Design of constrained Digital Signal Processing Systems



Marco Ghibaudi

ReTiS Lab.

Scuola Superiore Sant'Anna

A thesis submitted for the degree of

Doctor of Philosophy

Tutor: Prof. Marco Di Natale
Supervisor:
Dr. Stefan Haas (CERN) and Dr. Paolo Pagano (SSSUP)

June the XXth, 2014

Contents

Introduction 1

Part 1. Cost constrainted systems 3

Chapter 1. SEEDEYE, a Multimedia Wireless Sensor Network node 5
1.1. MWSNs and ITS integration: a step towards intelligent cities 6
1.2. SEEDEYE: Design of a MWSN node 9
1.2.1. SEEDEYE: Design of a MWSN node 9
1.2.2. The image processing 11
1.2.3. Sharing the information 12
1.2.4. The SEEDEYE board finalized 13

Chapter 2. SEEDEYE for ITS applications 15
2.1. Frame Difference Implementation 16
2.2. Supervised Machine Learning Implementation 17
2.2.1. Preprocessing function 19
2.2.2. Network Structure 22
2.2.3. Porting on the SEEDEYE 25

Chapter 3. SEEDEYE for streaming applications 27
3.1. Video streaming 27
3.2. Audio streaming 32

Part 2. Throughput constrained systems 37

Chapter 4. A DSP for Ultra-Fast Optical Communications 39
4.1. DSP implementation of an Ultra-Fast Optical Receiver 41
4.1.1. The structure of the transmitter. 42
4.1.2. The structure of the receiver. 42
4.1.3. Implementation of the system. 43
4.1.4. Frequency estimation and compensation 44
4.1.5. FeedForward Equalizer (FFE) 44

Chapter 5. Field Programmable Gate Technology 47
5.1. Clock network. 47
5.2. Configurable Logic Blocks. 49

5.3. Internal memories. 50
5.4. Digital Signal Processor blocks. 51
5.5. Available resources. 52
5.6. Development device 53

Chapter 6. A model for the DSP core 55
6.1. Inputs generation. 56
6.2. Automatic Frequency Compensation (AFC) 57
6.3. Power normalization and DC removal blocks 58
6.4. FFE block 60
6.5. Symbol decision 62
6.6. Coefficients update 63
6.7. Fixed Point conversion 64

Chapter 7. System implementation and validation 67
7.1. Converted model: preliminary results 68
7.2. A Multi-Instances implementation 70

Part 3. Latency constrained systems 73

Chapter 8. Digital Signal Processing in High Energy Physics Experiments 75
8.1. The LHC and the ATLAS Experiment 75
8.2. The Central Trigger Processor 77
8.3. CTP upgrade: motivations 80

Chapter 9. The Upgrade of the Central Trigger Processor Core Module 81
9.1. The CTPCORE+ module 81
9.2. Design validation: the Demonstrator setup 83
9.3. The Power Distribution Network 84
9.4. High speed links 86
9.5. Automatic pinout verification tool 87

Chapter 10. Firmware design 91
10.1. Monitoring and Control Interface 91
10.2. SDRAM Memory Controller 92
10.3. Chip to Chip communication protocol 93
10.4. Firmware validation 95

Chapter 11. CTPCORE+ module: validation and testing 97
11.1. CTPCORE+ board validation 97
11.1.1. Clock generator 97
11.1.2. Connectivity tests 98
11.1.3. Chip to Chip links testing 99
11.1.4. Power consumption and thermal dissipation investigations 100

11.2. CTPCORE+ Firmware 102
11.2.1. VME Interface 102
11.2.2. Firmware finalization 104

Bibliography 105

Introduction

IN modern society, telecommunications, economy, transportation systems, medicine and science heavily rely on electronic devices for transforming huge amount of data into information. To a great extent, generating, sharing and using information are the main purposes of all these systems. In almost all the applications, due to the higher level of accuracy and reliability achievable, data are processed mainly in their digital representation [**PM08**]. Being the signals digital, the processing of the data is referred as *Digital Signal Processing*.

