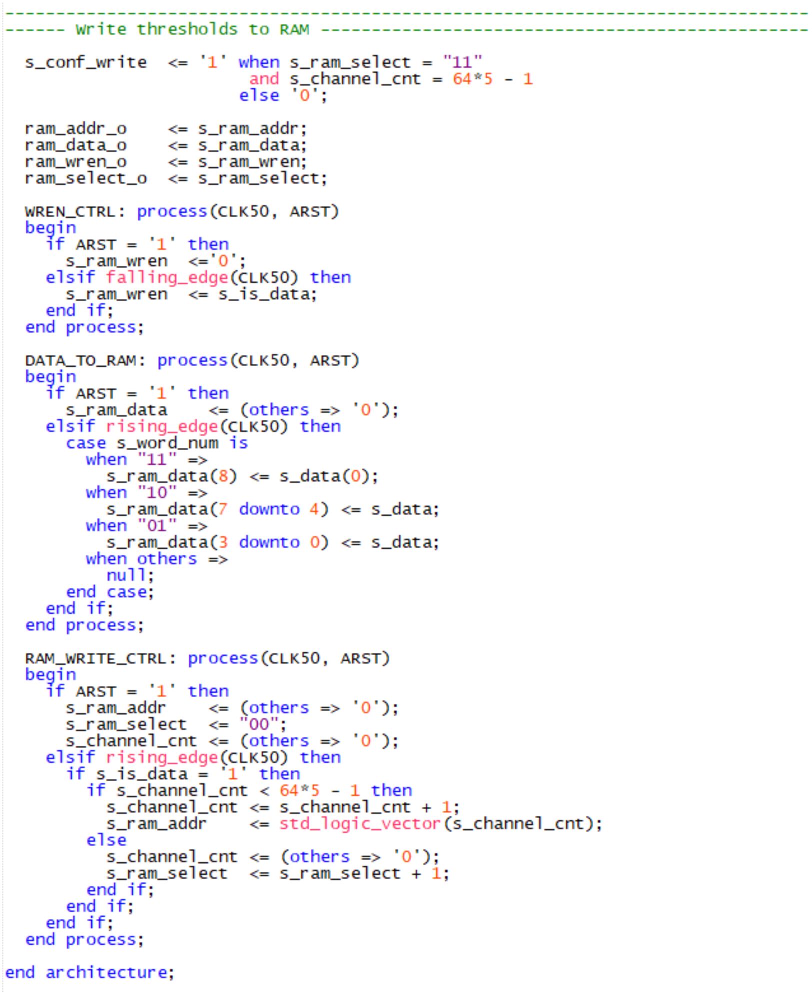


```vhdl
-------------------------------------------------------------------------------
-- Title      : System control
-- Project    : Column Controller CPV
-------------------------------------------------------------------------------
-- File       : sys_ctrl.vhd
-- Author     : Clive Seguna <clive.seguna@cern.ch>
-- Company    : University of Malta
-- Created    : 2018-01-01
-- Last update: 2020-03-04
-- Platform   : Quartus Prime 18.1.0
-- Target     : Cyclone V GX
-- Standard   : VHDL'93/02
-------------------------------------------------------------------------------
-- Description: System control creates instructions for other fabric blocks
--              depending on decoded LVDS commands.
--              Converts signals from 50 MHz to 10 MHz clock domain.
--              Provides control to fill internal RAM with thresholds.
-------------------------------------------------------------------------------
-- Copyright (c) 2020 CERN
-------------------------------------------------------------------------------

-------------------------------------------------------------------------------

Library IEEE;
Use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;


entity sys_ctrl is
  port (
    ARST          : in  std_logic;

    -- 50M LVDS clock domain
    CLK50         : in  std_logic;
    lvds_cmd_i    : in  std_logic_vector(3 downto 0);|
    realign_o     : out std_logic;
    sub_cmp_o     : out std_logic;
    ram_addr_o    : out std_logic_vector(8 downto 0);
    ram_data_o    : out std_logic_vector(8 downto 0);
    ram_wren_o    : out std_logic;
    ram_select_o  : out unsigned (1 downto 0);

    -- 10M Dilogic and Gassiplex clock domain
    CLK10         : in  std_logic;
    trigger_o     : out std_logic;
    stop_read_o   : out std_logic;
    read_conf_o   : out std_logic;
    write_conf_o  : out std_logic
  );
end entity;
```



```vhdl
architecture beh of sys_ctrl is

-------------------------------------------------------------------------------
---- Component declaration ----------------------------------------------------

-------------------------------------------------------------------------------
------ Signal declaration -----------------------------------------------------

  signal s_lvds_cmd       : std_logic_vector(3 downto 0) := x"0";

  signal s_zs_state       : std_logic := '0';
  signal s_zs_state_reg   : std_logic := '0';
  signal s_zs_off         : std_logic := '0';
  signal s_zs_on          : std_logic := '0';

  signal s_event_start    : std_logic := '0';
  signal s_event_stop     : std_logic := '0';
  signal s_conf_read      : std_logic := '0';
  signal s_slow_start     : std_logic := '0';
  signal s_slow_stop      : std_logic := '0';
  signal s_slow_conf_read : std_logic := '0';
  signal s_conf_write     : std_logic := '0';
  signal s_trigger_reg    : std_logic := '0';
  signal s_stop_reg       : std_logic := '0';
  signal s_read_conf_reg  : std_logic := '0';
  signal s_write_conf_reg : std_logic := '0';
  signal s_channel_cnt    : unsigned (8 downto 0) := (others => '0');

  signal s_is_data        : std_logic := '0';
  signal s_word_num       : std_logic_vector(1 downto 0) := "00";
  signal s_data           : std_logic_vector(3 downto 0) := (others => '0');
  signal s_ram_data       : std_logic_vector(8 downto 0) := (others => '0');
  signal s_ram_addr       : std_logic_vector(8 downto 0) := (others => '0');
  signal s_ram_wren       : std_logic := '0';
  signal s_ram_select     : unsigned (1 downto 0) := "00";

-------------------------------------------------------------------------------
------ Architecture begin -----------------------------------------------------
begin

  INST_lvds_decoder: entity work.lvds_decoder
    port map (
      ARST          => ARST,
      CLK           => CLK50,
      lvds_cmd_i    => lvds_cmd_i,
      conf_read_o   => s_conf_read,
      zs_off_o      => s_zs_off,
      zs_on_o       => s_zs_on,
      event_start_o => s_event_start,
      event_stop_o  => s_event_stop,
      realign_o     => realign_o,
      is_data_o     => s_is_data,
      word_num_o    => s_word_num,
      data_o        => s_data
    );

-------------------------------------------------------------------------------
```

```
------------------------------------------------------------------------------
------ Zero suppression -------------------------------------------------------

  s_zs_state_reg <= s_zs_state when rising_edge(CLK50);
  s_zs_state <= '1' when s_zs_on  = '1' else
                '0' when s_zs_off = '1' else
                s_zs_state_reg;
  sub_cmp_o <= s_zs_state;

------------------------------------------------------------------------------
------ Create 100 ns START pulse ----------------------------------------------

  START_HOLD: process(CLK50, ARST)
    variable v_start_cnt : natural range 0 to 7 := 0;
  begin
    if ARST = '1' then
      s_slow_start    <= '0';
      v_start_cnt     := 0;
    elsif rising_edge(CLK50) then
      if s_event_start = '1' then
        v_start_cnt   := 5;
      end if;
      if v_start_cnt > 0 then
        v_start_cnt   := v_start_cnt - 1;
        s_slow_start  <= '1';
      else
        s_slow_start  <= '0';
      end if;
    end if;
  end process START_HOLD;

------------------------------------------------------------------------------
------ Create 100 ns STOP pulse -----------------------------------------------

  STOP_HOLD: process(CLK50, ARST)
    variable v_stop_cnt : natural range 0 to 7 := 0;
  begin
    if ARST = '1' then
      s_slow_stop   <= '0';
      v_stop_cnt    := 0;
    elsif rising_edge(CLK50) then
      if s_event_stop = '1' then
        v_stop_cnt  := 5;
      end if;
      if v_stop_cnt > 0 then
        v_stop_cnt  := v_stop_cnt - 1;
        s_slow_stop <= '1';
      else
        s_slow_stop <= '0';
      end if;
    end if;
  end process STOP_HOLD;


--------------------------------------------------------------------------------
------ Create 100 ns CONF_READ pulse -------------------------------------------

  CONF_READ_HOLD: process(CLK50, ARST)
    variable v_conf_read_cnt : natural range 0 to 7 := 0;
  begin
    if ARST = '1' then
      s_slow_conf_read  <= '0';
      v_conf_read_cnt   := 0;
    elsif rising_edge(CLK50) then
      if s_conf_read = '1' then
        v_conf_read_cnt := 5;
      end if;
      if v_conf_read_cnt > 0 then
        v_conf_read_cnt := v_conf_read_cnt - 1;
        s_slow_conf_read <= '1';
      else
        s_slow_conf_read <= '0';
      end if;
    end if;
  end process CONF_READ_HOLD;

--------------------------------------------------------------------------------
------ Clock domain crossing 50M -> 10M ----------------------------------------

  CDC_TRG: process(CLK10, ARST)
  begin
    if ARST = '1' then
      s_trigger_reg     <= '0';
      s_stop_reg        <= '0';
      s_read_conf_reg   <= '0';
      s_write_conf_reg  <= '0';
    elsif rising_edge(CLK10) then
      s_trigger_reg     <= s_slow_start;
      s_stop_reg        <= s_slow_stop;
      s_read_conf_reg   <= s_slow_conf_read;
      s_write_conf_reg  <= s_conf_write;
    end if;
  end process CDC_TRG;

  trigger_o     <= s_trigger_reg;
  stop_read_o   <= s_stop_reg;
  read_conf_o   <= s_read_conf_reg;
  write_conf_o  <= s_write_conf_reg;
```

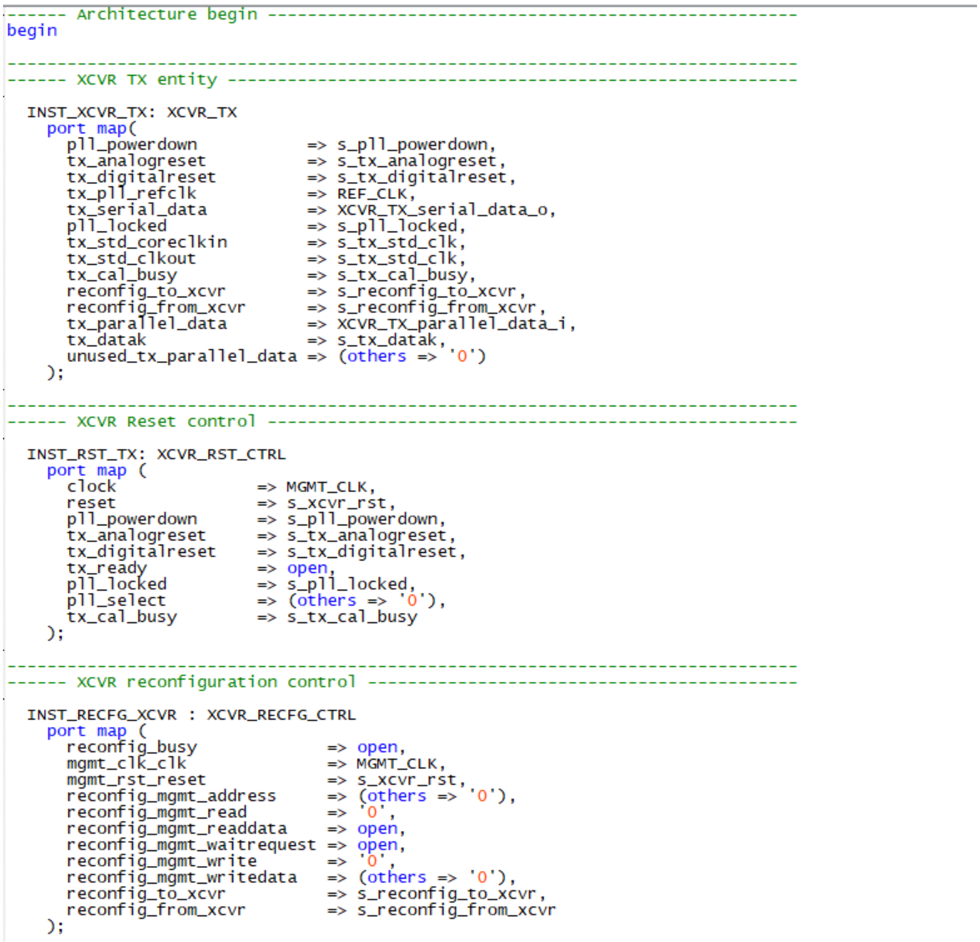


```vhdl
--------------------------------------------------------------------------------
------ Write thresholds to RAM -------------------------------------------------

  s_conf_write  <= '1' when s_ram_select = "11"
                        and s_channel_cnt = 64*5 - 1
                       else '0';

  ram_addr_o    <= s_ram_addr;
  ram_data_o    <= s_ram_data;
  ram_wren_o    <= s_ram_wren;
  ram_select_o  <= s_ram_select;

  WREN_CTRL: process(CLK50, ARST)
  begin
    if ARST = '1' then
      s_ram_wren  <='0';
    elsif falling_edge(CLK50) then
      s_ram_wren  <= s_is_data;
    end if;
  end process;

  DATA_TO_RAM: process(CLK50, ARST)
  begin
    if ARST = '1' then
      s_ram_data    <= (others => '0');
    elsif rising_edge(CLK50) then
      case s_word_num is
        when "11" =>
          s_ram_data(8) <= s_data(0);
        when "10" =>
          s_ram_data(7 downto 4) <= s_data;
        when "01" =>
          s_ram_data(3 downto 0) <= s_data;
        when others =>
          null;
      end case;
    end if;
  end process;

  RAM_WRITE_CTRL: process(CLK50, ARST)
  begin
    if ARST = '1' then
      s_ram_addr    <= (others => '0');
      s_ram_select  <= "00";
      s_channel_cnt <= (others => '0');
    elsif rising_edge(CLK50) then
      if s_is_data = '1' then
        if s_channel_cnt < 64*5 - 1 then
          s_channel_cnt <= s_channel_cnt + 1;
          s_ram_addr    <= std_logic_vector(s_channel_cnt);
        else
          s_channel_cnt <= (others => '0');
          s_ram_select  <= s_ram_select + 1;
        end if;
      end if;
    end if;
  end process;

end architecture;
```

Appendix A.4 - Column Controller Firmware – xcvr_wrapper.vhd

```
-------------------------------------------------------------------------------
-- Title      : XCVR wrapper
-- Project    : Column Controller CPV
-------------------------------------------------------------------------------
-- File       : xcvr_wrapper.vhd
-- Author     : Clive Seguna  <clive.seguna@cern.ch>
-- Company    : University of Malta
-- Created    : 2018-01-01
-- Last update: 2020-02-23
-- Platform   : Quartus Prime 18.1.0
-- Target     : Cyclone V GX
-- Standard   : VHDL'93/02
-------------------------------------------------------------------------------
-- Description: XCVR TX wrapper. RX has an input and can be used if needed.
--              Provides convertion of 32-bit words to serial bitstream.
--              Includes reconfiguration and reset.
-------------------------------------------------------------------------------
-- Copyright (c) 2020 CERN
-------------------------------------------------------------------------------


Library IEEE;
Use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;


entity XCVR_WRAPPER is
  port (
    ARST                      : in  std_logic;
    MGMT_CLK                  : in  std_logic;                            -- 50 MHz
    REF_CLK                   : in  std_logic_vector (0 downto 0);        -- 156.25 MHz

    XCVR_TX_parallel_data_i : in  std_logic_vector (31 downto 0);
    XCVR_TX_serial_data_o   : out std_logic_vector (0 downto 0);
    XCVR_TX_parallel_clk_o  : out std_logic_vector (0 downto 0)       -- 78.125 MHz
  );
end entity XCVR_WRAPPER;

architecture structural of XCVR_WRAPPER is

-------------------------------------------------------------------------------
------ Component declaration --------------------------------------------------

  component XCVR_TX is
    port (
      pll_powerdown           : in  std_logic_vector(0 downto 0);
      tx_analogreset          : in  std_logic_vector(0 downto 0);
      tx_digitalreset         : in  std_logic_vector(0 downto 0);
      tx_pll_refclk           : in  std_logic_vector(0 downto 0);
      tx_serial_data          : out std_logic_vector(0 downto 0);
      pll_locked              : out std_logic_vector(0 downto 0);
      tx_std_coreclkin        : in  std_logic_vector(0 downto 0);
      tx_std_clkout           : out std_logic_vector(0 downto 0);
      tx_cal_busy             : out std_logic_vector(0 downto 0);
      reconfig_to_xcvr        : in  std_logic_vector(139 downto 0);
      reconfig_from_xcvr      : out std_logic_vector(91 downto 0);
      tx_parallel_data        : in  std_logic_vector(31 downto 0);
      tx_datak                : in  std_logic_vector(3 downto 0);
      unused_tx_parallel_data : in  std_logic_vector(7 downto 0)
    );
  end component XCVR_TX;
```

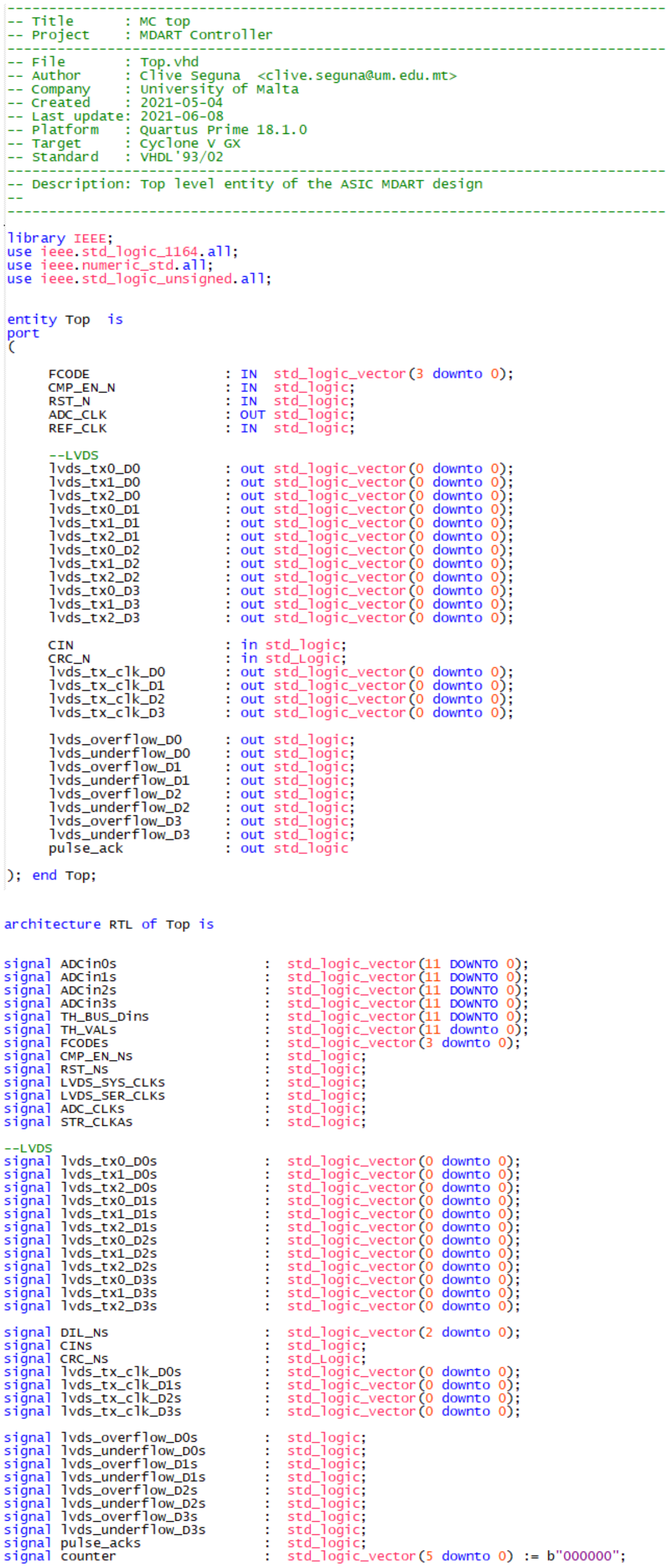
```
component XCVR_RST_CTRL is
  port (
    clock               : in  std_logic;
    reset               : in  std_logic;
    pll_powerdown       : out std_logic_vector(0 downto 0);
    tx_analogreset      : out std_logic_vector(0 downto 0);
    tx_digitalreset     : out std_logic_vector(0 downto 0);
    tx_ready            : out std_logic_vector(0 downto 0);
    pll_locked          : in  std_logic_vector(0 downto 0);
    pll_select          : in  std_logic_vector(0 downto 0);
    tx_cal_busy         : in  std_logic_vector(0 downto 0)
  );
end component XCVR_RST_CTRL;

component XCVR_RECFG_CTRL is
  port (
    reconfig_busy              : out std_logic;
    mgmt_clk_clk               : in  std_logic;
    mgmt_rst_reset             : in  std_logic;
    reconfig_mgmt_address      : in  std_logic_vector(6 downto 0);
    reconfig_mgmt_read         : in  std_logic;
    reconfig_mgmt_readdata     : out std_logic_vector(31 downto 0);
    reconfig_mgmt_waitrequest  : out std_logic;
    reconfig_mgmt_write        : in  std_logic;
    reconfig_mgmt_writedata    : in  std_logic_vector(31 downto 0);
    reconfig_to_xcvr           : out std_logic_vector(139 downto 0);
    reconfig_from_xcvr         : in  std_logic_vector(91 downto 0)
  );
end component XCVR_RECFG_CTRL;

----------------------------------------------------------------------------
----- Signal declaration ---------------------------------------------------

signal s_xcvr_rst          : std_logic :='0';
signal s_scvr_rst_cnt      : unsigned (3 downto 0) := x"0";

signal s_pll_powerdown     : std_logic_vector(0 downto 0)    := (others => '0');
signal s_tx_analogreset    : std_logic_vector(0 downto 0)    := (others => '0');
signal s_tx_digitalreset   : std_logic_vector(0 downto 0)    := (others => '0');
signal s_tx_serial_data    : std_logic_vector(0 downto 0)    := (others => '0');
signal s_pll_locked        : std_logic_vector(0 downto 0)    := (others => '0');
signal s_tx_std_clk        : std_logic_vector(0 downto 0)    := (others => '0');
signal s_tx_cal_busy       : std_logic_vector(0 downto 0)    := (others => '0');
signal s_reconfig_to_xcvr  : std_logic_vector(139 downto 0)  := (others => '0');
signal s_reconfig_from_xcvr : std_logic_vector(91 downto 0)  := (others => '0');
signal s_tx_parallel_data  : std_logic_vector(31 downto 0)   := (others => '0');
signal s_tx_datak          : std_logic_vector(3 downto 0)    := (others => '0');

----------------------------------------------------------------------------
```



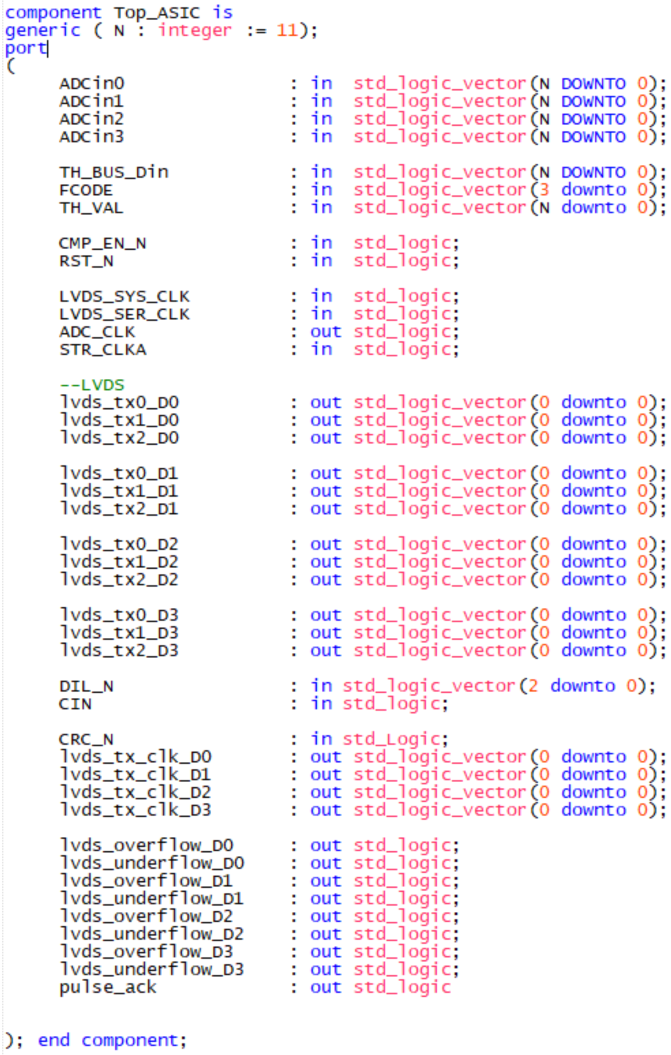
```
------ Architecture begin -------------------------------------------------
begin

---------------------------------------------------------------------------
------ XCVR TX entity -----------------------------------------------------

  INST_XCVR_TX: XCVR_TX
    port map(
      pll_powerdown           => s_pll_powerdown,
      tx_analogreset          => s_tx_analogreset,
      tx_digitalreset         => s_tx_digitalreset,
      tx_pll_refclk           => REF_CLK,
      tx_serial_data          => XCVR_TX_serial_data_o,
      pll_locked              => s_pll_locked,
      tx_std_coreclkin        => s_tx_std_clk,
      tx_std_clkout           => s_tx_std_clk,
      tx_cal_busy             => s_tx_cal_busy,
      reconfig_to_xcvr        => s_reconfig_to_xcvr,
      reconfig_from_xcvr      => s_reconfig_from_xcvr,
      tx_parallel_data        => XCVR_TX_parallel_data_i,
      tx_datak                => s_tx_datak,
      unused_tx_parallel_data => (others => '0')
    );

---------------------------------------------------------------------------
------ XCVR Reset control -------------------------------------------------

  INST_RST_TX: XCVR_RST_CTRL
    port map (
      clock               => MGMT_CLK,
      reset               => s_xcvr_rst,
      pll_powerdown       => s_pll_powerdown,
      tx_analogreset      => s_tx_analogreset,
      tx_digitalreset     => s_tx_digitalreset,
      tx_ready            => open,
      pll_locked          => s_pll_locked,
      pll_select          => (others => '0'),
      tx_cal_busy         => s_tx_cal_busy
    );

---------------------------------------------------------------------------
------ XCVR reconfiguration control ---------------------------------------

  INST_RECFG_XCVR : XCVR_RECFG_CTRL
    port map (
      reconfig_busy              => open,
      mgmt_clk_clk               => MGMT_CLK,
      mgmt_rst_reset             => s_xcvr_rst,
      reconfig_mgmt_address      => (others => '0'),
      reconfig_mgmt_read         => '0',
      reconfig_mgmt_readdata     => open,
      reconfig_mgmt_waitrequest  => open,
      reconfig_mgmt_write        => '0',
      reconfig_mgmt_writedata    => (others => '0'),
      reconfig_to_xcvr           => s_reconfig_to_xcvr,
      reconfig_from_xcvr         => s_reconfig_from_xcvr
    );
```

```vhdl
--------------------------------------------------------------------------------
------ Reset and other signals -------------------------------------------------

  process (MGMT_CLK, ARST)
  begin
    if ARST = '1' then
      s_scvr_rst_cnt <= x"0";
    elsif rising_edge(MGMT_CLK) then
      if s_scvr_rst_cnt < x"C" then
        s_scvr_rst_cnt <= s_scvr_rst_cnt  + 1;
      end if;
    end if;
  end process;

  s_xcvr_rst  <= '1' when s_scvr_rst_cnt > x"5" and s_scvr_rst_cnt < x"9" else '0';

  XCVR_TX_parallel_clk_o  <= s_tx_std_clk;

  s_tx_datak <= "0101" when XCVR_TX_parallel_data_i = X"C5BCC5BC" else "0000" ;

  -- s_tx_datak is data/control indicator of the byte sent.
  -- "0" means data; "1" means control.

end architecture;
```

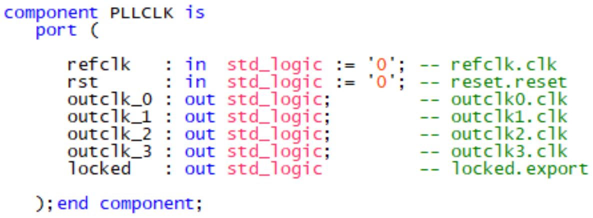
Appendix B.1 – VHDL code for MDART ASIC - Top.vhd



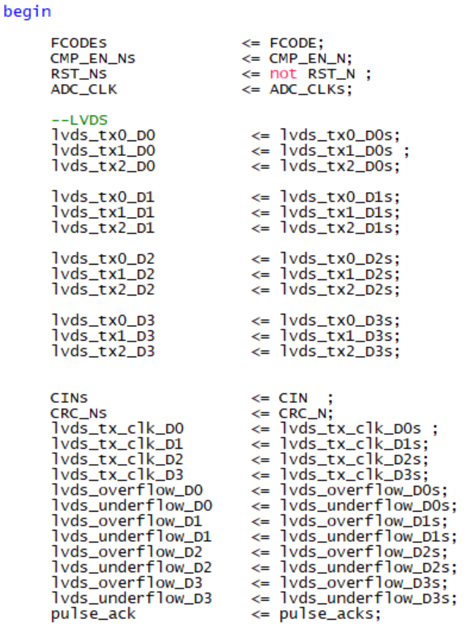
```vhdl
--------------------------------------------------------------------------------
-- Title       : MC top
-- Project     : MDART Controller
--------------------------------------------------------------------------------
-- File        : Top.vhd
-- Author      : Clive Seguna  <clive.seguna@um.edu.mt>
-- Company     : University of Malta
-- Created     : 2021-05-04
-- Last update: 2021-06-08
-- Platform    : Quartus Prime 18.1.0
-- Target      : Cyclone V GX
-- Standard    : VHDL'93/02
--------------------------------------------------------------------------------
-- Description: Top level entity of the ASIC MDART design
--
--------------------------------------------------------------------------------

library IEEE;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.std_logic_unsigned.all;


entity Top  is
port
(

     FCODE                 : IN  std_logic_vector(3 downto 0);
     CMP_EN_N              : IN  std_logic;
     RST_N                 : IN  std_logic;
     ADC_CLK               : OUT std_logic;
     REF_CLK               : IN  std_logic;

     --LVDS
     lvds_tx0_D0           : out std_logic_vector(0 downto 0);
     lvds_tx1_D0           : out std_logic_vector(0 downto 0);
     lvds_tx2_D0           : out std_logic_vector(0 downto 0);
     lvds_tx0_D1           : out std_logic_vector(0 downto 0);
     lvds_tx1_D1           : out std_logic_vector(0 downto 0);
     lvds_tx2_D1           : out std_logic_vector(0 downto 0);
     lvds_tx0_D2           : out std_logic_vector(0 downto 0);
     lvds_tx1_D2           : out std_logic_vector(0 downto 0);
     lvds_tx2_D2           : out std_logic_vector(0 downto 0);
     lvds_tx0_D3           : out std_logic_vector(0 downto 0);
     lvds_tx1_D3           : out std_logic_vector(0 downto 0);
     lvds_tx2_D3           : out std_logic_vector(0 downto 0);

     CIN                   : in std_logic;
     CRC_N                 : in std_Logic;
     lvds_tx_clk_D0        : out std_logic_vector(0 downto 0);
     lvds_tx_clk_D1        : out std_logic_vector(0 downto 0);
     lvds_tx_clk_D2        : out std_logic_vector(0 downto 0);
     lvds_tx_clk_D3        : out std_logic_vector(0 downto 0);

     lvds_overflow_D0      : out std_logic;
     lvds_underflow_D0     : out std_logic;
     lvds_overflow_D1      : out std_logic;
     lvds_underflow_D1     : out std_logic;
     lvds_overflow_D2      : out std_logic;
     lvds_underflow_D2     : out std_logic;
     lvds_overflow_D3      : out std_logic;
     lvds_underflow_D3     : out std_logic;
     pulse_ack             : out std_logic

);  end Top;


architecture RTL of Top is


signal ADCin0s                  :  std_logic_vector(11 DOWNTO 0);
signal ADCin1s                  :  std_logic_vector(11 DOWNTO 0);
signal ADCin2s                  :  std_logic_vector(11 DOWNTO 0);
signal ADCin3s                  :  std_logic_vector(11 DOWNTO 0);
signal TH_BUS_Dins              :  std_logic_vector(11 DOWNTO 0);
signal TH_VALS                  :  std_logic_vector(11 downto 0);
signal FCODES                   :  std_logic_vector(3 downto 0);
signal CMP_EN_Ns                :  std_logic;
signal RST_Ns                   :  std_logic;
signal LVDS_SYS_CLKs            :  std_logic;
signal LVDS_SER_CLKs            :  std_logic;
signal ADC_CLKs                 :  std_logic;
signal STR_CLKAs                :  std_logic;

--LVDS
signal lvds_tx0_D0s             :  std_logic_vector(0 downto 0);
signal lvds_tx1_D0s             :  std_logic_vector(0 downto 0);
signal lvds_tx2_D0s             :  std_logic_vector(0 downto 0);
signal lvds_tx0_D1s             :  std_logic_vector(0 downto 0);
signal lvds_tx1_D1s             :  std_logic_vector(0 downto 0);
signal lvds_tx2_D1s             :  std_logic_vector(0 downto 0);
signal lvds_tx0_D2s             :  std_logic_vector(0 downto 0);
signal lvds_tx1_D2s             :  std_logic_vector(0 downto 0);
signal lvds_tx2_D2s             :  std_logic_vector(0 downto 0);
signal lvds_tx0_D3s             :  std_logic_vector(0 downto 0);
signal lvds_tx1_D3s             :  std_logic_vector(0 downto 0);
signal lvds_tx2_D3s             :  std_logic_vector(0 downto 0);

signal DIL_Ns                   :  std_logic_vector(2 downto 0);
signal CINs                     :  std_logic;
signal CRC_Ns                   :  std_Logic;
signal lvds_tx_clk_D0s          :  std_logic_vector(0 downto 0);
signal lvds_tx_clk_D1s          :  std_logic_vector(0 downto 0);
signal lvds_tx_clk_D2s          :  std_logic_vector(0 downto 0);
signal lvds_tx_clk_D3s          :  std_logic_vector(0 downto 0);

signal lvds_overflow_D0s        :  std_logic;
signal lvds_underflow_D0s       :  std_logic;
signal lvds_overflow_D1s        :  std_logic;
signal lvds_underflow_D1s       :  std_logic;
signal lvds_overflow_D2s        :  std_logic;
signal lvds_underflow_D2s       :  std_logic;
signal lvds_overflow_D3s        :  std_logic;
signal lvds_underflow_D3s       :  std_logic;
signal pulse_acks               :  std_logic;
signal counter                  :  std_logic_vector(5 downto 0) := b"000000";
```



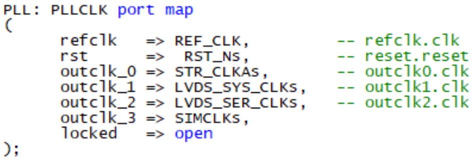
```
component Top_ASIC is
generic ( N : integer := 11);
port
(
     ADCin0                   : in  std_logic_vector(N DOWNTO 0);
     ADCin1                   : in  std_logic_vector(N DOWNTO 0);
     ADCin2                   : in  std_logic_vector(N DOWNTO 0);
     ADCin3                   : in  std_logic_vector(N DOWNTO 0);

     TH_BUS_Din               : in  std_logic_vector(N DOWNTO 0);
     FCODE                    : in  std_logic_vector(3 downto 0);
     TH_VAL                   : in  std_logic_vector(N downto 0);

     CMP_EN_N                 : in  std_logic;
     RST_N                    : in  std_logic;

     LVDS_SYS_CLK             : in  std_logic;
     LVDS_SER_CLK             : in  std_logic;
     ADC_CLK                  : out std_logic;
     STR_CLKA                 : in  std_logic;

     --LVDS
     lvds_tx0_D0              : out std_logic_vector(0 downto 0);
     lvds_tx1_D0              : out std_logic_vector(0 downto 0);
     lvds_tx2_D0              : out std_logic_vector(0 downto 0);

     lvds_tx0_D1              : out std_logic_vector(0 downto 0);
     lvds_tx1_D1              : out std_logic_vector(0 downto 0);
     lvds_tx2_D1              : out std_logic_vector(0 downto 0);

     lvds_tx0_D2              : out std_logic_vector(0 downto 0);
     lvds_tx1_D2              : out std_logic_vector(0 downto 0);
     lvds_tx2_D2              : out std_logic_vector(0 downto 0);

     lvds_tx0_D3              : out std_logic_vector(0 downto 0);
     lvds_tx1_D3              : out std_logic_vector(0 downto 0);
     lvds_tx2_D3              : out std_logic_vector(0 downto 0);

     DIL_N                    : in std_logic_vector(2 downto 0);
     CIN                      : in std_logic;

     CRC_N                    : in std_Logic;
     lvds_tx_clk_D0           : out std_logic_vector(0 downto 0);
     lvds_tx_clk_D1           : out std_logic_vector(0 downto 0);
     lvds_tx_clk_D2           : out std_logic_vector(0 downto 0);
     lvds_tx_clk_D3           : out std_logic_vector(0 downto 0);

     lvds_overflow_D0         : out std_logic;
     lvds_underflow_D0        : out std_logic;
     lvds_overflow_D1         : out std_logic;
     lvds_underflow_D1        : out std_logic;
     lvds_overflow_D2         : out std_logic;
     lvds_underflow_D2        : out std_logic;
     lvds_overflow_D3         : out std_logic;
     lvds_underflow_D3        : out std_logic;
     pulse_ack                : out std_logic

); end component;
```



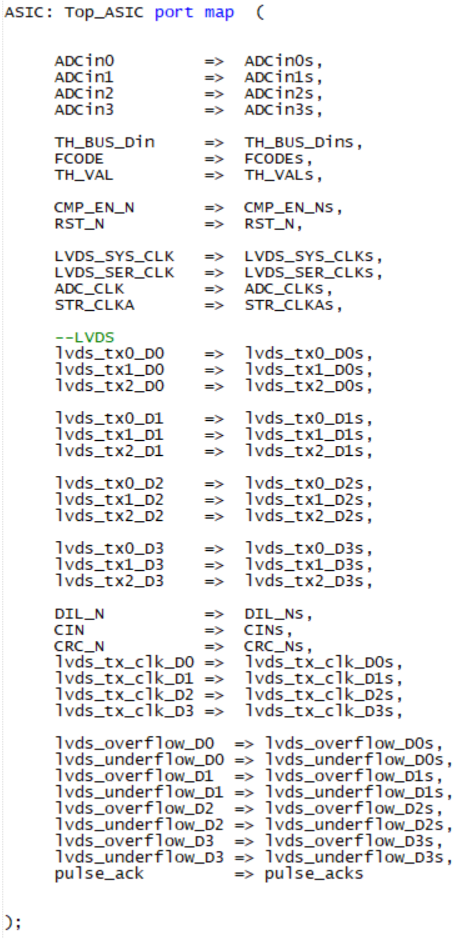
```
component PLLCLK is
   port (

      refclk   : in  std_logic := '0'; -- refclk.clk
      rst      : in  std_logic := '0'; -- reset.reset
      outclk_0 : out std_logic;        -- outclk0.clk
      outclk_1 : out std_logic;        -- outclk1.clk
      outclk_2 : out std_logic;        -- outclk2.clk
      outclk_3 : out std_logic;        -- outclk3.clk
      locked   : out std_logic         -- locked.export

   );end component;
```



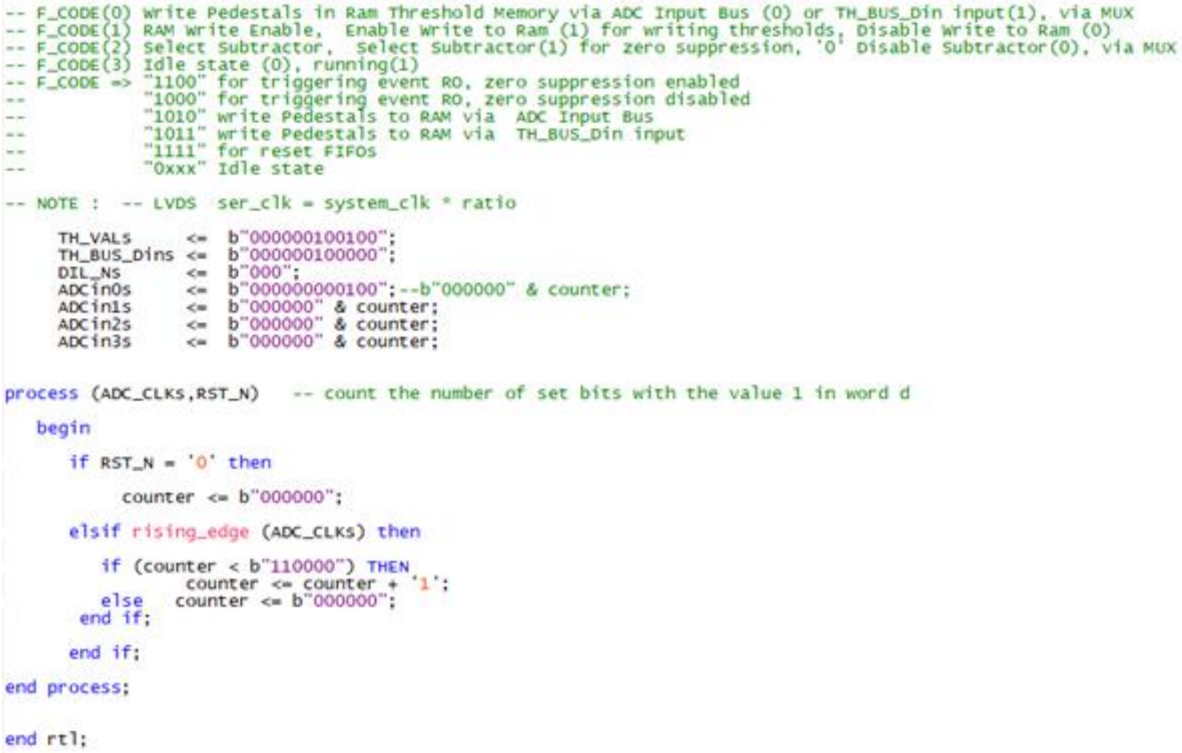
```
begin
      FCODES                 <= FCODE;
      CMP_EN_NS              <= CMP_EN_N;
      RST_NS                 <= not RST_N ;
      ADC_CLK                <= ADC_CLKS;

      --LVDS
      lvds_tx0_D0             <= lvds_tx0_D0s;
      lvds_tx1_D0             <= lvds_tx1_D0s ;
      lvds_tx2_D0             <= lvds_tx2_D0s;

      lvds_tx0_D1             <= lvds_tx0_D1s;
      lvds_tx1_D1             <= lvds_tx1_D1s;
      lvds_tx2_D1             <= lvds_tx2_D1s;

      lvds_tx0_D2             <= lvds_tx0_D2s;
      lvds_tx1_D2             <= lvds_tx1_D2s;
      lvds_tx2_D2             <= lvds_tx2_D2s;

      lvds_tx0_D3             <= lvds_tx0_D3s;
      lvds_tx1_D3             <= lvds_tx1_D3s;
      lvds_tx2_D3             <= lvds_tx2_D3s;


      CINS                    <= CIN  ;
      CRC_NS                  <= CRC_N;
      lvds_tx_clk_D0          <= lvds_tx_clk_D0s ;
      lvds_tx_clk_D1          <= lvds_tx_clk_D1s;
      lvds_tx_clk_D2          <= lvds_tx_clk_D2s;
      lvds_tx_clk_D3          <= lvds_tx_clk_D3s;
      lvds_overflow_D0        <= lvds_overflow_D0s;
      lvds_underflow_D0       <= lvds_underflow_D0s;
      lvds_overflow_D1        <= lvds_overflow_D1s;
      lvds_underflow_D1       <= lvds_underflow_D1s;
      lvds_overflow_D2        <= lvds_overflow_D2s;
      lvds_underflow_D2       <= lvds_underflow_D2s;
      lvds_overflow_D3        <= lvds_overflow_D3s;
      lvds_underflow_D3       <= lvds_underflow_D3s;
      pulse_ack               <= pulse_acks;
```



```
PLL: PLLCLK port map
(
       refclk    => REF_CLK,          -- refclk.clk
       rst       =>  RST_NS,          -- reset.reset
       outclk_0 => STR_CLKAS,         -- outclk0.clk
       outclk_1 => LVDS_SYS_CLKS,     -- outclk1.clk
       outclk_2 => LVDS_SER_CLKS,     -- outclk2.clk
       outclk_3 => SIMCLKS,
       locked    => open
);
```