Digital Signal Processing — from now on the acronym **DSP** will be used equivalently — can be seen as a toolbox containing all the tools (hardware components and software routines) that can be exploited for extracting, from the data, the information required. It must be stressed out that, to a large extent, the same information can be obtained in extremely different ways. Hardware and software can be seen as block-sets that — if properly combined — can produce systems capable of satisfying the requirements. Common project specifications contain constraints like:

- *Total cost of the system must be lower than* $\mathscr{C}$.
  Designers must consider that in a wide range of applications, $\mathscr{C}$ will be composed by the cost of the device and by the cost for running and maintaining the system;
- *Throughput of the system must be higher than* $\mathscr{Q}$.
  $\mathscr{Q}$ is a measure of the amount of information that can be generated per unit of time by the system.
- *Latency of the system must be lower than* $\mathscr{T}$.
  With $\mathscr{T}$, the time required by the system for transforming the input signals into the requested information.

Whenever one of these aspects gains more importance over the others, alternative design techniques, as well as, different hardware and software should be considered. To a large extent, designers of cost critical systems will run simple algorithms on low cost hardware units in order to minimize both hardware and software design costs. For the design of throughput constrained systems, in order to maximize the amount of operations performed per unit of time the design will be parallelized/pipelined, or faster hardware will be used. When the system latency represents the main constraint, the determinism of the execution duration will be considered a basilar goal. Given the complexity and variability of the process, a unified design approach seems hardly achievable in the near future. When addressing the implementation of a new DSP system, the designer should reuse solutions and know-how, referring whenever is possible to existing systems.

This work presents three different systems, attaining to the three afore-mentioned scenarios. **Part I** presents the research activity that has been conducted on a specific family of cost constrained systems

adopted in an Infomobility project, **Part II** addresses the solution adopted for maximizing the throughput in a Telecommunication project while **Part III** presents the work conducted on a latency critical component of an High Energy Physics experiment.

Part 1

Cost constrainted systems

CHAPTER 1

SEEDEYE, a Multimedia Wireless Sensor Network node

NOWADAYS , a large number of systems rely on distributed sensing for implementing advanced functionalities. Multiple sensing units are connected forming a so-called Sensor Network (SN). Whenever the communication between the nodes is achieved through wireless links — as in the almost all the applications — the system is referred as Wireless Sensor Network (WSN). Examples of WSNs can be found everywhere. Industrial plants are controlled by sophisticated networks of RFID transponders, temperature and acceleration sensors [**SBP+12**], forest and crops are monitored by distributed smoke, moisture and temperature sensors [**SLXX10**], enemy territories can be patrolled in a human-free fashion through cameras and microphones [**WTK11**] and cities start to be monitored by networks of CO2 sensors and smart cameras [**SGG+13**].

WSNs are composed by nodes, sometimes referred as motes, that can be connected in different topologies. To a great extent, motes are cost constrained hardware devices that implement three main functionalities: they provide an interface to the sensors, they process the sensor readings for generating information and, finally, they transmit the extracted information through the wireless interface. The complexity and the associate computational cost of the intermediate data processing passage depend on the processing paradigm adopted. In some scenarios, simple averaging operations suffice for extracting the requested information, while in others, complex neural networks [**GPK11**] or advanced filtering operations [**BGS+11**] are needed. According to Taj et al. [**TC11**], WSNs can process data in three fashions:

- *Centralized processing.*
  A single node, called *sink*, is in charge of actually transforming data into information, while all the other sensor nodes of the network (*sources*), are simply transmitting their readings. To a great extent, *sinks* are connected to the power grid and thus not significantly limited in terms of computational resources.
- *Decentralized or clustered processing.*
  Within a group of nodes (a *cluster*), one of the units is in charge of the real data processing, while the other members of the network are simply sending to it the readings collected from their sensors. Compared to the centralized processing, this approach significantly reduces the amount of data that has to be shared between the nodes.
- *Distributed processing.*
  Each node performs the largest part of the on board signal processing, transmitting only aggregated pieces of information on the network.

To reduce the transmission cost, the modern tendency is to move from the centralized approach towards a distributed one. Historically, WSN nodes have been designed for transmitting small amounts

of data, read from the sensors, to more powerful units, the sinks. The perfect example of this kind of application is the monitoring of forests. Kovàcs et al. [**KMH10**] proposes a solution based on simple motes that can provide an effective fires detection system. In their proposal, the deployed motes have on-board temperature, moisture and smog sensors that are queried every 10 seconds. When the on-board sensors detect a fire, an alarm is transmitted to a sink unit that confirms it via an acknowledge message. In normal conditions, with no fire detected, a heartbeat packet is sent every fifteen minutes. The transmission bandwidth required by each node is so small (in the order of tens of bps) that even with a transmission bandwidth smaller than 100 kbps, hundreds of nodes can operate in the same network using, for example, a Time Division Medium Access (TDMA) policy. Concerning the computational power of the sink unit, if it happens to be insufficient, this unit can be easily substituted by a more powerful one.