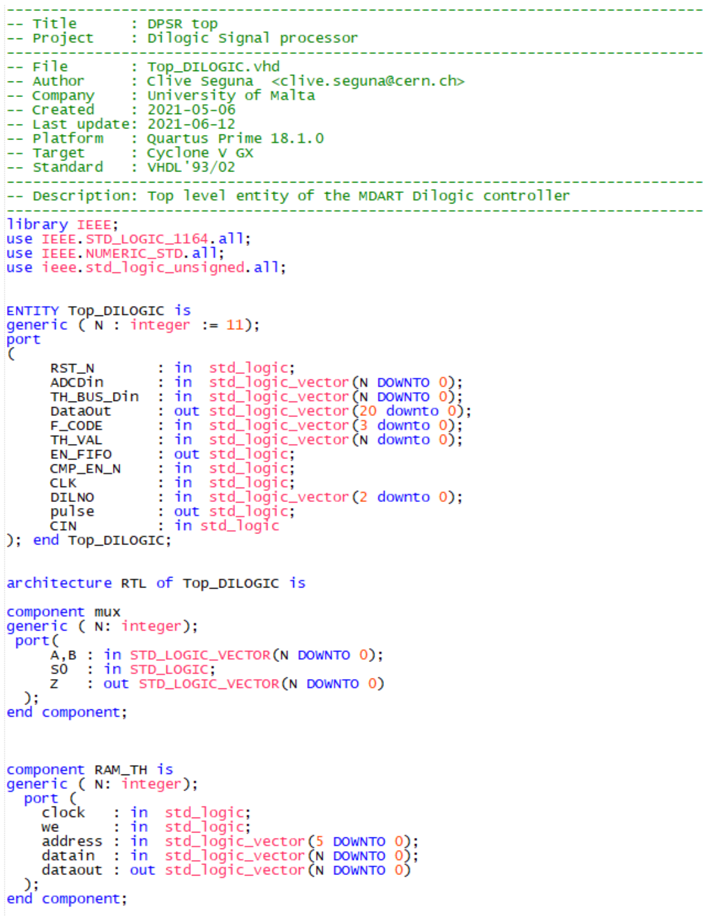


```
ASIC: Top_ASIC port map  (


      ADCin0           =>  ADCin0s,
      ADCin1           =>  ADCin1s,
      ADCin2           =>  ADCin2s,
      ADCin3           =>  ADCin3s,

      TH_BUS_Din       =>  TH_BUS_Dins,
      FCODE            =>  FCODES,
      TH_VAL           =>  TH_VALS,

      CMP_EN_N         =>  CMP_EN_NS,
      RST_N            =>  RST_N,

      LVDS_SYS_CLK     =>  LVDS_SYS_CLKS,
      LVDS_SER_CLK     =>  LVDS_SER_CLKS,
      ADC_CLK          =>  ADC_CLKS,
      STR_CLKA         =>  STR_CLKAS,

      --LVDS
      lvds_tx0_D0      =>  lvds_tx0_D0s,
      lvds_tx1_D0      =>  lvds_tx1_D0s,
      lvds_tx2_D0      =>  lvds_tx2_D0s,

      lvds_tx0_D1      =>  lvds_tx0_D1s,
      lvds_tx1_D1      =>  lvds_tx1_D1s,
      lvds_tx2_D1      =>  lvds_tx2_D1s,

      lvds_tx0_D2      =>  lvds_tx0_D2s,
      lvds_tx1_D2      =>  lvds_tx1_D2s,
      lvds_tx2_D2      =>  lvds_tx2_D2s,

      lvds_tx0_D3      =>  lvds_tx0_D3s,
      lvds_tx1_D3      =>  lvds_tx1_D3s,
      lvds_tx2_D3      =>  lvds_tx2_D3s,

      DIL_N            =>  DIL_NS,
      CIN              =>  CINS,
      CRC_N            =>  CRC_NS,
      lvds_tx_clk_D0 =>  lvds_tx_clk_D0s,
      lvds_tx_clk_D1 =>  lvds_tx_clk_D1s,
      lvds_tx_clk_D2 =>  lvds_tx_clk_D2s,
      lvds_tx_clk_D3 =>  lvds_tx_clk_D3s,

      lvds_overflow_D0   => lvds_overflow_D0s,
      lvds_underflow_D0 => lvds_underflow_D0s,
      lvds_overflow_D1   => lvds_overflow_D1s,
      lvds_underflow_D1 => lvds_underflow_D1s,
      lvds_overflow_D2   => lvds_overflow_D2s,
      lvds_underflow_D2 => lvds_underflow_D2s,
      lvds_overflow_D3   => lvds_overflow_D3s,
      lvds_underflow_D3 => lvds_underflow_D3s,
      pulse_ack          => pulse_acks

);
```

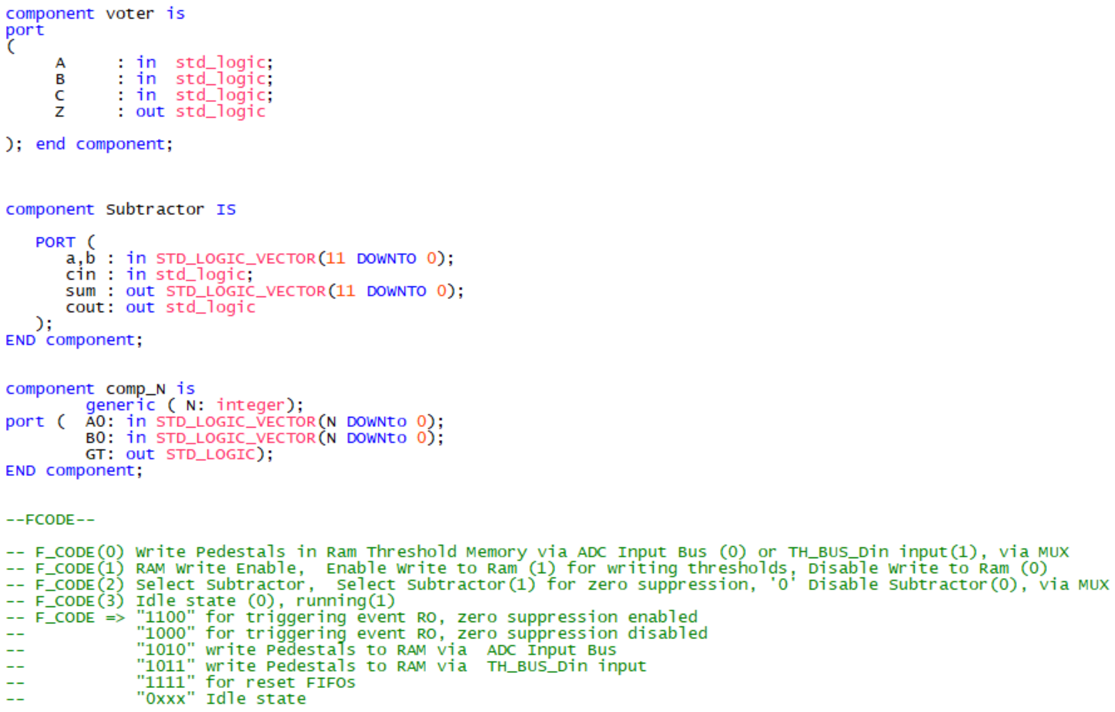


```
-- F_CODE(0) Write Pedestals in Ram Threshold Memory via ADC Input Bus (0) or TH_BUS_Din input(1), via MUX
-- F_CODE(1) RAM Write Enable,  Enable Write to Ram (1) for writing thresholds, Disable Write to Ram (0)
-- F_CODE(2) Select Subtractor,  Select Subtractor(1) for zero suppression, '0' Disable Subtractor(0), via MUX
-- F_CODE(3) Idle state (0), running(1)
-- F_CODE => "1100" for triggering event RO, zero suppression enabled
--           "1000" for triggering event RO, zero suppression disabled
--           "1010" write Pedestals to RAM via  ADC Input Bus
--           "1011" write Pedestals to RAM via  TH_BUS_Din input
--           "1111" for reset FIFOs
--           "0xxx" Idle state

-- NOTE :  -- LVDS  ser_clk = system_clk * ratio

     TH_VALS      <=  b"000000100100";
     TH_BUS_Dins  <=  b"000000100000";
     DIL_Ns       <=  b"000";
     ADCin0s      <=  b"000000000100";--b"000000" & counter;
     ADCin1s      <=  b"000000" & counter;
     ADCin2s      <=  b"000000" & counter;
     ADCin3s      <=  b"000000" & counter;


process (ADC_CLKS,RST_N)   -- count the number of set bits with the value 1 in word d

   begin

      if RST_N = '0' then

           counter <= b"000000";

      elsif rising_edge (ADC_CLKS) then

         if (counter < b"110000") THEN
                counter <= counter + '1';
         else   counter <= b"000000";
       end if;

      end if;

end process;


end rtl;
```

Appendix B.2 – VHDL code for MDART ASIC – Top_DILOGIC.vhd



```
-------------------------------------------------------------------------------
-- Title      : DPSR top
-- Project    : Dilogic Signal processor
-------------------------------------------------------------------------------
-- File       : Top_DILOGIC.vhd
-- Author     : Clive Seguna  <clive.seguna@cern.ch>
-- Company    : University of Malta
-- Created    : 2021-05-06
-- Last update: 2021-06-12
-- Platform   : Quartus Prime 18.1.0
-- Target     : Cyclone V GX
-- Standard   : VHDL'93/02
-------------------------------------------------------------------------------
-- Description: Top level entity of the MDART Dilogic controller
-------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.NUMERIC_STD.all;
use ieee.std_logic_unsigned.all;


ENTITY Top_DILOGIC is
generic ( N : integer := 11);
port
(
     RST_N       : in  std_logic;
     ADCDin      : in  std_logic_vector(N DOWNTO 0);
     TH_BUS_Din  : in  std_logic_vector(N DOWNTO 0);
     DataOut     : out std_logic_vector(20 downto 0);
     F_CODE      : in  std_logic_vector(3 downto 0);
     TH_VAL      : in  std_logic_vector(N downto 0);
     EN_FIFO     : out std_logic;
     CMP_EN_N    : in  std_logic;
     CLK         : in  std_logic;
     DILNO       : in  std_logic_vector(2 downto 0);
     pulse       : out std_logic;
     CIN         : in std_logic
);  end Top_DILOGIC;


architecture RTL of Top_DILOGIC is

component mux
generic ( N: integer);
 port(
     A,B  : in STD_LOGIC_VECTOR(N DOWNTO 0);
     s0   : in STD_LOGIC;
     Z    : out STD_LOGIC_VECTOR(N DOWNTO 0)
   );
end component;


component RAM_TH is
generic ( N: integer);
  port (
    clock   : in  std_logic;
    we      : in  std_logic;
    address : in  std_logic_vector(5 DOWNTO 0);
    datain  : in  std_logic_vector(N DOWNTO 0);
    dataout : out std_logic_vector(N DOWNTO 0)
  );
end component;
```

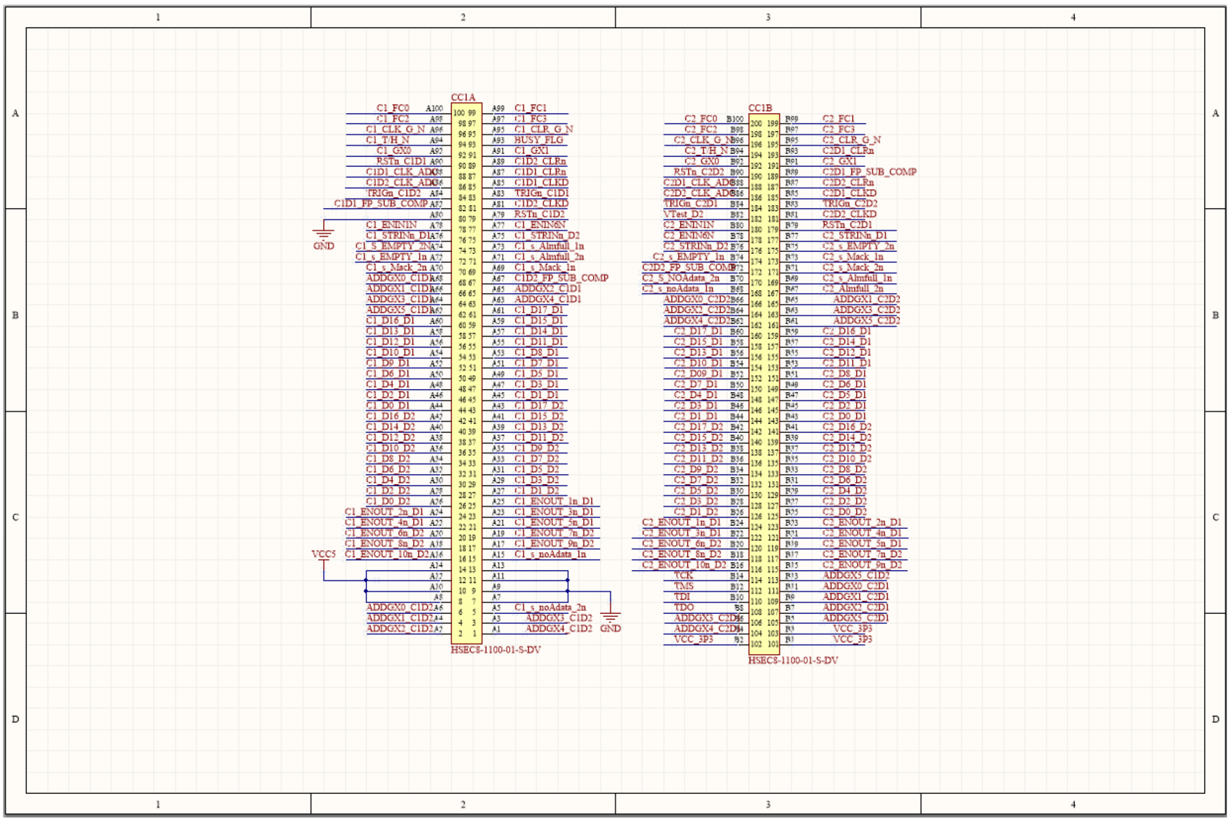


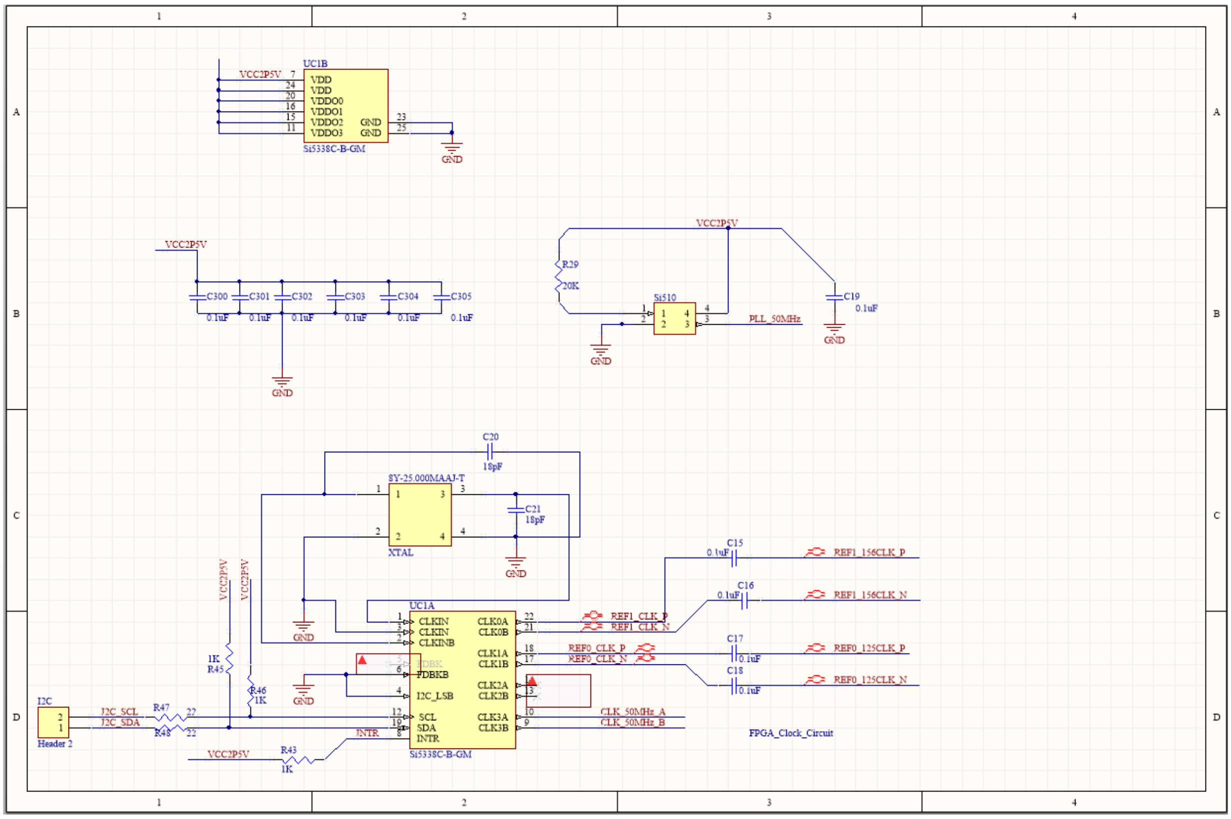
```
component voter  is
port
(
      A      : in  std_logic;
      B      : in  std_logic;
      C      : in  std_logic;
      Z      : out std_logic

);  end component;


component Subtractor IS

   PORT (
      a,b : in STD_LOGIC_VECTOR(11 DOWNTO 0);
      cin : in std_logic;
      sum : out STD_LOGIC_VECTOR(11 DOWNTO 0);
      cout: out std_logic
   );
END component;


component comp_N is
        generic ( N: integer);
port (  A0: in STD_LOGIC_VECTOR(N DOWNto 0);
        B0: in STD_LOGIC_VECTOR(N DOWNto 0);
        GT: out STD_LOGIC);
END component;


--FCODE--

-- F_CODE(0) Write Pedestals in Ram Threshold Memory via ADC Input Bus (0) or TH_BUS_Din input(1), via MUX
-- F_CODE(1) RAM Write Enable,  Enable write to Ram (1) for writing thresholds, Disable write to Ram (0)
-- F_CODE(2) Select Subtractor,  Select Subtractor(1) for zero suppression, '0' Disable Subtractor(0), via MUX
-- F_CODE(3) Idle state (0), running(1)
-- F_CODE => "1100" for triggering event RO, zero suppression enabled
--           "1000" for triggering event RO, zero suppression disabled
--           "1010" write Pedestals to RAM via  ADC Input Bus
--           "1011" write Pedestals to RAM via  TH_BUS_Din input
--           "1111" for reset FIFOs
--           "0xxx" Idle state
```

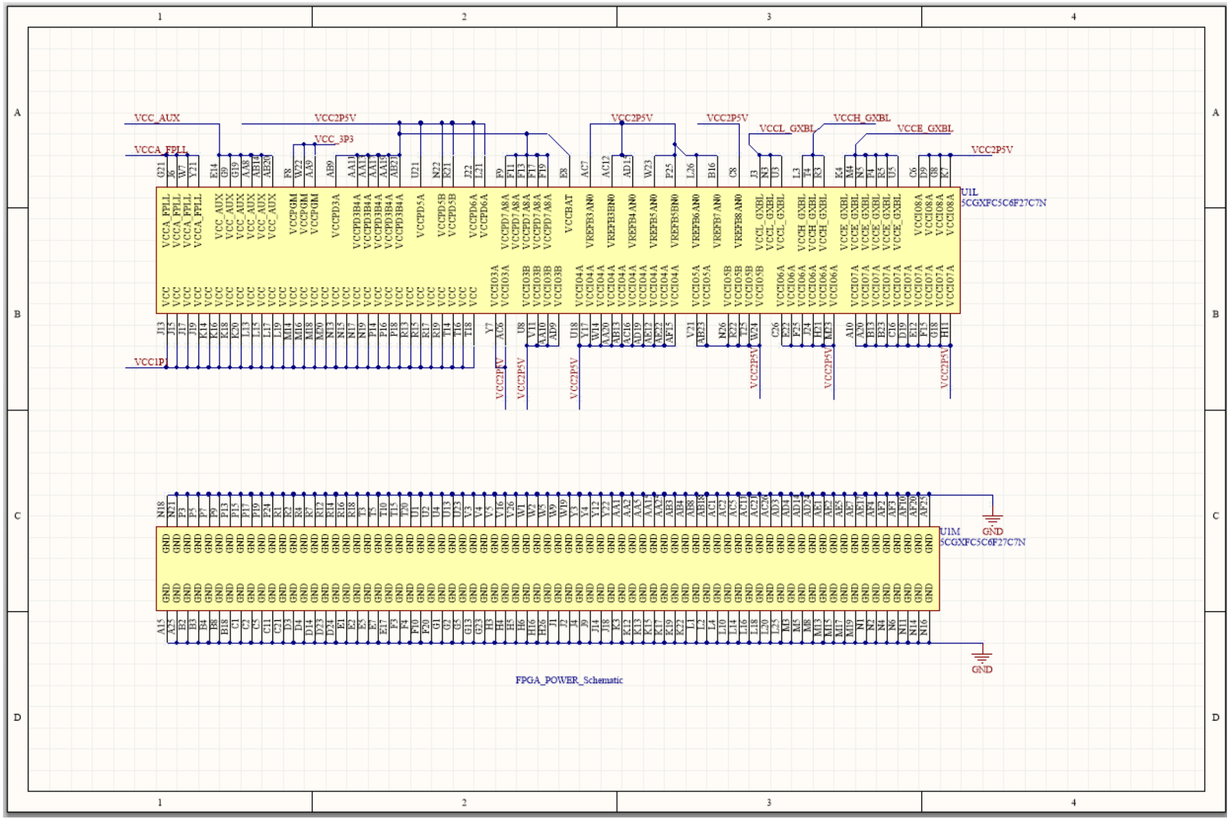
```vhdl
signal MUXOut1_S  : std_logic_vector(11 downto 0);
signal MSEL2_s    : std_logic;
signal EN_CMP_s   : std_logic;
signal en_THWR_s  : std_logic;

signal F_CODEu    : std_logic_vector(3 downto 0);
signal F_CODEs    : std_logic_vector(3 downto 0);
signal RAMOut1_s  : std_logic_vector(11 downto 0);
signal RAMOut2_s  : std_logic_vector(11 downto 0);
signal RAMOut3_s  : std_logic_vector(11 downto 0);

signal CIN_S      : std_logic;
signal COUT1_S    : std_logic;
signal COUT2_S    : std_logic;
signal COUT3_S    : std_logic;

signal SUBOUT1_s  : std_logic_vector(11 downto 0);
signal SUBOUT2_s  : std_logic_vector(11 downto 0);
signal SUBOUT3_s  : std_logic_vector(11 downto 0);

signal FIFO_EN    : std_logic;
signal matchout_s : std_logic;
signal DataOutMUX2: std_logic_vector(11 downto 0);
signal CLK_RAM_S  : std_logic;
signal CH_ADDR    : std_logic_vector(5 downto 0) := b"000000";
signal RamVoter_out : std_logic_vector(11 downto 0);
signal SUBVoter_out : std_logic_vector(11 downto 0);
signal dataout_s    : std_logic_vector(20 downto 0);
signal ADC_CHS_s    : std_logic_vector(1 downto 0);
signal trigger      : std_logic;


begin

      CLK_RAM_S   <=  CLK when (F_CODEs = "1010" or F_CODEs = "1000" or F_CODEs = "1100") ELSE
                      '0';

      trigger     <= ( ( ( (not F_CODEs(1) ) and (FIFO_EN) ) or (  (not F_CODEs(1) ) and (CMP_EN_N) ) ) and F_CODEs(3));
      pulse       <= (trigger);
      EN_FIFO     <= (trigger);
      CIN_S       <=  Cin ;
      F_CODEs     <=  F_CODE ;


MUX1:MUX
 GENERIC MAP (N => 11)
PORT MAP(

     A  =>  ADCDin,
     B  =>  TH_BUS_Din,
     S0 => F_CODEs(0),      --input
     Z  => MUXOut1_S
);



-------------------------------------------------

RAM_PED1 : RAM_TH
generic map( N => 11)
port map(

    clock   => CLK_RAM_S,
    we      => F_CODEs(1), --input
    address => CH_ADDR,
    datain  => MUXOut1_S,
    dataout => RAMOut1_s
);
```

```
SUB1: Subtractor
PORT MAP(

   a      => ADCDin,
   b      => RamVoter_out,
   cin    => CIN_S,
   sum    => SUBOUT1_s,
   cout   => COUT1_S
);


-----------------------------------------------------

RAM_PED2 : RAM_TH
generic map( N => 11)
port map(

    clock   => CLK_RAM_S,
    we      => F_CODES(1), --input
    address => CH_ADDR,
    datain  => MUXOut1_S,
    dataout => RAMOut2_s
);


SUB2: Subtractor
PORT MAP(

   a      => ADCDin,
   b      => RamVoter_out,
   cin    => CIN_S,
   sum    => SUBOUT2_s,
   cout   => COUT2_S
);


--------------------------------------------------------

RAM_PED3 : RAM_TH
generic map( N => 11)
port map(

    clock   => CLK_RAM_S,
    we      => F_CODES(1), --input
    address => CH_ADDR,
    datain  => MUXOut1_S,
    dataout => RAMOut3_s
);


SUB3: Subtractor
PORT MAP(

   a      => ADCDin,
   b      => RamVoter_out,
   cin    => CIN_S,
   sum    => SUBOUT3_s,
   cout   => COUT3_S
);


----------------------RAM & SUBTRACTOR VOTERS ----------------------------

VOT: for k in 0 to (11) generate

        VRAM: Voter port map(RAMOut1_s(k) , RAMOut2_s(k), RAMOut3_s(k), RamVoter_out(k) );
        VSUB: Voter port map(SUBOUT1_s(k) , SUBOUT2_s(k), SUBOUT3_s(k), SUBVoter_out(k) );

     end generate VOT;


MUX2:MUX
 GENERIC MAP (N => 11)
PORT MAP(

     A  => ADCDin,
     B  => SUBVoter_out, -- subtractor input
     S0 => F_CODES(2),  -- Select input
     Z  => DataOutMUX2
);


COMP: comp_N      ---   for Zero suppression
        generic MAP ( N => 11)
port MAP (
        A0 => SUBVoter_out,
        B0 => TH_VAL,
        GT => FIFO_EN
);


process( CLK, RST_N)

begin

if(RST_N = '0') then

    CH_ADDR   <= "000000";
    DataOut      <= b"0000000000000000000000";
    dataout_s    <= b"0000000000000000000000";


elsif rising_edge( CLK) then


                     if ( (CH_ADDR < b"110000") and ( (F_CODES = b"1010" ) or (F_CODES = b"1011" ) ) ) then
                            CH_ADDR      <=   CH_ADDR + "1"; --write ram

                     elsif ( (CH_ADDR < b"110000") and ( (F_CODES = b"1000") or (F_CODES = b"1100") ) ) then

                            CH_ADDR      <=   CH_ADDR + "1";  --write fifo
                            DataOut      <=  ( DILNO & CH_ADDR  & DataOutMUX2 ) ;
                            dataout_s    <=  ( DILNO & CH_ADDR  & DataOutMUX2 ) ;
                     else

                            CH_ADDR    <= b"000000";
                            DataOut    <= dataout_s;
                     end if;


end if;

end process;
```

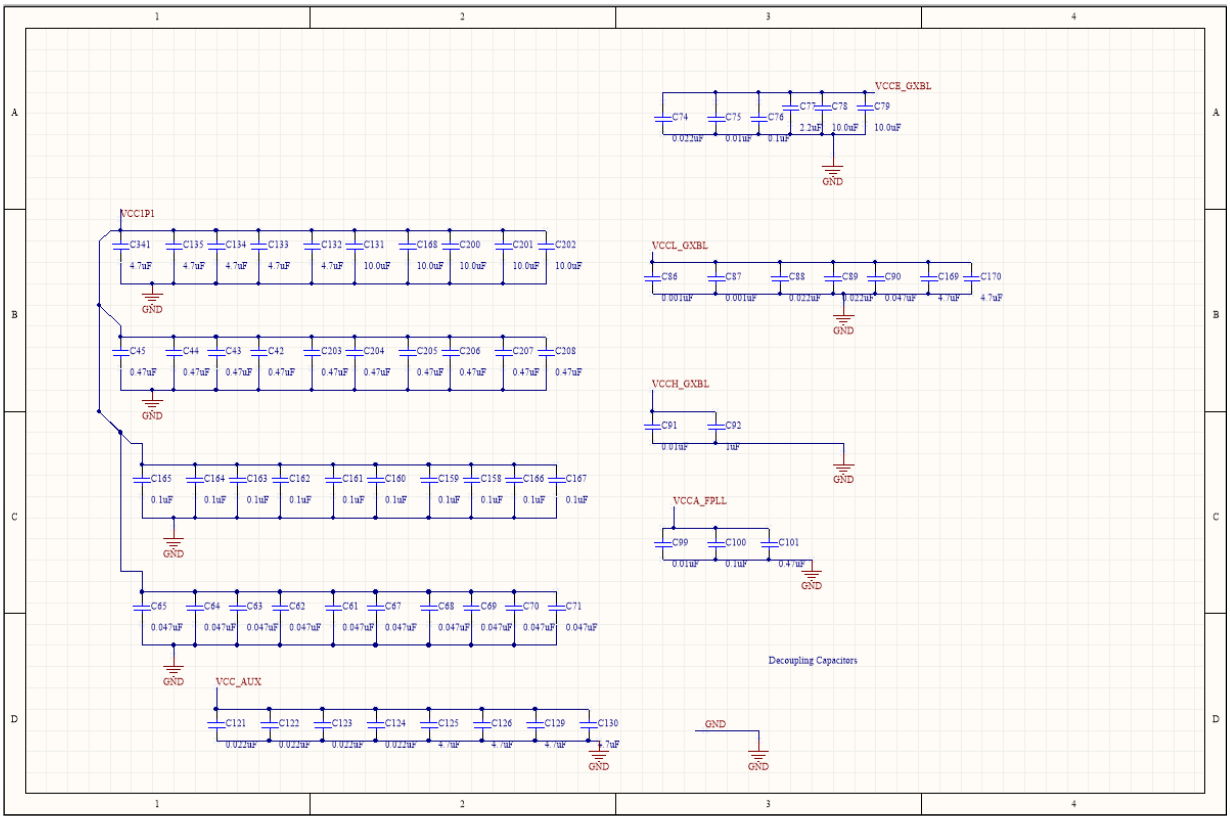
Appendix C – CPV and HMPID Column Controller Card