Unfortunately, for an increasing number of WSN applications, bandwidth and power consumption limitations are preventing the adoption of the centralized processing approach. The former constraint originates from the wireless protocols used by WSNs. As a matter of example, the most largely adopted one, the IEEE 802.15.4 protocol provides a net data throughput of roughly 100 kbps at the application layer. With a centralized approach, when the number of nodes increases, the establishment of all the node-sink links may require a bandwidth that overpasses the channel budget. Even where the node number has not changed the amount of data that each node must send has increased. Scalar data, such as acceleration or temperature cannot provide some valuable pieces of information that can be obtained by using different type of sensors. Multi-dimensional data, in particular images and audio signals, given their higher information content, start to be exploited for extending the applicability of the WSNs technology. Objects tracking [**Zha11**] and human behavior recognition [**DPFG+09**] are examples of functionalities that the so-called Multimedia WSNs (MWSN) can implement by exploiting multidimensional/multimedia data coming from cameras and microphones. Unfortunately, the bandwidth required for transmitting raw images or unprocessed audio signals is two-three orders of magnitude higher than the one required for sending scalar readings. As a result, MWSN cannot adopt a centralized processing approach and they must rely on a decentralized or a distributed schema.

As a result, motes that can be used effectively for traditional WSN networks cannot be exploited in MWSNs, due to their limited computational power. In fact, WSN nodes are now in charge of performing complex digital processing operations that are required for extracting significant information from a relatively large amount of data. MWSN nodes are generally considered more complex to design that traditional WSN, in both the hardware and software components and unfortunately, there are no tools available for simplifying these processes. The design of the hardware and software elements composing one of the first MWSN units will be investigated in the proceedings, focusing on various DSP algorithm deployed on top of it.

1.1. MWSNs and ITS integration: a step towards intelligent cities

One of the most promising application fields for the WSN technology is represented by the so-called *Intelligent Transport Systems*, ITS. ITS can increase the quality of all the transportation systems by improving their security, efficiency and effectiveness. ITS has received approval and investments in

a significant number of countries, thanks to the decisive support received from the European Parliament [**EU10**]. Massive efforts and investments have been spent in the last years and the technologies look mature enough for a large scale adoption [**PPAS13**].

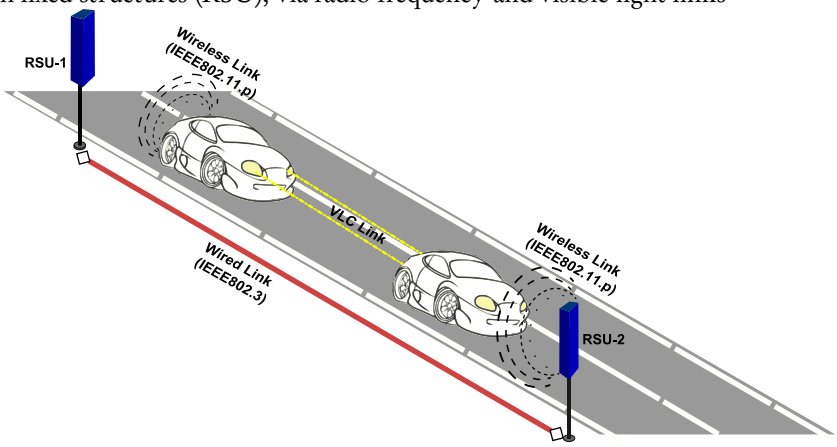
In particular, academia and industry are proposing solutions that can be brought back to two different approaches. On the one hand side, solutions for increasing the "intelligence" and security of vehicles are investigated, while on the other hand side, the introduction of distributed monitoring systems aims at boosting the efficiency of cities and roads. The two approaches are extremely promising and they deserve a more in-depth analysis.