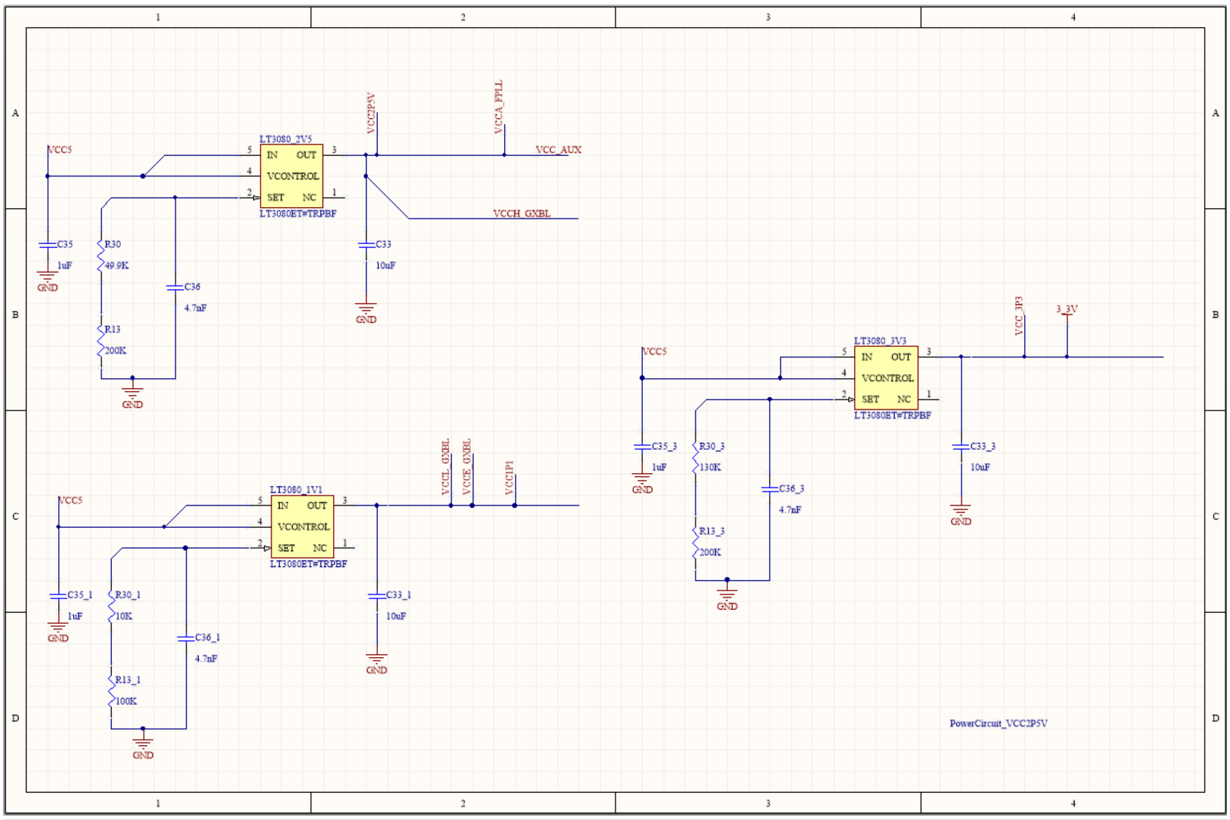


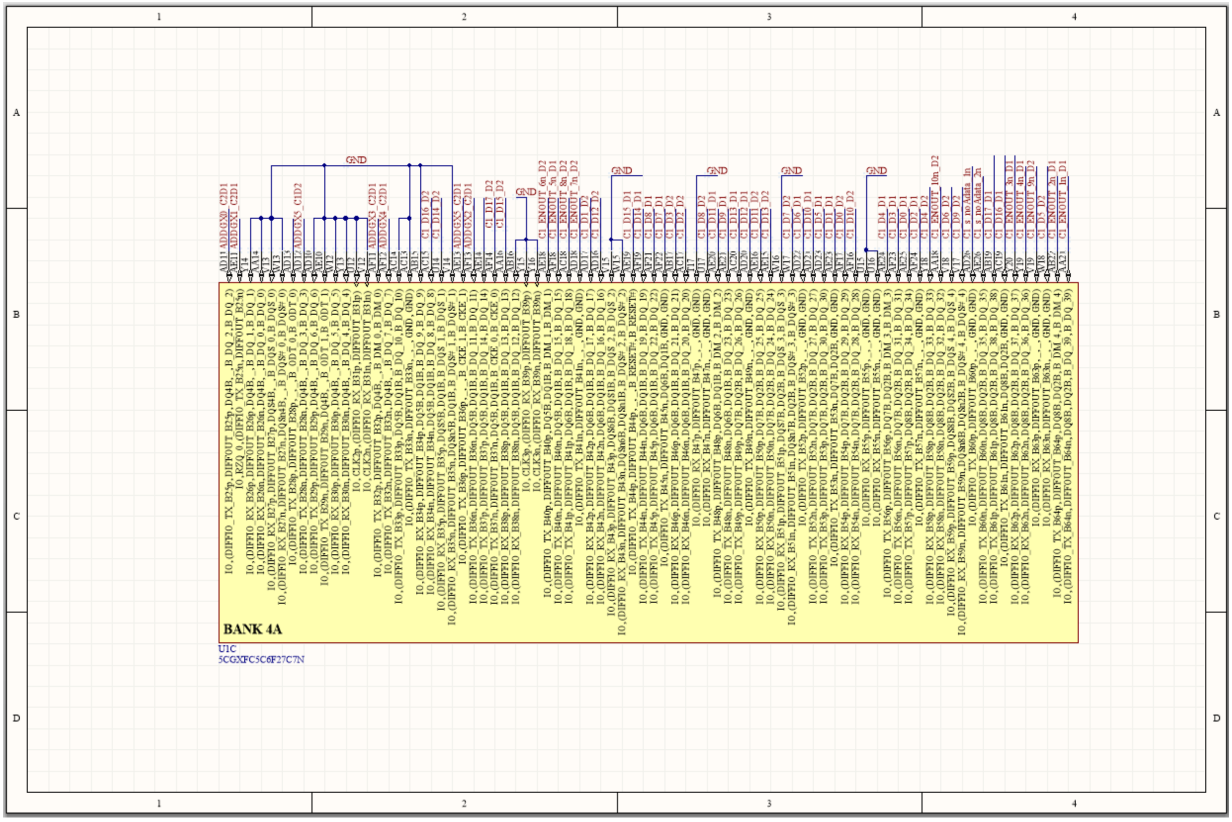


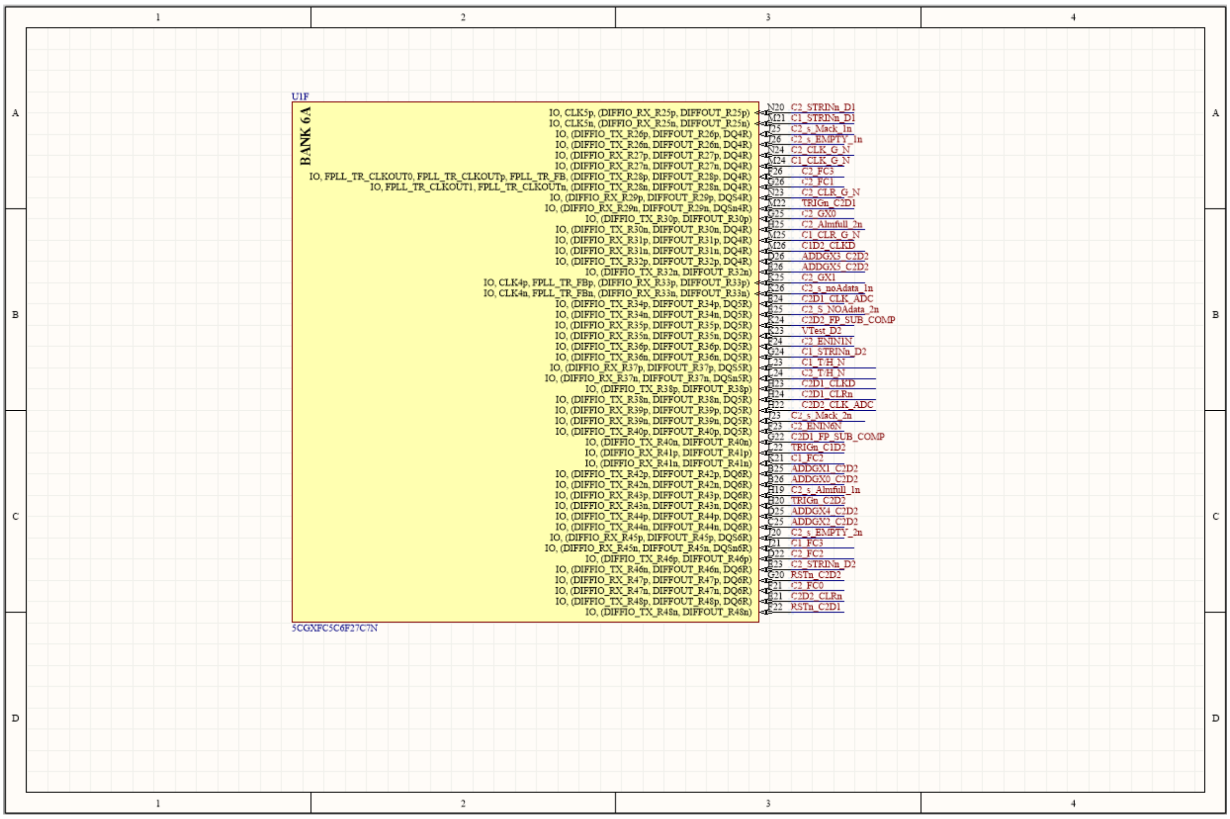
FPGA_POWER_Schematic

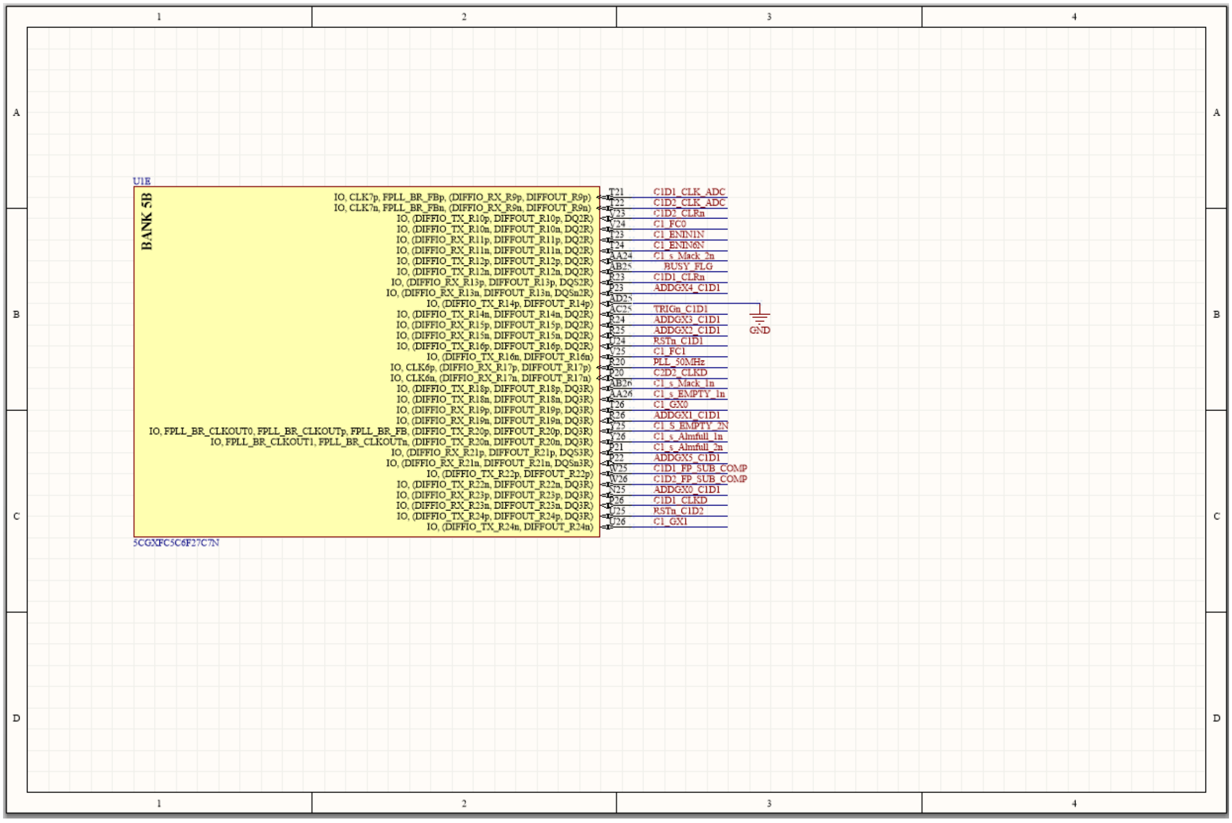


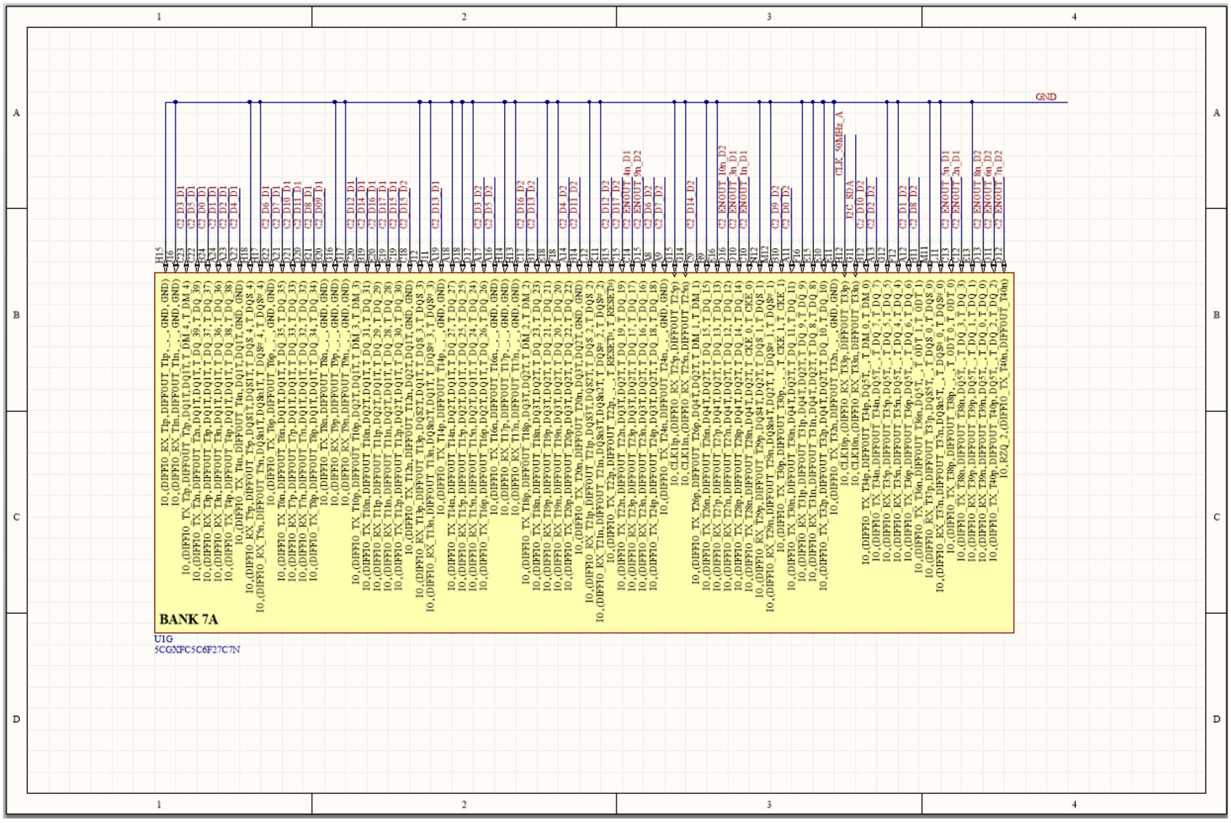
Decoupling Capacitors

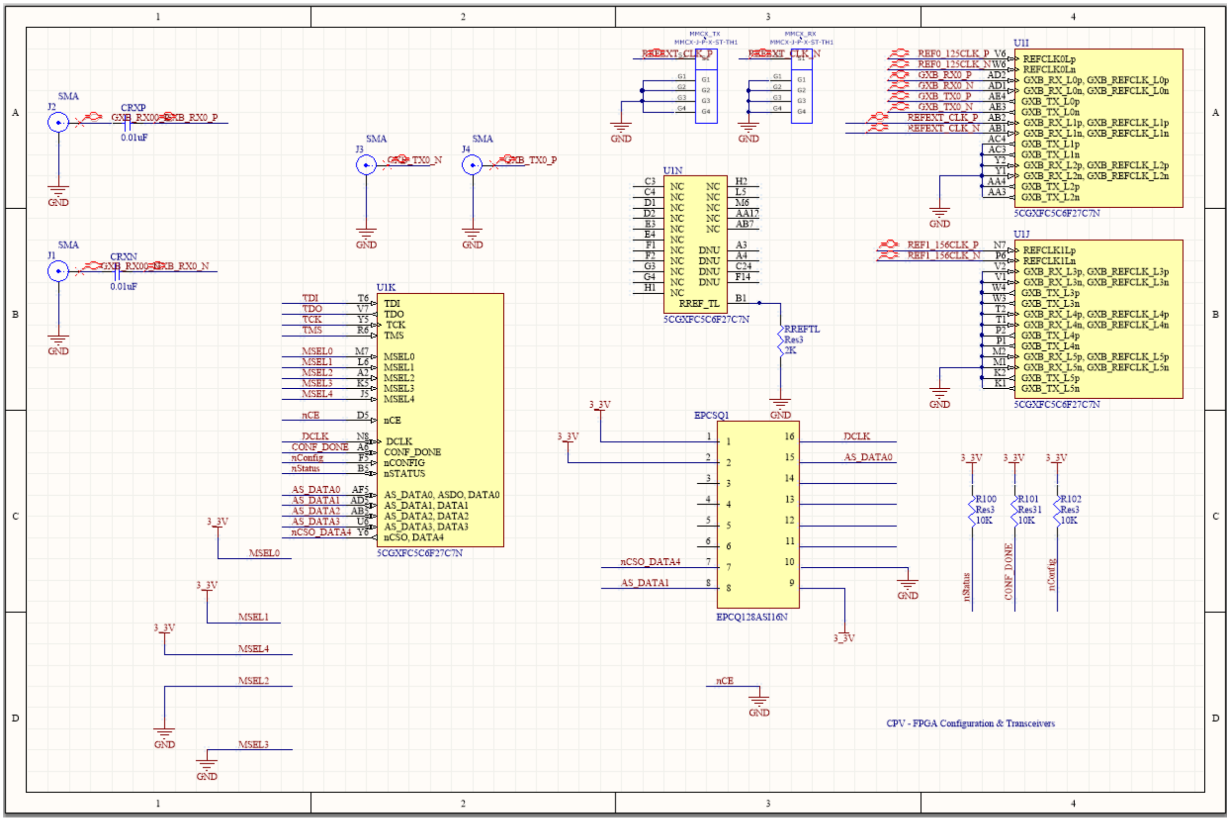

VCC5
LT3080_2V5
IN OUT
VCONTROL
SET NC
LT3080ET#TRPBF
VCC2P5V
VCCA_FPLL
VCC_AUX
VCCH_GXBL
C35
1uF
R30
49.9K
C36
4.7nF
R13
200K
C33
10uF
GND
LT3080_3V3
LT3080ET#TRPBF
VCC_3P3
3_3V
C35_3
1uF
R30_3
130K
C36_3
4.7nF
R13_3
200K
C33_3
10uF
LT3080_1V1
LT3080ET#TRPBF
VCCL_GXBL
VCCE_GXBL
VCCP1
C35_1
1uF
R30_1
10K
C36_1
4.7nF
R13_1
100K
C33_1
10uF
PowerCircuit_VCC2P5V
1 2 3 4
A B C D

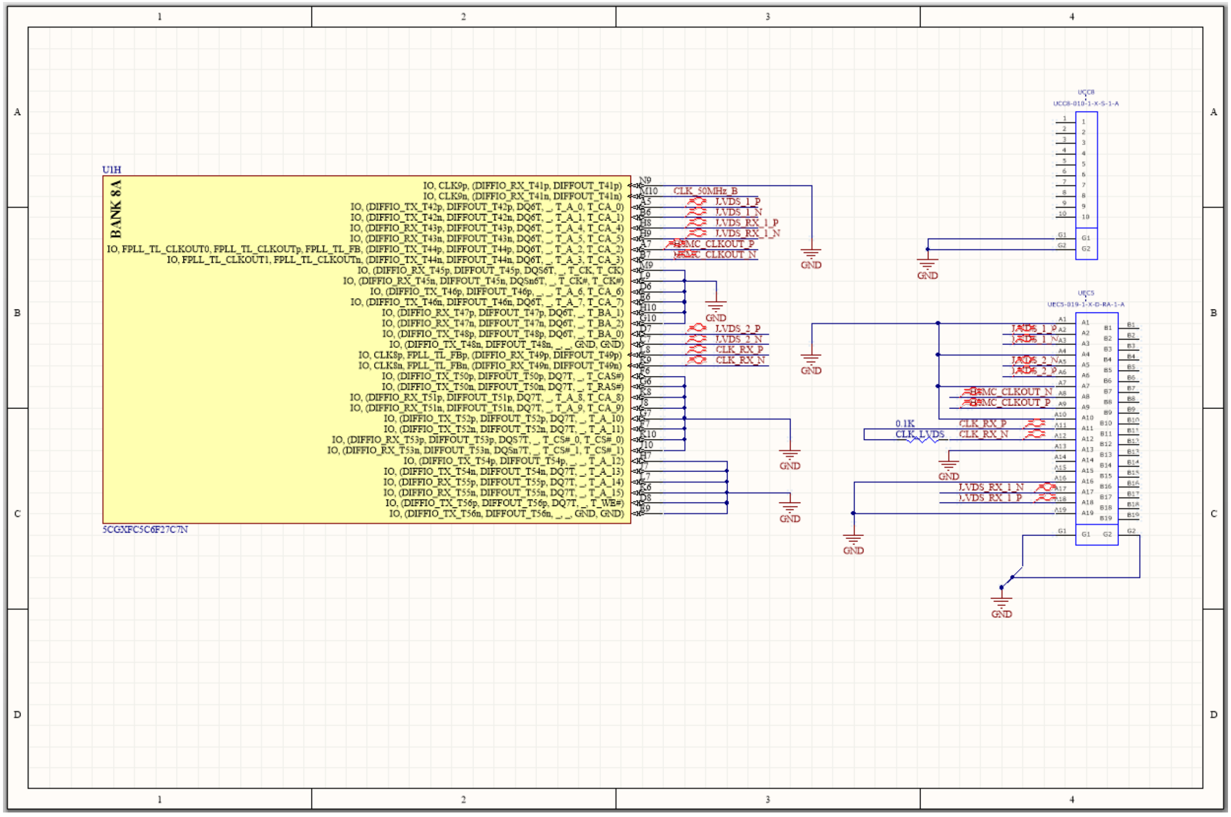

GND
BANK 4A
U1C
5CGXFC5C6F27C7N
1 2 3 4
A B C D

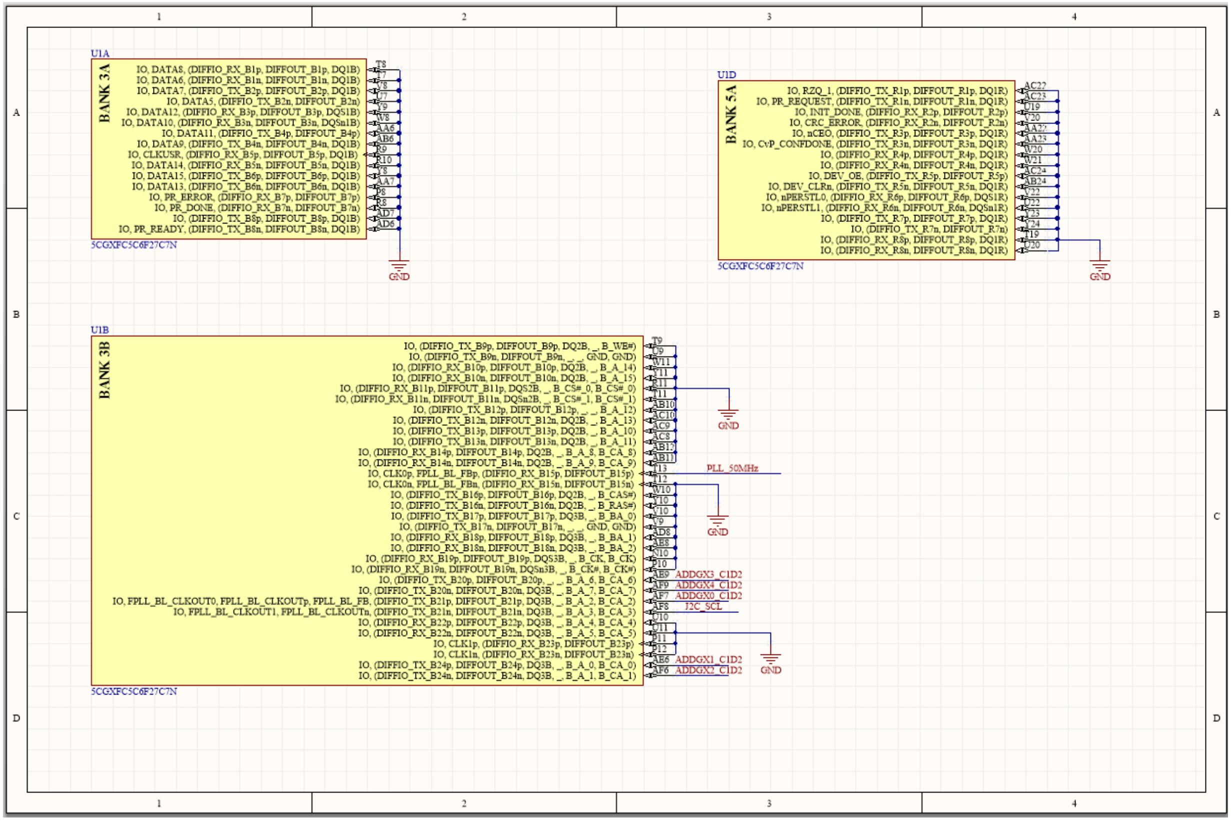

1
2
3
4
A
B
C
D
U1F
BANK 6A
IO, CLK5p, (DIFFIO_RX_R25p, DIFFOUT_R25p) N20 C2_STRINn_D1
IO, CLK5n, (DIFFIO_RX_R25n, DIFFOUT_R25n) M21 C1_STRINn_D1
IO, (DIFFIO_TX_R26p, DIFFOUT_R26p, DQ4R) J25 C2_s_Mack_1n
IO, (DIFFIO_TX_R26n, DIFFOUT_R26n, DQ4R) J26 C2_s_EMPTY_1n
IO, (DIFFIO_RX_R27p, DIFFOUT_R27p, DQ4R) N24 C2_CLK_G_N
IO, (DIFFIO_RX_R27n, DIFFOUT_R27n, DQ4R) M24 C1_CLK_G_N
IO, FPLL_TR_CLKOUT0, FPLL_TR_CLKOUTp, FPLL_TR_FB, (DIFFIO_TX_R28p, DIFFOUT_R28p, DQ4R) F26 C2_FC3
IO, FPLL_TR_CLKOUT1, FPLL_TR_CLKOUTn, (DIFFIO_TX_R28n, DIFFOUT_R28n, DQ4R) G26 C2_FC1
IO, (DIFFIO_RX_R29p, DIFFOUT_R29p, DQS4R) N23 C2_CLR_G_N
IO, (DIFFIO_RX_R29n, DIFFOUT_R29n, DQSn4R) M22 TRIGn_C2D1
IO, (DIFFIO_TX_R30p, DIFFOUT_R30p) G25 C2_GX0
IO, (DIFFIO_TX_R30n, DIFFOUT_R30n, DQ4R) H25 C2_Almfull_2n
IO, (DIFFIO_RX_R31p, DIFFOUT_R31p, DQ4R) M25 C1_CLR_G_N
IO, (DIFFIO_RX_R31n, DIFFOUT_R31n, DQ4R) M26 C1D2_CLKD
IO, (DIFFIO_TX_R32p, DIFFOUT_R32p, DQ4R) D26 ADDGX3_C2D2
IO, (DIFFIO_TX_R32n, DIFFOUT_R32n) E26 ADDGX5_C2D2
IO, CLK4p, FPLL_TR_FBp, (DIFFIO_RX_R33p, DIFFOUT_R33p) K25 C2_GX1
IO, CLK4n, FPLL_TR_FBn, (DIFFIO_RX_R33n, DIFFOUT_R33n) K26 C2_s_noAdata_1n
IO, (DIFFIO_TX_R34p, DIFFOUT_R34p, DQ5R) E24 C2D1_CLK_ADC
IO, (DIFFIO_TX_R34n, DIFFOUT_R34n, DQ5R) E25 C2_S_NOAdata_2n
IO, (DIFFIO_RX_R35p, DIFFOUT_R35p, DQ5R) K24 C2D2_FP_SUB_COMP
IO, (DIFFIO_RX_R35n, DIFFOUT_R35n, DQ5R) K23 VTest_D2
IO, (DIFFIO_TX_R36p, DIFFOUT_R36p, DQ5R) F24 C2_ENIN1N
IO, (DIFFIO_TX_R36n, DIFFOUT_R36n, DQ5R) G24 C1_STRINn_D2
IO, (DIFFIO_RX_R37p, DIFFOUT_R37p, DQS5R) L23 C1_T/H_N
IO, (DIFFIO_RX_R37n, DIFFOUT_R37n, DQSn5R) L24 C2_T/H_N
IO, (DIFFIO_TX_R38p, DIFFOUT_R38p) H23 C2D1_CLKD
IO, (DIFFIO_TX_R38n, DIFFOUT_R38n, DQ5R) H24 C2D1_CLRn
IO, (DIFFIO_RX_R39p, DIFFOUT_R39p, DQ5R) H22 C2D2_CLK_ADC
IO, (DIFFIO_RX_R39n, DIFFOUT_R39n, DQ5R) J23 C2_s_Mack_2n
IO, (DIFFIO_TX_R40p, DIFFOUT_R40p, DQ5R) F23 C2_ENIN6N
IO, (DIFFIO_TX_R40n, DIFFOUT_R40n) G22 C2D1_FP_SUB_COMP
IO, (DIFFIO_RX_R41p, DIFFOUT_R41p) L22 TRIGn_C1D2
IO, (DIFFIO_RX_R41n, DIFFOUT_R41n) K21 C1_FC2
IO, (DIFFIO_TX_R42p, DIFFOUT_R42p, DQ6R) B25 ADDGX1_C2D2
IO, (DIFFIO_TX_R42n, DIFFOUT_R42n, DQ6R) B26 ADDGX0_C2D2
IO, (DIFFIO_RX_R43p, DIFFOUT_R43p, DQ6R) H19 C2_s_Almfull_1n
IO, (DIFFIO_RX_R43n, DIFFOUT_R43n, DQ6R) H20 TRIGn_C2D2
IO, (DIFFIO_TX_R44p, DIFFOUT_R44p, DQ6R) D25 ADDGX4_C2D2
IO, (DIFFIO_TX_R44n, DIFFOUT_R44n, DQ6R) C25 ADDGX2_C2D2
IO, (DIFFIO_RX_R45p, DIFFOUT_R45p, DQS6R) J20 C2_s_EMPTY_2n
IO, (DIFFIO_RX_R45n, DIFFOUT_R45n, DQSn6R) J21 C1_FC3
IO, (DIFFIO_TX_R46p, DIFFOUT_R46p) D22 C2_FC2
IO, (DIFFIO_TX_R46n, DIFFOUT_R46n, DQ6R) E23 C2_STRINn_D2
IO, (DIFFIO_RX_R47p, DIFFOUT_R47p, DQ6R) G20 RSTn_C2D2
IO, (DIFFIO_RX_R47n, DIFFOUT_R47n, DQ6R) F21 C2_FC0
IO, (DIFFIO_TX_R48p, DIFFOUT_R48p, DQ6R) E21 C2D2_CLRn
IO, (DIFFIO_TX_R48n, DIFFOUT_R48n) F22 RSTn_C2D1
5CGXFC5C6F27C7N

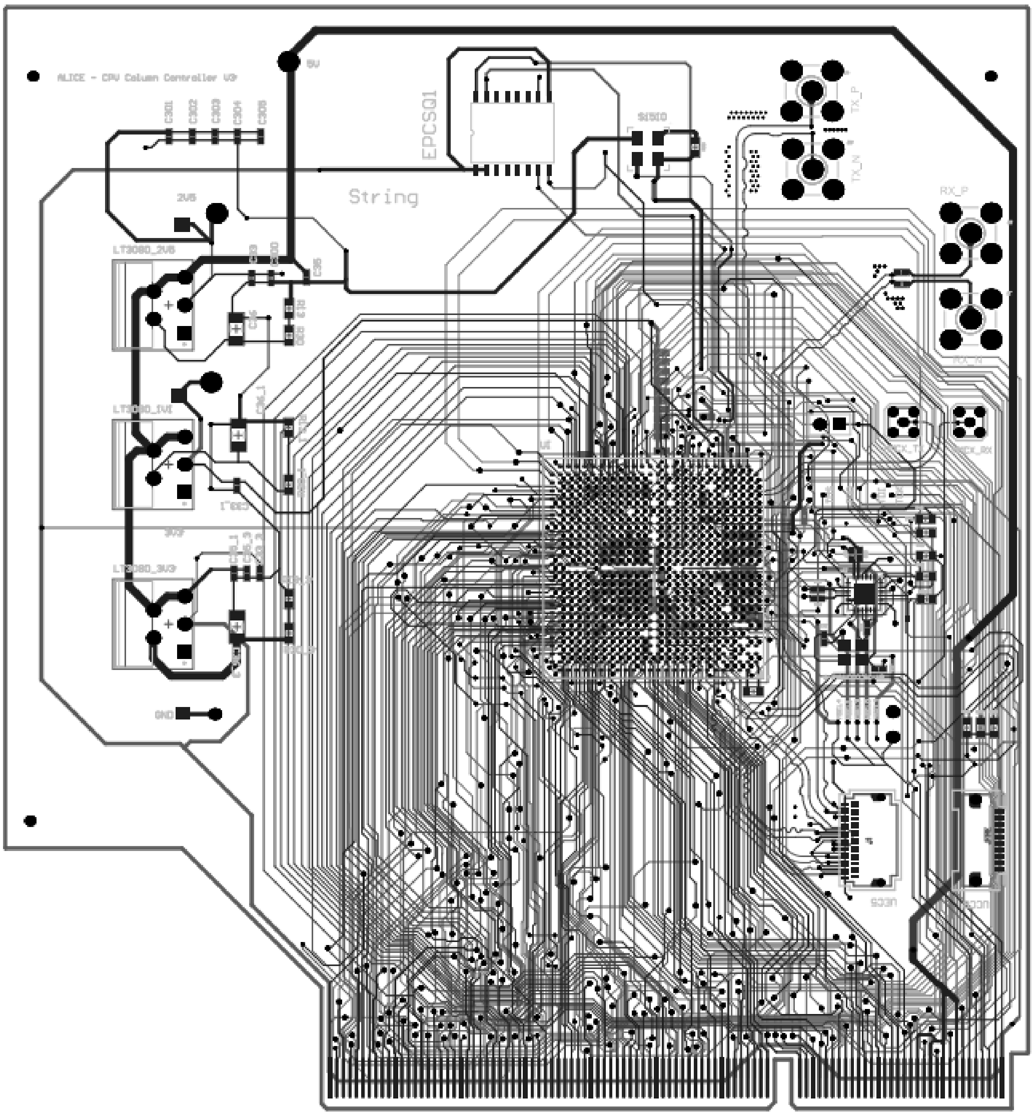

1
2
3
4
A
B
C
D
U1E
BANK 5B
IO, CLK7p, FPLL_BR_FBp, (DIFFIO_RX_R9p, DIFFOUT_R9p) T21 C1D1_CLK_ADC
IO, CLK7n, FPLL_BR_FBn, (DIFFIO_RX_R9n, DIFFOUT_R9n) T22 C1D2_CLK_ADC
IO, (DIFFIO_TX_R10p, DIFFOUT_R10p, DQ2R) V23 C1D2_CLRn
IO, (DIFFIO_TX_R10n, DIFFOUT_R10n, DQ2R) V24 C1_FC0
IO, (DIFFIO_RX_R11p, DIFFOUT_R11p, DQ2R) T23 C1_ENIN1N
IO, (DIFFIO_RX_R11n, DIFFOUT_R11n, DQ2R) T24 C1_ENIN6N
IO, (DIFFIO_TX_R12p, DIFFOUT_R12p, DQ2R) AA24 C1_s_Mack_2n
IO, (DIFFIO_TX_R12n, DIFFOUT_R12n, DQ2R) AB25 BUSY_FLG
IO, (DIFFIO_RX_R13p, DIFFOUT_R13p, DQS2R) R23 C1D1_CLRn
IO, (DIFFIO_RX_R13n, DIFFOUT_R13n, DQSn2R) P23 ADDGX4_C1D1
IO, (DIFFIO_TX_R14p, DIFFOUT_R14p) AD25
IO, (DIFFIO_TX_R14n, DIFFOUT_R14n, DQ2R) AC25 TRIGn_C1D1
IO, (DIFFIO_RX_R15p, DIFFOUT_R15p, DQ2R) R24 ADDGX3_C1D1
IO, (DIFFIO_RX_R15n, DIFFOUT_R15n, DQ2R) R25 ADDGX2_C1D1
IO, (DIFFIO_TX_R16p, DIFFOUT_R16p, DQ2R) U24 RSTn_C1D1
IO, (DIFFIO_TX_R16n, DIFFOUT_R16n) V25 C1_FC1
IO, CLK6p, (DIFFIO_RX_R17p, DIFFOUT_R17p) R20 PLL_50MHz
IO, CLK6n, (DIFFIO_RX_R17n, DIFFOUT_R17n) P20 C2D2_CLKD
IO, (DIFFIO_TX_R18p, DIFFOUT_R18p, DQ3R) AB26 C1_s_Mack_1n
IO, (DIFFIO_TX_R18n, DIFFOUT_R18n, DQ3R) AA26 C1_s_EMPTY_1n
IO, (DIFFIO_RX_R19p, DIFFOUT_R19p, DQ3R) T26 C1_GX0
IO, (DIFFIO_RX_R19n, DIFFOUT_R19n, DQ3R) R26 ADDGX1_C1D1
IO, FPLL_BR_CLKOUT0, FPLL_BR_CLKOUTp, FPLL_BR_FB, (DIFFIO_TX_R20p, DIFFOUT_R20p, DQ3R) Y25 C1_S_EMPTY_2N
IO, FPLL_BR_CLKOUT1, FPLL_BR_CLKOUTn, (DIFFIO_TX_R20n, DIFFOUT_R20n, DQ3R) Y26 C1_s_Almfull_1n
IO, (DIFFIO_RX_R21p, DIFFOUT_R21p, DQS3R) P21 C1_s_Almfull_2n
IO, (DIFFIO_RX_R21n, DIFFOUT_R21n, DQSn3R) P22 ADDGX5_C1D1
IO, (DIFFIO_TX_R22p, DIFFOUT_R22p) W25 C1D1_FP_SUB_COMP
IO, (DIFFIO_TX_R22n, DIFFOUT_R22n, DQ3R) W26 C1D2_FP_SUB_COMP
IO, (DIFFIO_RX_R23p, DIFFOUT_R23p, DQ3R) N25 ADDGX0_C1D1
IO, (DIFFIO_RX_R23n, DIFFOUT_R23n, DQ3R) P26 C1D1_CLKD
IO, (DIFFIO_TX_R24p, DIFFOUT_R24p, DQ3R) U25 RSTn_C1D2
IO, (DIFFIO_TX_R24n, DIFFOUT_R24n) U26 C1_GX1
GND
5CGXFC5C6F27C7N


GND
BANK 7A
U1G
5CGXFC5C6F27C7N
CLK_50MHz_A
I2C_SDA

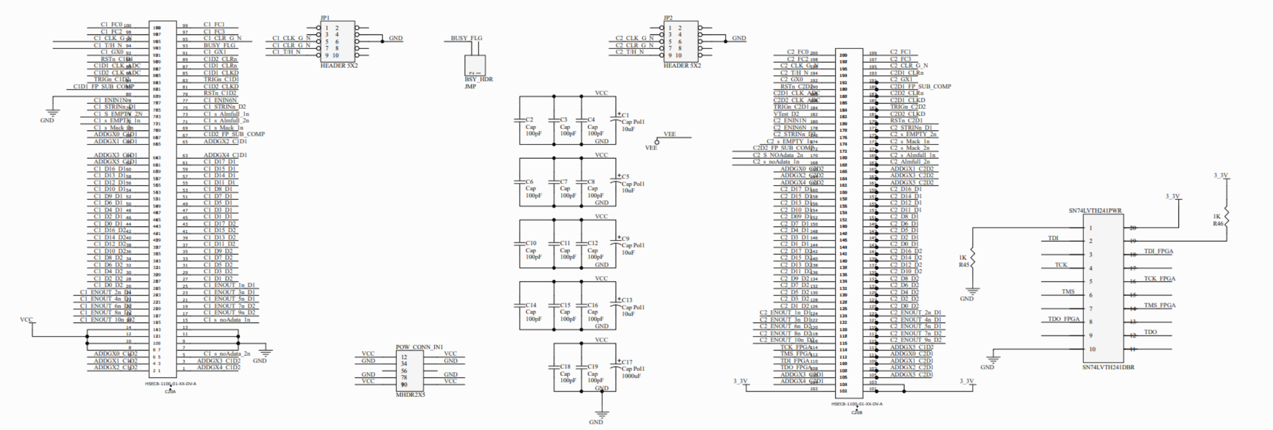

SMA
J2
CRXP
GXB_RX0_P
0.01uF
SMA
J1
CRXN
GXB_RX0_N
0.01uF
SMA
J3
GXB_TX0_N
SMA
J4
GXB_TX0_P
GND
MMCX_TX
MMCX-J-P-X-ST-TH1
MMCX_RX
MMCX-J-P-X-ST-TH1
REFEXT_CLK_P
REFEXT_CLK_N
U1I
REF0_125CLK_P
REF0_125CLK_N
REFCLK0Lp
REFCLK0Ln
GXB_RX_L0p, GXB_REFCLK_L0p
GXB_RX_L0n, GXB_REFCLK_L0n
GXB_TX_L0p
GXB_TX_L0n
GXB_RX_L1p, GXB_REFCLK_L1p
GXB_RX_L1n, GXB_REFCLK_L1n
GXB_TX_L1p
GXB_TX_L1n
GXB_RX_L2p, GXB_REFCLK_L2p
GXB_RX_L2n, GXB_REFCLK_L2n
GXB_TX_L2p
GXB_TX_L2n
5CGXFC5C6F27C7N
U1J
REF1_156CLK_P
REF1_156CLK_N
REFCLK1Lp
REFCLK1Ln
GXB_RX_L3p, GXB_REFCLK_L3p
GXB_RX_L3n, GXB_REFCLK_L3n
GXB_TX_L3p
GXB_TX_L3n
GXB_RX_L4p, GXB_REFCLK_L4p
GXB_RX_L4n, GXB_REFCLK_L4n
GXB_TX_L4p
GXB_TX_L4n
GXB_RX_L5p, GXB_REFCLK_L5p
GXB_RX_L5n, GXB_REFCLK_L5n
GXB_TX_L5p
GXB_TX_L5n
5CGXFC5C6F27C7N
U1N
NC
DNU
RREF_TL
5CGXFC5C6F27C7N
RREFTL
Res3
2K
U1K
TDI
TDO
TCK
TMS
MSEL0
MSEL1
MSEL2
MSEL3
MSEL4
nCE
DCLK
CONF_DONE
nCONFIG
nSTATUS
AS_DATA0, ASDO, DATA0
AS_DATA1, DATA1
AS_DATA2, DATA2
AS_DATA3, DATA3
nCSO, DATA4
5CGXFC5C6F27C7N
3_3V
EPCSQ1
EPCQ128ASI16N
DCLK
AS_DATA0
nCSO_DATA4
AS_DATA1
R100
Res3
10K
R101
Res31
10K
R102
Res3
10K
nStatus
CONF_DONE
nConfig
CPV - FPGA Configuration &amp; Transceivers

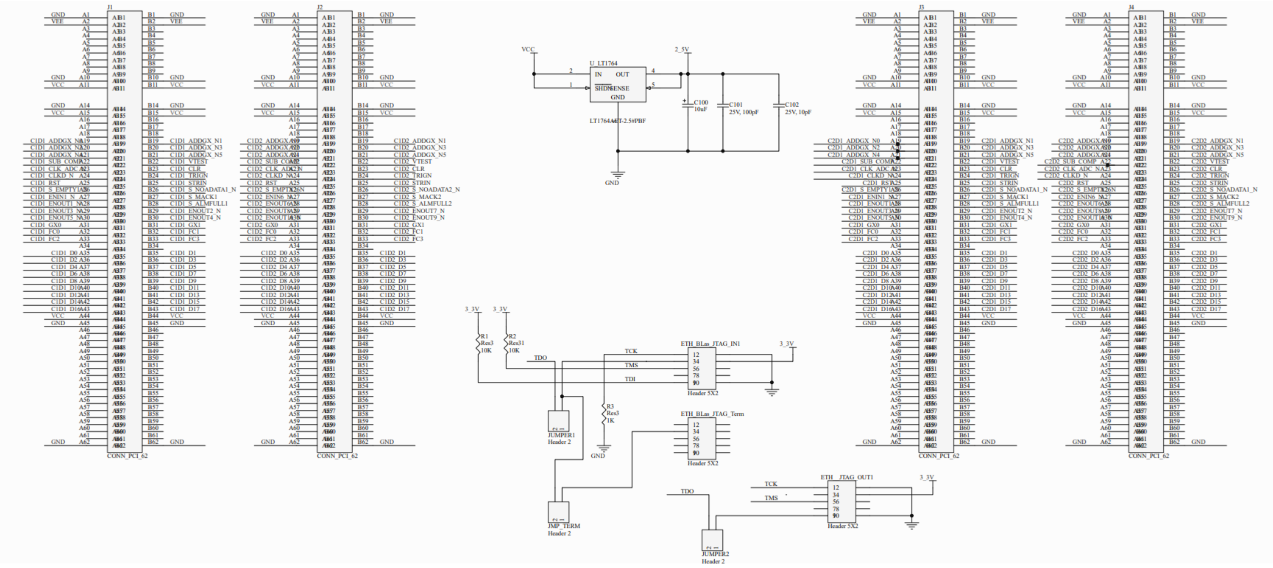

U1H
BANK 8A
5CGXFC5C6F27C7N
CLK_50MHz_B
LVDS_1_P
LVDS_1_N
LVDS_RX_1_P
LVDS_RX_1_N
HSMC_CLKOUT_P
HSMC_CLKOUT_N
LVDS_2_P
LVDS_2_N
CLK_RX_P
CLK_RX_N
GND
UCC8
UCC8-010-1-X-S-1-A
UEC5
UEC5-019-1-X-D-RA-1-A
0.1K
CLK_LVDS
LVDS_RX_1_N
LVDS_RX_1_P

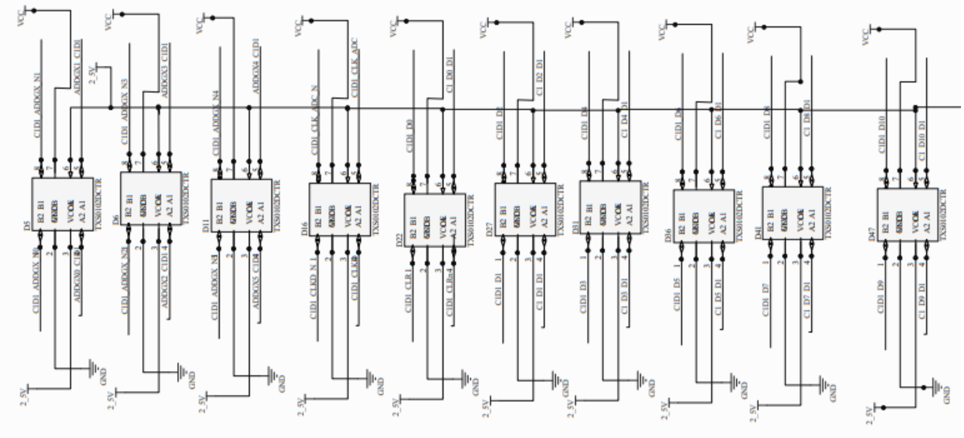

U1A
BANK 3A
5CGXFC5C6F27C7N
U1D
BANK 5A
5CGXFC5C6F27C7N
U1B
BANK 3B
5CGXFC5C6F27C7N
PLL_50MHz
ADDGX3_C1D2
ADDGX4_C1D2
ADDGX0_C1D2
I2C_SCL
ADDGX1_C1D2
ADDGX2_C1D2
GND


ALICE - CPV Column Controller V3
5V
C301
C302
C303
C304
C305
EPCSQ1
String
SI510
TX_P
TX_N
RX_P
RX_N
2V5
LT3080_2V5
LT3080_1V1
3V3
LT3080_3V3
GND
U1

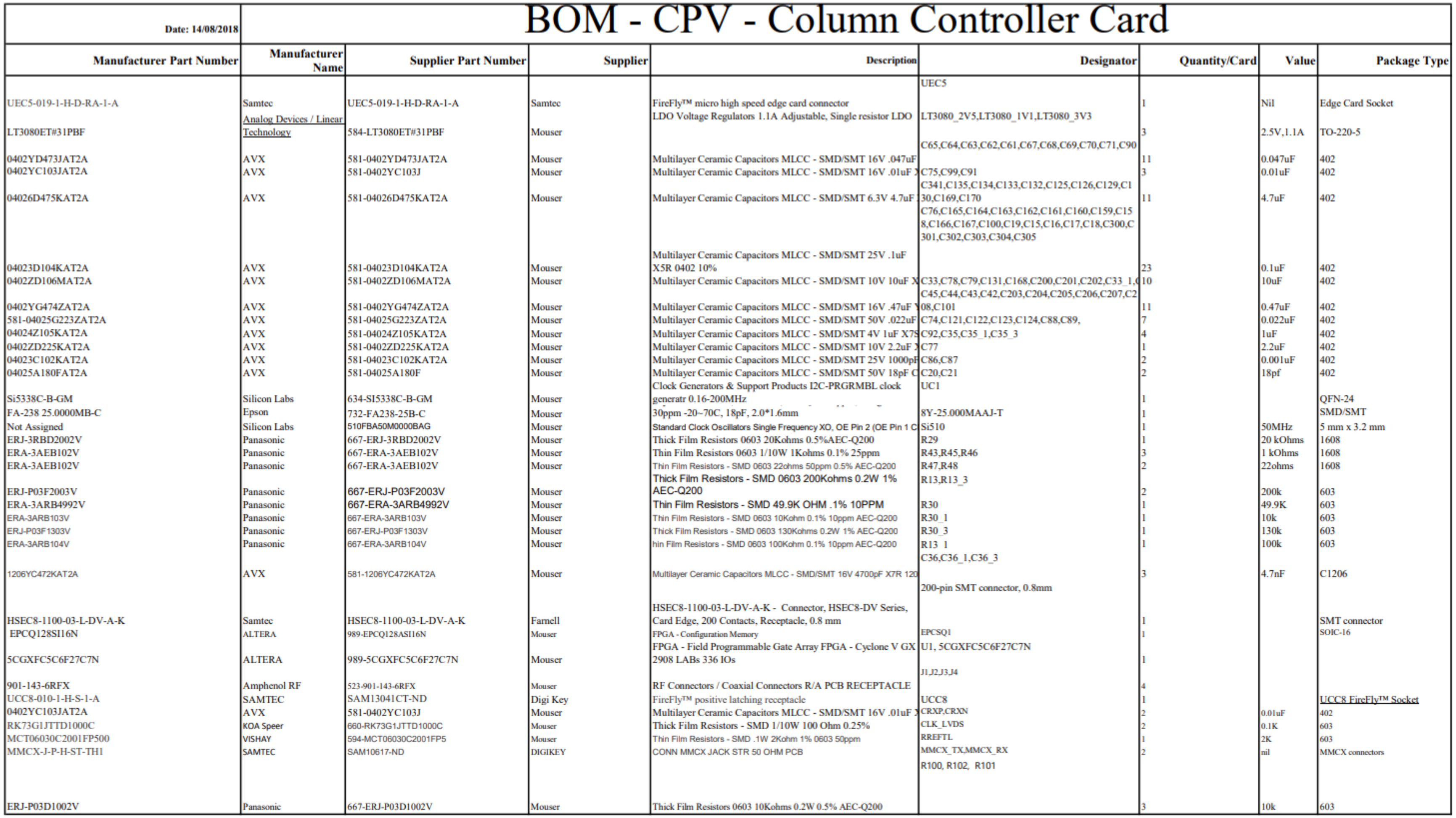
Appendix D – CPV Segment Interface Board (part of)







Appendix E – BOM – CPV Column Controller Card

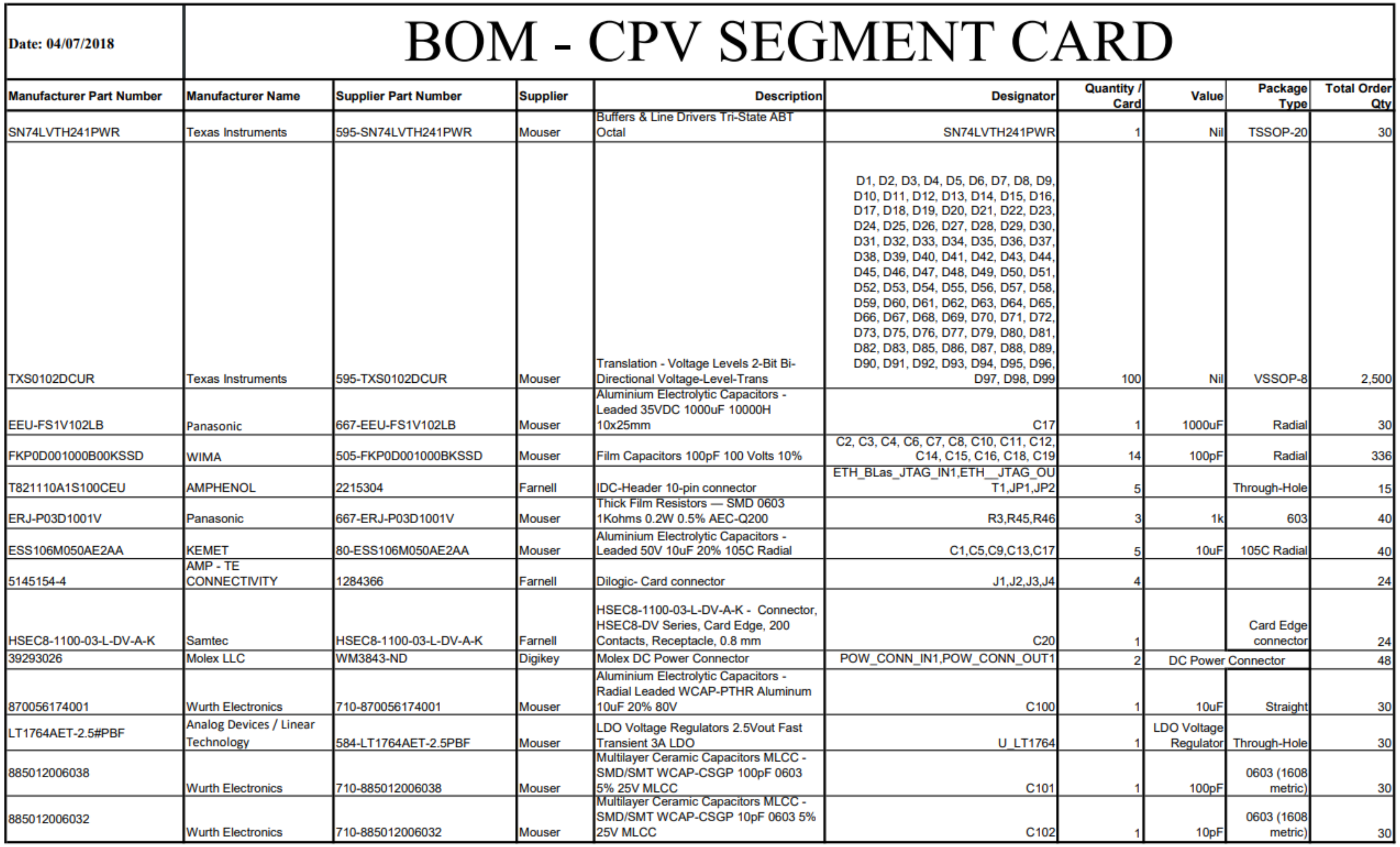


Date: 14/08/2018

BOM - CPV - Column Controller Card

Manufacturer Part Number	Manufacturer Name	Supplier Part Number	Supplier	Description	Designator	Quantity/Card	Value	Package Type
UEC5-019-1-H-D-RA-1-A	Samtec	UEC5-019-1-H-D-RA-1-A	Samtec	FireFly™ micro high speed edge card connector	UEC5	1	Nil	Edge Card Socket
LT3080ET#31PBF	Analog Devices / Linear Technology	584-LT3080ET#31PBF	Mouser	LDO Voltage Regulators 1.1A Adjustable, Single resistor LDO	LT3080_2V5,LT3080_1V1,LT3080_3V3	3	2.5V,1.1A	TO-220-5
0402YD473JAT2A	AVX	581-0402YD473JAT2A	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT 16V .047uF	C65,C64,C63,C62,C61,C67,C68,C69,C70,C71,C90	11	0.047uF	402
0402YC103JAT2A	AVX	581-0402YC103J	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT 16V .01uF X	C75,C99,C91	3	0.01uF	402
04026D475KAT2A	AVX	581-04026D475KAT2A	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT 6.3V 4.7uF	C341,C135,C134,C133,C132,C125,C126,C129,C130,C169,C170	11	4.7uF	402
04023D104KAT2A	AVX	581-04023D104KAT2A	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT 25V .1uF X5R 0402 10%	C76,C165,C164,C163,C162,C161,C160,C159,C158,C166,C167,C100,C19,C15,C16,C17,C18,C300,C301,C302,C303,C304,C305	23	0.1uF	402
0402ZD106MAT2A	AVX	581-0402ZD106MAT2A	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT 10V 10uF X	C33,C78,C79,C131,C168,C200,C201,C202,C33_1,C	10	10uF	402
0402YG474ZAT2A	AVX	581-0402YG474ZAT2A	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT 16V .47uF Y	C45,C44,C43,C42,C203,C204,C205,C206,C207,C208,C101	11	0.47uF	402
581-04025G223ZAT2A	AVX	581-04025G223ZAT2A	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT 50V .022uF	C74,C121,C122,C123,C124,C88,C89,	7	0.022uF	402
04024Z105KAT2A	AVX	581-04024Z105KAT2A	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT 4V 1uF X7S	C92,C35,C35_1,C35_3	4	1uF	402
0402ZD225KAT2A	AVX	581-0402ZD225KAT2A	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT 10V 2.2uF X	C77	1	2.2uF	402
04023C102KAT2A	AVX	581-04023C102KAT2A	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT 25V 1000pF	C86,C87	2	0.001uF	402
04025A180FAT2A	AVX	581-04025A180F	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT 50V 18pF C	C20,C21	2	18pf	402
Si5338C-B-GM	Silicon Labs	634-SI5338C-B-GM	Mouser	Clock Generators & Support Products I2C-PRGRMBL clock generatr 0.16-200MHz	UC1	1		QFN-24
FA-238 25.0000MB-C	Epson	732-FA238-25B-C	Mouser	30ppm -20~70C, 18pF, 2.0*1.6mm	8Y-25.000MAAJ-T	1		SMD/SMT
Not Assigned	Silicon Labs	510FBA50M0000BAG	Mouser	Standard Clock Oscillators Single Frequency XO, OE Pin 2 (OE Pin 1 C	Si510	1	50MHz	5 mm x 3.2 mm
ERJ-3RBD2002V	Panasonic	667-ERJ-3RBD2002V	Mouser	Thick Film Resistors 0603 20Kohms 0.5%AEC-Q200	R29	1	20 kOhms	1608
ERA-3AEB102V	Panasonic	667-ERA-3AEB102V	Mouser	Thin Film Resistors 0603 1/10W 1Kohms 0.1% 25ppm	R43,R45,R46	3	1 kOhms	1608
ERA-3AEB102V	Panasonic	667-ERA-3AEB102V	Mouser	Thin Film Resistors - SMD 0603 22ohms 50ppm 0.5% AEC-Q200	R47,R48	2	22ohms	1608
ERJ-P03F2003V	Panasonic	667-ERJ-P03F2003V	Mouser	Thick Film Resistors - SMD 0603 200Kohms 0.2W 1% AEC-Q200	R13,R13_3	2	200k	603
ERA-3ARB4992V	Panasonic	667-ERA-3ARB4992V	Mouser	Thin Film Resistors - SMD 49.9K OHM .1% 10PPM	R30	1	49.9K	603
ERA-3ARB103V	Panasonic	667-ERA-3ARB103V	Mouser	Thin Film Resistors - SMD 0603 10Kohm 0.1% 10ppm AEC-Q200	R30_1	1	10k	603
ERJ-P03F1303V	Panasonic	667-ERJ-P03F1303V	Mouser	Thick Film Resistors - SMD 0603 130Kohms 0.2W 1% AEC-Q200	R30_3	1	130k	603
ERA-3ARB104V	Panasonic	667-ERA-3ARB104V	Mouser	hin Film Resistors - SMD 0603 100Kohm 0.1% 10ppm AEC-Q200	R13_1	1	100k	603
1206YC472KAT2A	AVX	581-1206YC472KAT2A	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT 16V 4700pF X7R 120	C36,C36_1,C36_3	3	4.7nF	C1206
HSEC8-1100-03-L-DV-A-K	Samtec	HSEC8-1100-03-L-DV-A-K	Farnell	HSEC8-1100-03-L-DV-A-K - Connector, HSEC8-DV Series, Card Edge, 200 Contacts, Receptacle, 0.8 mm	200-pin SMT connector, 0.8mm	1		SMT connector
EPCQ128SI16N	ALTERA	989-EPCQ128ASI16N	Mouser	FPGA - Configuration Memory	EPCSQ1	1		SOIC-16
5CGXFC5C6F27C7N	ALTERA	989-5CGXFC5C6F27C7N	Mouser	FPGA - Field Programmable Gate Array FPGA - Cyclone V GX 2908 LABs 336 IOs	U1, 5CGXFC5C6F27C7N	1		
901-143-6RFX	Amphenol RF	523-901-143-6RFX	Mouser	RF Connectors / Coaxial Connectors R/A PCB RECEPTACLE	J1,J2,J3,J4	4		
UCC8-010-1-H-S-1-A	SAMTEC	SAM13041CT-ND	Digi Key	FireFly™ positive latching receptacle	UCC8	1		UCC8 FireFly™ Socket
0402YC103JAT2A	AVX	581-0402YC103J	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT 16V .01uF X	CRXP,CRXN	2	0.01uF	402
RK73G1JTTD1000C	KOA Speer	660-RK73G1JTTD1000C	Mouser	Thick Film Resistors - SMD 1/10W 100 Ohm 0.25%	CLK_LVDS	2	0.1K	603
MCT06030C2001FP500	VISHAY	594-MCT06030C2001FP5	Mouser	Thin Film Resistors - SMD .1W 2Kohm 1% 0603 50ppm	RREFTL	1	2K	603
MMCX-J-P-H-ST-TH1	SAMTEC	SAM10617-ND	DIGIKEY	CONN MMCX JACK STR 50 OHM PCB	MMCX_TX,MMCX_RX	2	nil	MMCX connectors
ERJ-P03D1002V	Panasonic	667-ERJ-P03D1002V	Mouser	Thick Film Resistors 0603 10Kohms 0.2W 0.5% AEC-Q200	R100, R102, R101	3	10k	603

Appendix F – BOM – CPV Segment Card

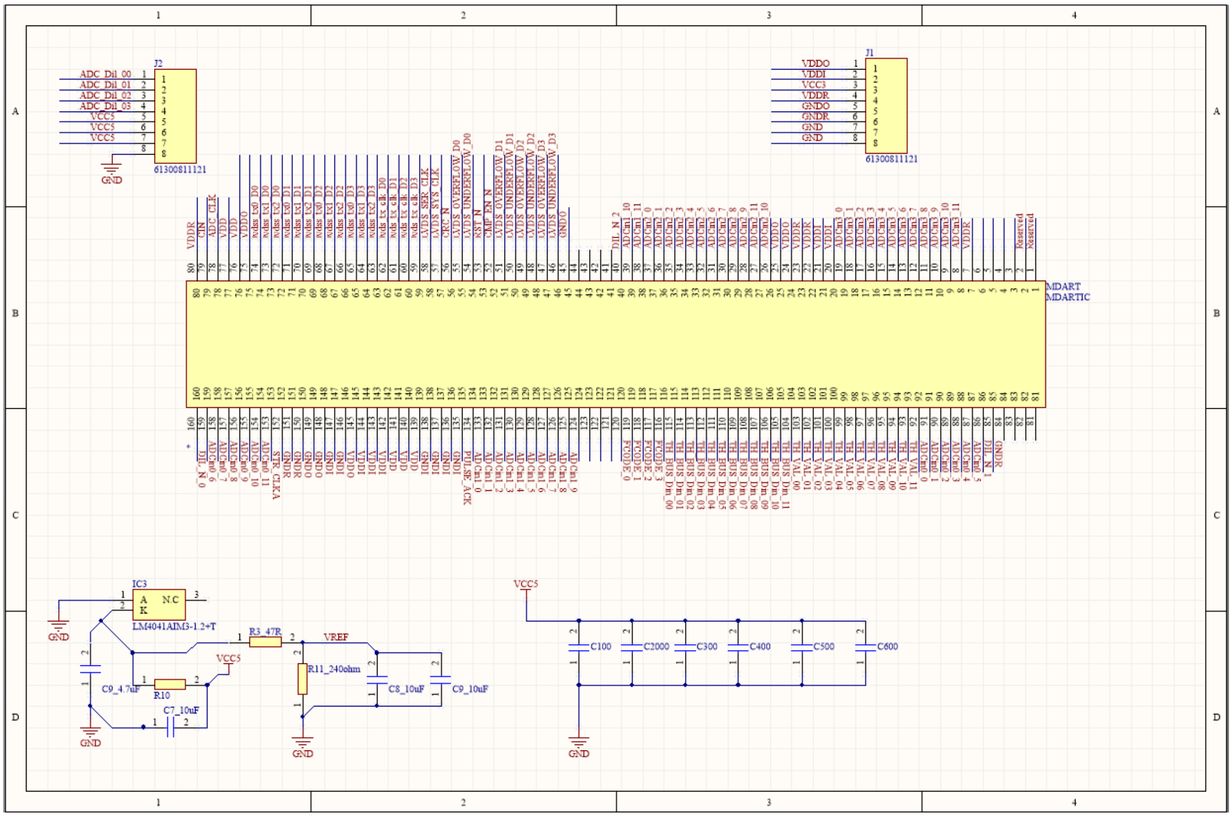


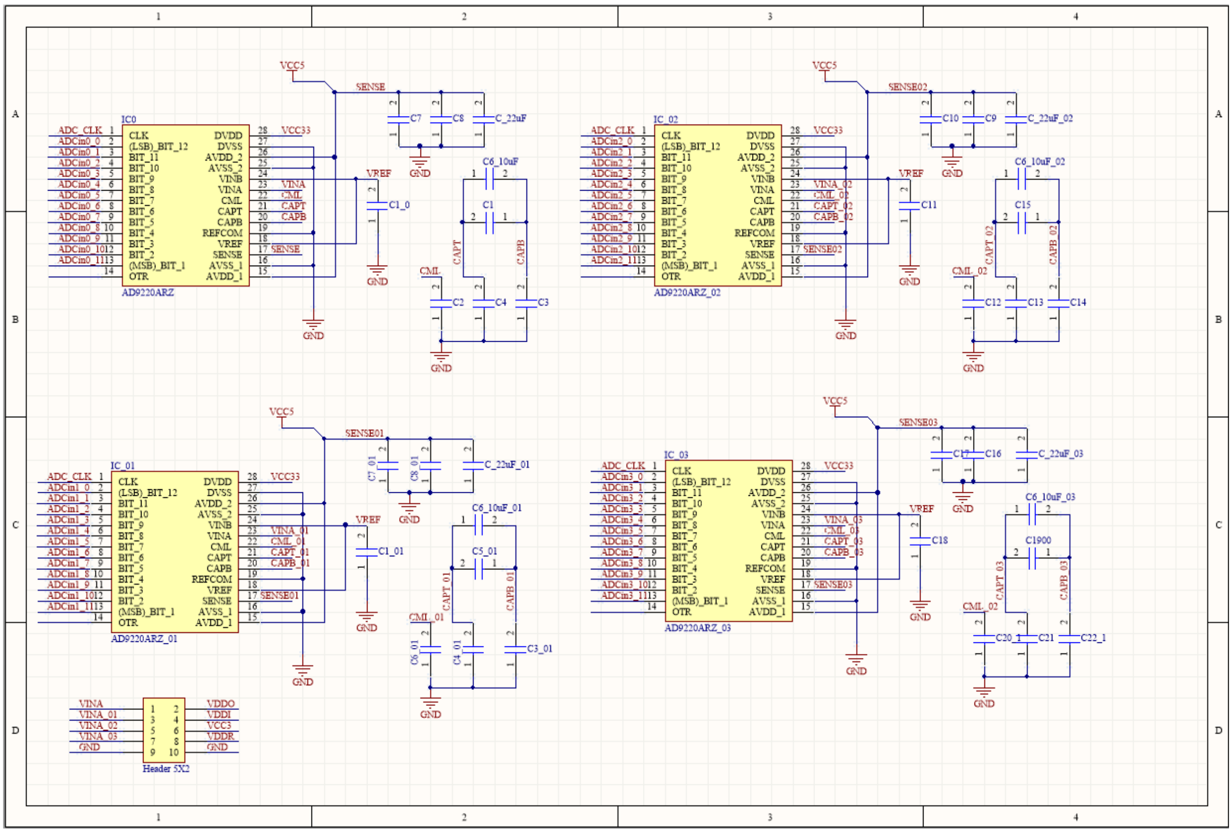
Date: 04/07/2018

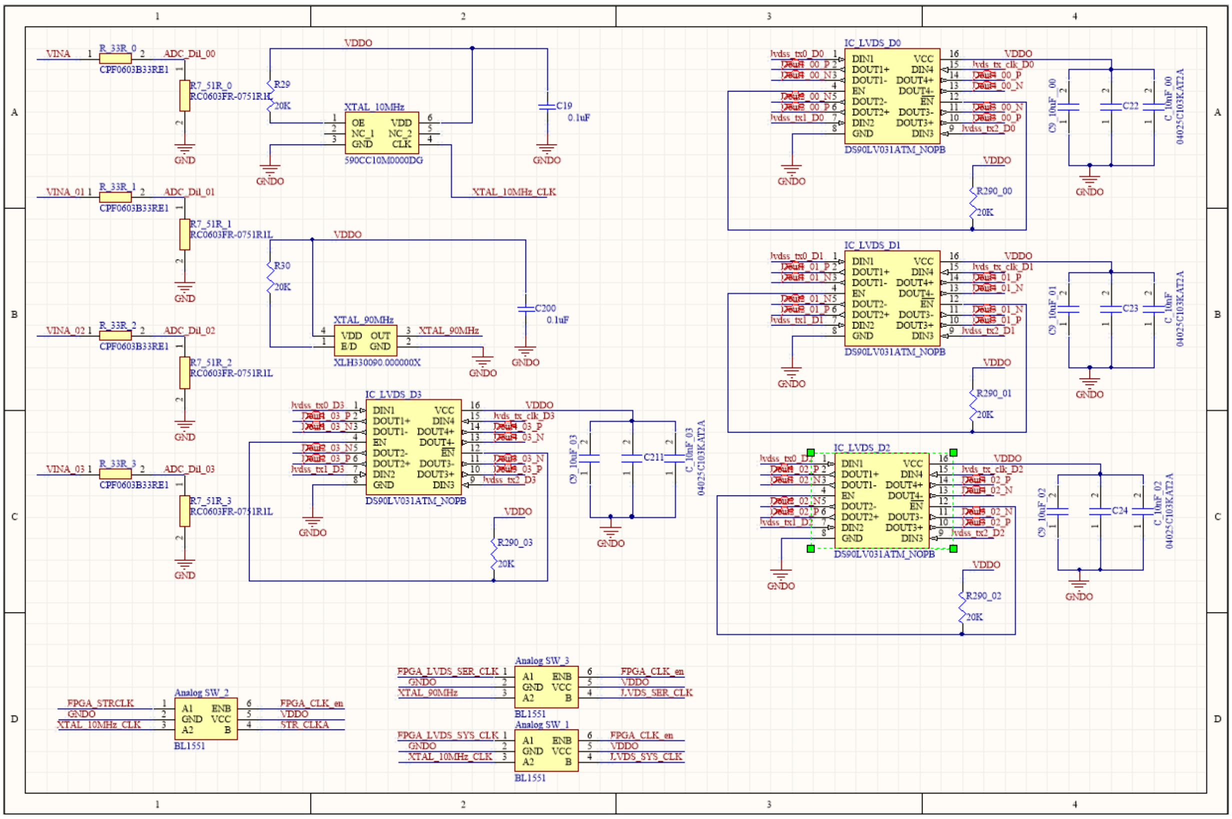
BOM - CPV SEGMENT CARD

Manufacturer Part Number	Manufacturer Name	Supplier Part Number	Supplier	Description	Designator	Quantity / Card	Value	Package Type	Total Order Qty
SN74LVTH241PWR	Texas Instruments	595-SN74LVTH241PWR	Mouser	Buffers & Line Drivers Tri-State ABT Octal	SN74LVTH241PWR	1	Nil	TSSOP-20	30
TXS0102DCUR	Texas Instruments	595-TXS0102DCUR	Mouser	Translation - Voltage Levels 2-Bit Bi-Directional Voltage-Level-Trans	D1, D2, D3, D4, D5, D6, D7, D8, D9, D10, D11, D12, D13, D14, D15, D16, D17, D18, D19, D20, D21, D22, D23, D24, D25, D26, D27, D28, D29, D30, D31, D32, D33, D34, D35, D36, D37, D38, D39, D40, D41, D42, D43, D44, D45, D46, D47, D48, D49, D50, D51, D52, D53, D54, D55, D56, D57, D58, D59, D60, D61, D62, D63, D64, D65, D66, D67, D68, D69, D70, D71, D72, D73, D75, D76, D77, D79, D80, D81, D82, D83, D85, D86, D87, D88, D89, D90, D91, D92, D93, D94, D95, D96, D97, D98, D99	100	Nil	VSSOP-8	2,500
EEU-FS1V102LB	Panasonic	667-EEU-FS1V102LB	Mouser	Aluminium Electrolytic Capacitors - Leaded 35VDC 1000uF 10000H 10x25mm	C17	1	1000uF	Radial	30
FKP0D001000B00KSSD	WIMA	505-FKP0D001000BKSSD	Mouser	Film Capacitors 100pF 100 Volts 10%	C2, C3, C4, C6, C7, C8, C10, C11, C12, C14, C15, C16, C18, C19	14	100pF	Radial	336
T821110A1S100CEU	AMPHENOL	2215304	Farnell	IDC-Header 10-pin connector	ETH_BLas_JTAG_IN1,ETH__JTAG_OUT1,JP1,JP2	5		Through-Hole	15
ERJ-P03D1001V	Panasonic	667-ERJ-P03D1001V	Mouser	Thick Film Resistors — SMD 0603 1Kohms 0.2W 0.5% AEC-Q200	R3,R45,R46	3	1k	603	40
ESS106M050AE2AA	KEMET	80-ESS106M050AE2AA	Mouser	Aluminium Electrolytic Capacitors - Leaded 50V 10uF 20% 105C Radial	C1,C5,C9,C13,C17	5	10uF	105C Radial	40
5145154-4	AMP - TE CONNECTIVITY	1284366	Farnell	Dilogic- Card connector	J1,J2,J3,J4	4			24
HSEC8-1100-03-L-DV-A-K	Samtec	HSEC8-1100-03-L-DV-A-K	Farnell	HSEC8-1100-03-L-DV-A-K - Connector, HSEC8-DV Series, Card Edge, 200 Contacts, Receptacle, 0.8 mm	C20	1		Card Edge connector	24
39293026	Molex LLC	WM3843-ND	Digikey	Molex DC Power Connector	POW_CONN_IN1,POW_CONN_OUT1	2	DC Power Connector		48
870056174001	Wurth Electronics	710-870056174001	Mouser	Aluminium Electrolytic Capacitors - Radial Leaded WCAP-PTHR Aluminum 10uF 20% 80V	C100	1	10uF	Straight	30
LT1764AET-2.5#PBF	Analog Devices / Linear Technology	584-LT1764AET-2.5PBF	Mouser	LDO Voltage Regulators 2.5Vout Fast Transient 3A LDO	U_LT1764	1	LDO Voltage Regulator	Through-Hole	30
885012006038	Wurth Electronics	710-885012006038	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT WCAP-CSGP 100pF 0603 5% 25V MLCC	C101	1	100pF	0603 (1608 metric)	30
885012006032	Wurth Electronics	710-885012006032	Mouser	Multilayer Ceramic Capacitors MLCC - SMD/SMT WCAP-CSGP 10pF 0603 5% 25V MLCC	C102	1	10pF	0603 (1608 metric)	30

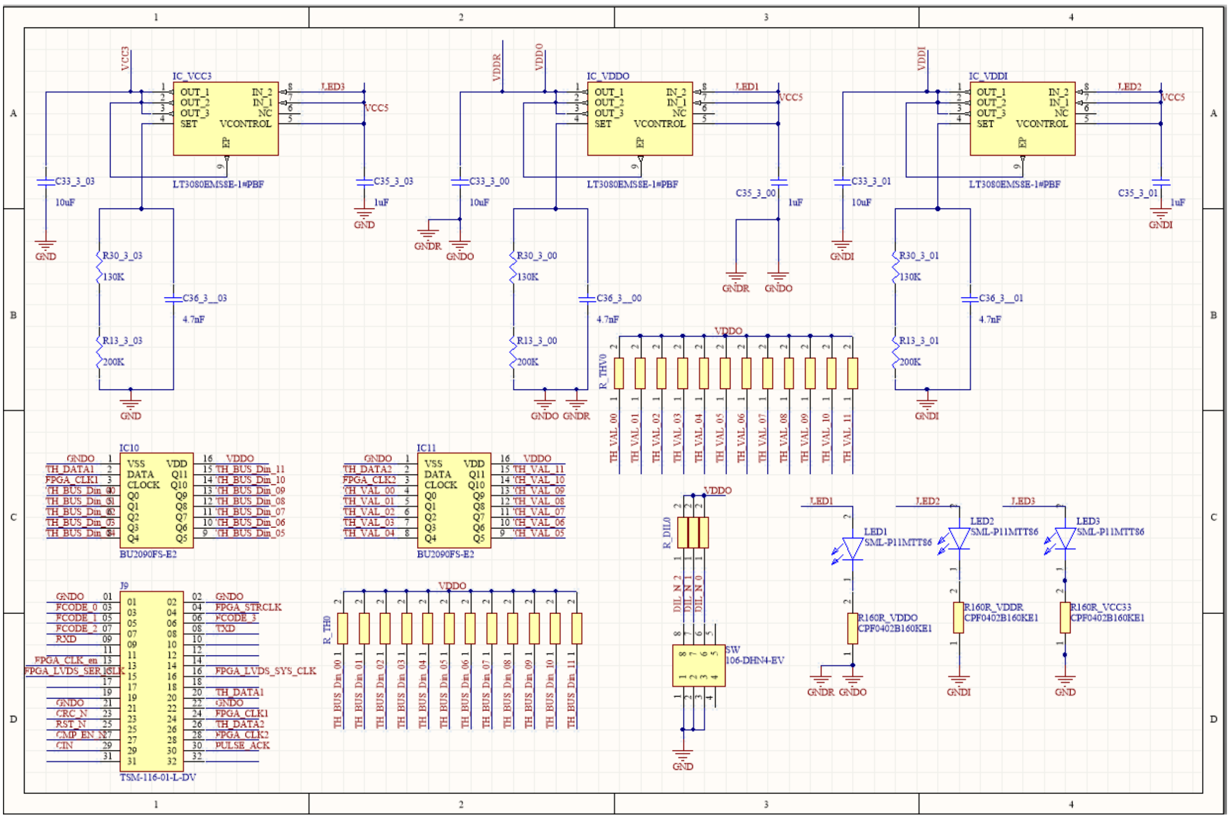
Appendix G – MDART PCB Test Jig Circuit



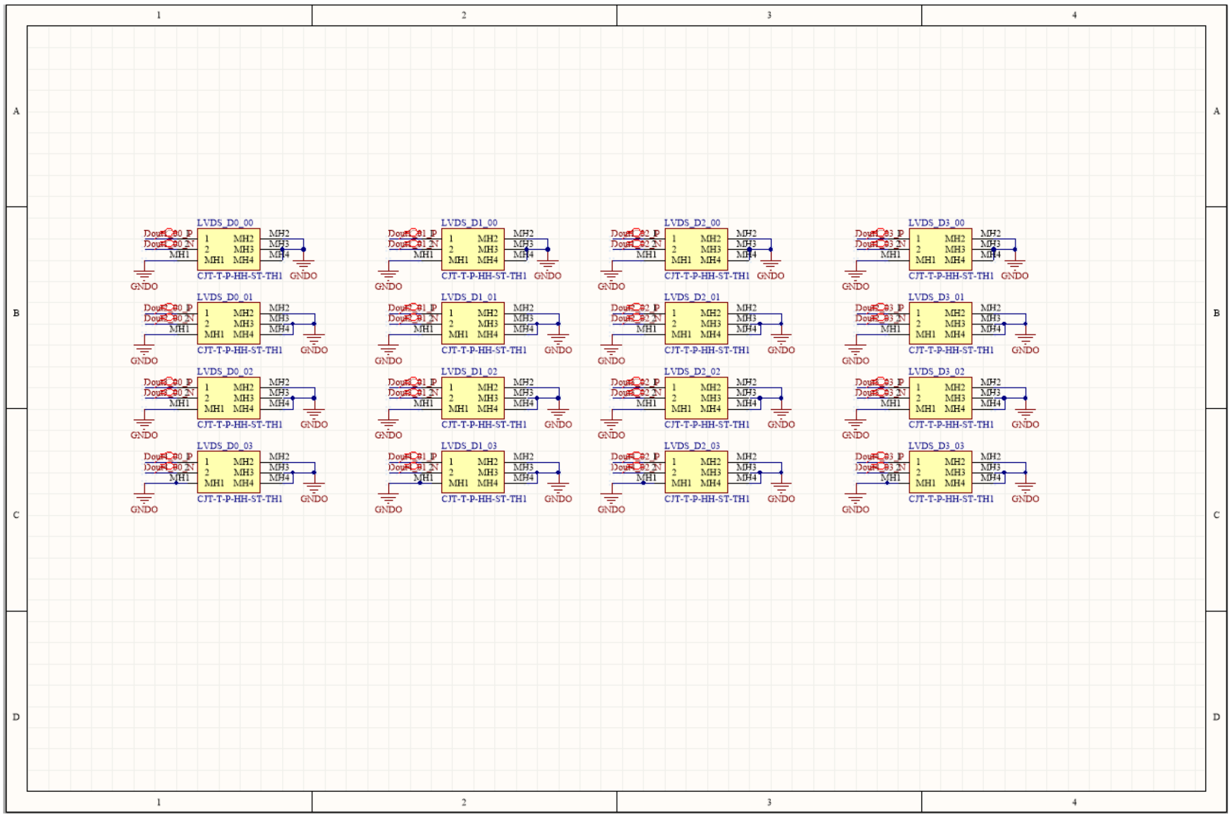




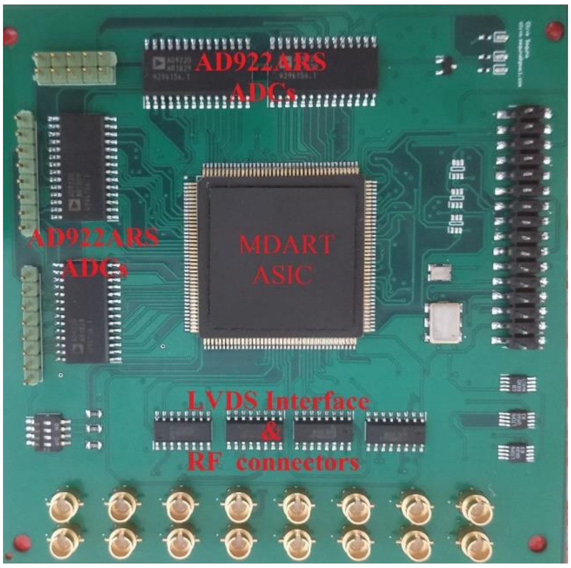
VINA
R_33R_0
ADC_Dil_00
CPF0603B33RE1
R7_51R_0
RC0603FR-0751RIL
GND
VDDO
R29
20K
XTAL_10MHz
OE VDD
NC_1 NC_2
GND CLK
590CC10M0000DG
C19
0.1uF
GNDO
XTAL_10MHz_CLK
VINA_01
R_33R_1
ADC_Dil_01
R7_51R_1
R30
XTAL_90MHz
VDD OUT
E/D GND
XLH330090.000000X
C200
0.1uF
VINA_02
R_33R_2
ADC_Dil_02
R7_51R_2
VINA_03
R_33R_3
ADC_Dil_03
R7_51R_3
IC_LVDS_D0
IC_LVDS_D1
IC_LVDS_D2
IC_LVDS_D3
DIN1 VCC
DOUT1+ DIN4
DOUT1- DOUT4+
EN DOUT4-
DOUT2- EN
DOUT2+ DOUT3-
DIN2 DOUT3+
GND DIN3
DS90LV031ATM_NOPB
R290_00
R290_01
R290_02
R290_03
20K
C22
C23
C24
C211
C9_10nF_00
C_10nF_00
04025C103KAT2A
Analog SW_2
Analog SW_3
Analog SW_1
A1 ENB
GND VCC
A2 B
BL1551
FPGA_STRCLK
XTAL_10MHz_CLK
FPGA_CLK_en
STR_CLKA
FPGA_LVDS_SER_CLK
XTAL_90MHz
LVDS_SER_CLK
FPGA_LVDS_SYS_CLK
LVDS_SYS_CLK

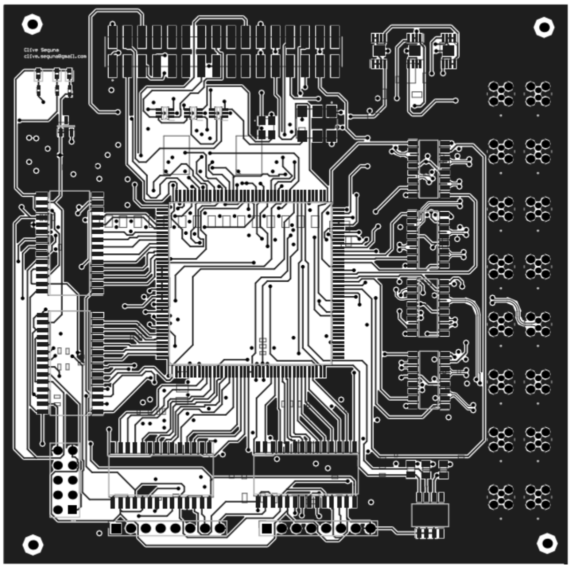


VCC3
IC_VCC3
OUT_1 IN_2
OUT_2 IN_1
OUT_3 NC
SET VCONTROL
LT3080EMSSE-1#PBF
LED3
VCC5
C33_3_03
10uF
C35_3_03
1uF
GND
R30_3_03
130K
C36_3_03
4.7nF
R13_3_03
200K
VDDR
VDDO
IC_VDDO
LED1
C33_3_00
C35_3_00
GNDR
GNDO
R30_3_00
C36_3_00
R13_3_00
VDDI
IC_VDDI
LED2
C33_3_01
C35_3_01
GNDI
R30_3_01
C36_3_01
R13_3_01
R_THV0
TH_VAL_00
TH_VAL_11
IC10
IC11
VSS VDD
DATA Q11
CLOCK Q10
BU2090FS-E2
TH_DATA1
FPGA_CLK1
TH_BUS_Din_00
TH_DATA2
FPGA_CLK2
R_DIL0
DIL_N_2
DIL_N_1
DIL_N_0
SW
106-DHN4-EV
LED1
SML-P11MTT86
LED2
LED3
R160R_VDDO
R160R_VDDR
R160R_VCC33
CPF0402B160KE1
J9
TSM-116-01-L-DV
FCODE_0
FCODE_1
FCODE_2
FCODE_3
RXD
TXD
FPGA_CLK_en
FPGA_LVDS_SER_CLK
FPGA_LVDS_SYS_CLK
FPGA_STRCLK
CRC_N
RST_N
CMP_EN_N
CIN
PULSE_ACK
R_TH0
TH_BUS_Din_00
TH_BUS_Din_11

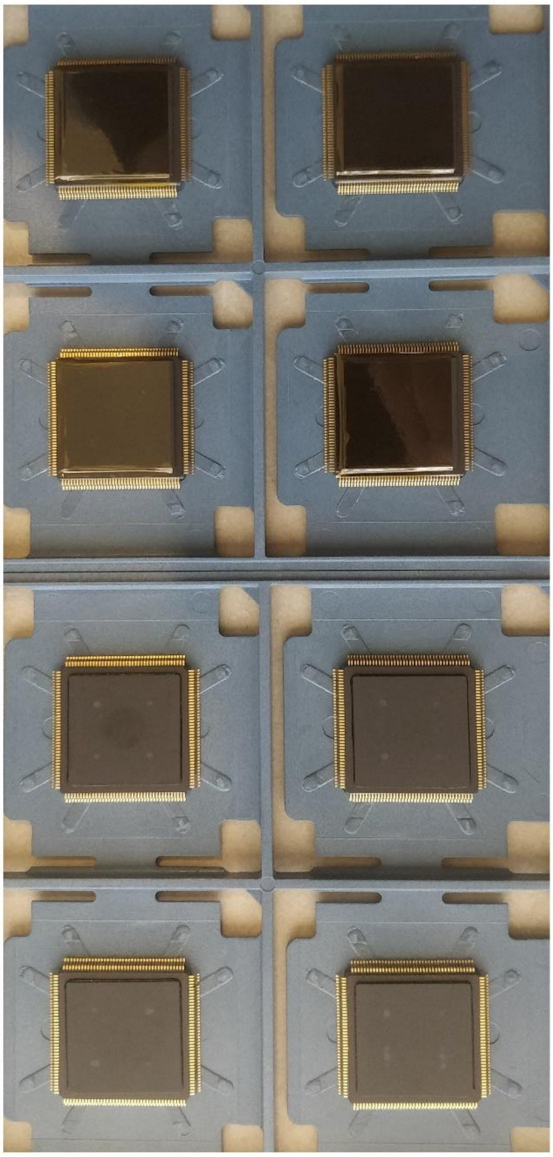

1
2
3
4
A
B
C
D
LVDS_D0_00
LVDS_D1_00
LVDS_D2_00
LVDS_D3_00
LVDS_D0_01
LVDS_D1_01
LVDS_D2_01
LVDS_D3_01
LVDS_D0_02
LVDS_D1_02
LVDS_D2_02
LVDS_D3_02
LVDS_D0_03
LVDS_D1_03
LVDS_D2_03
LVDS_D3_03
CJT-T-P-HH-ST-TH1
MH1
MH2
MH3
MH4
GNDO

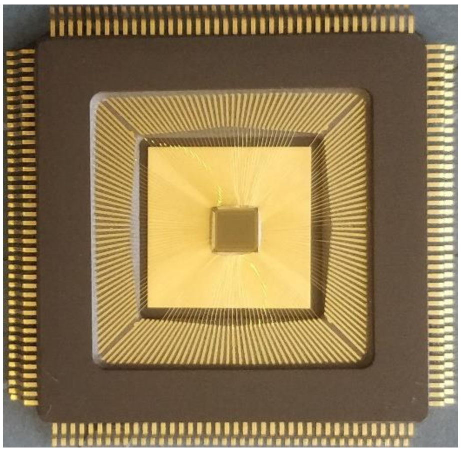
Appendix H – MDART PCB Test Jig Circuit, and Layout



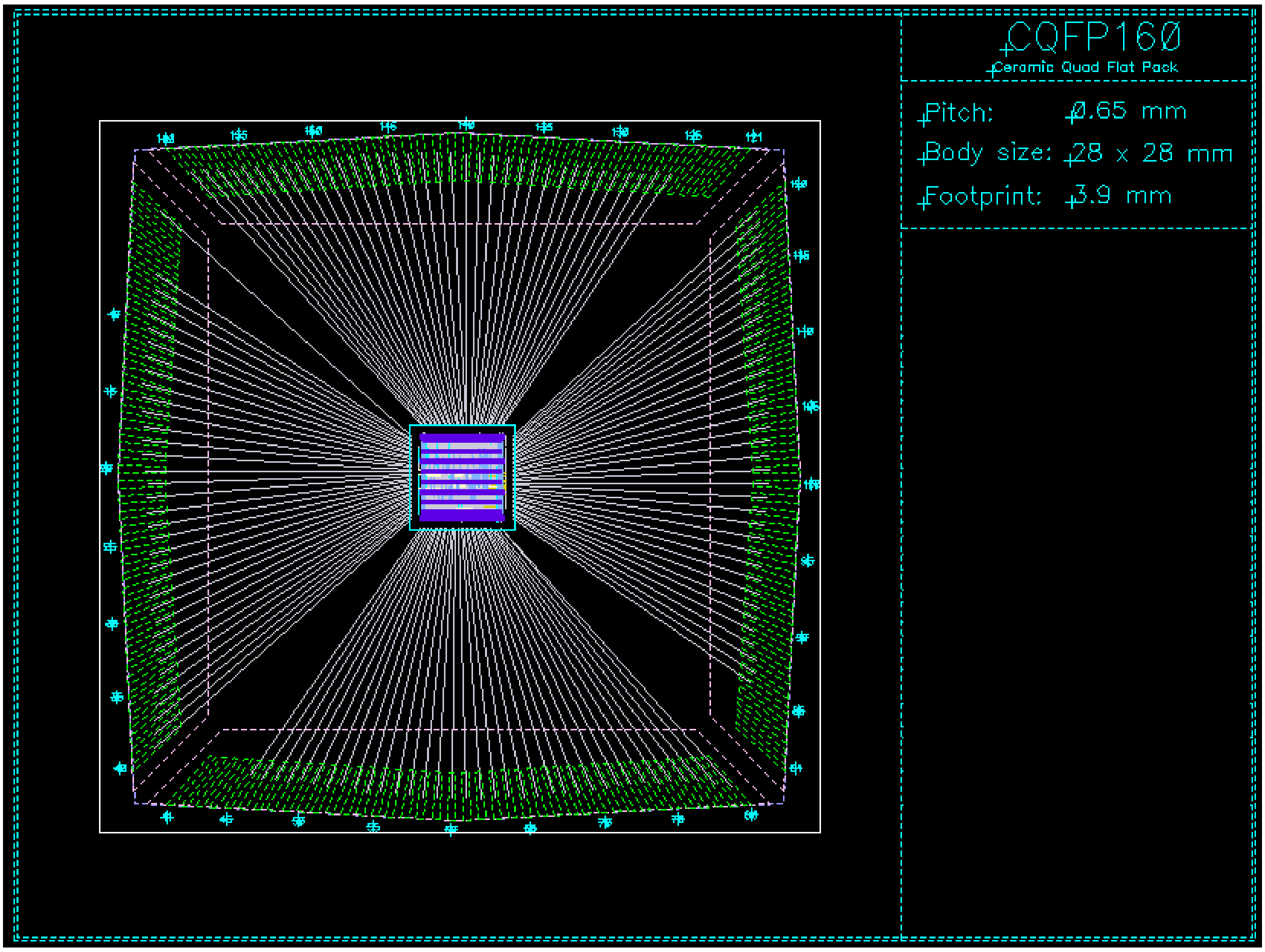


Appendix I – Samples of the Manufactured MDART ASIC





Appendix J – Bonding Diagram for the Manufactured ASIC


CQFP160
Ceramic Quad Flat Pack
Pitch: 0.65 mm
Body size: 28 x 28 mm
Footprint: 3.9 mm