Focusing on the so called "on-board intelligence", various examples can better characterize this approach. Starting from commercial products, it can be easily seen that state-of-the-art navigation systems can actively produce pieces of information that other units can use to enhance the driving experience. Notifications of traffic jams, road blocks and car incidents can be used by navigation systems for updating the route, minimizing travel time and increasing drivers security. This approach relies on a centralized approach: computer farms aggregate and elaborate data received from sensing units installed on the road sides. The information inferred from these readings is finally broadcasted through cellular networks to the navigation systems and thus to the users. Navigators must be connected to the mobile network infrastructures for receiving the notifications and, unfortunately, cost associated to the data transmission represents the only significant drawback of this approach.

In order to remove this limitation, a significant number of researches are investigating distributed processing approaches. In these systems, data collected by a vehicle are shared locally with "neighbour units", that are represented by other vehicles or units installed on the roads sides. The distributed approach has two benefits. First, the communication between cars can prevent accidents, alerting preventively the drivers of dangerous situations that are not easy to spot (e.g. a flat wheel of another vehicle, slippery road ahead, accidents). Second, the RSUs, being connected through cabled networks, can distribute these pieces of information in a free-of-charge fashion to all the connected nodes, significantly boosting the transportation efficiency. Following this approach, Petracca et al. [**PPP+12**] propose a low cost device that supports the IEEE802.11p standard for vehicular networks. Their prototype can query the content of internal Electronic Central Unit (**ECU**) of modern vehicles, providing an easy way to access and transmit to other vehicles extremely valuable pieces of information (e.g. ABS and brakes status). Units supporting the IEEE802.11p standard can receive the data read from the ECU and process them for inferring important viability related pieces of information. The RSU, in particular, can be connected to network backbones for distributing this information to the other road users.

A complimentary activity has been conducted by Corsini et al. [**CPC+12**] that are proposing their solution for establishing a high speed car to car communication link characterized by an extremely an incredible high throughput and low setup time. The authors adopt the so-called Visible Light Communication (VLC) technology for providing high bandwidths between vehicles in the same Field Of View (FOV). The high communication bandwidth achievable via the VLC technology - in the order of gigabits per second - represents an extremely valuable feature of this proposal while the IEEE802.11.p protocol addressed, allows a lightweight/low latency network establishment procedure. Figure 1 depicts a possible integration of these two different proposals.

FIGURE 1. Integrated ITS scenario. Cars can exchange information among them and with fixed structures (RSU), via radio-frequency and visible light links



Moving the focus from the Smart Vehicles to the Road Side Units, it can be seen how these components represent the cornerstone of the Smart Cities ITS revolution. Single nodes have limited coverage but when multiple units are dislocated, large areas can be extensively covered. The information gathered by the nodes can be sent to control centres that can operate traffic lights and information panels for enhancing roads safety. This approach imposes a great pervasiveness of the sensing units. A higher density of nodes, in fact, provides a better insight of the roads, allowing corrective operations to take place on a real-time basis. RSUs can be implemented with different technologies but a significant number of studies have identified WSNs - MWSNs in the specific - as the best candidate. To corroborate this claim, various projects based on this approach have produced excellent results. IPERMOB [**MRP+11**], for example, that was based on MWSN nodes produced an effective system that was tested with a public demo session at the Pisa Airport. This project produced a set of devices, infrastructure and systems capable of gathering extremely valuable traffic related information, useful for enhancing the driving experience and increasing roads security. IPERMOB comprised three main subsystems:

- *A parking monitoring system.*
  IPERMOB exploited a low cost system, MWSN based, for monitoring the utilization of a parking area. Compared to traditional approaches based on magnetic or ultrasonic sensors, MWSNs motes used in IPERMOB can implement a less costly and more accurate system.
- *A traffic monitoring system.*
  MWSN nodes were exploited for monitoring portion of the roads. The nodes were used for extracting information such as the number of vehicles per hours and their average speed.
- *A control and communication infrastructure.*
  The parking monitoring system and the traffic flow systems were connected to a control room via high speed wireless links. The control room stores the information collected by the sensors in unique entries of a database that could be accessed remotely.

While the control and communication infrastructure was largely implemented by means of commercial devices, the motes used for monitoring the parking area and for analysing the traffic flows were

custom designs. Among the developed devices, the so-called SEEDEYE board represents an excellent example of Multimedia Wireless Sensor Network node. This board was in charge of extracting the occupancy of the parking area and was adopted as communication-bridge for connecting IEEE802.15.4 operating nodes with Ethernet-based devices. The design of this board will be addressed in the proceeding.

1.2. SEEDEYE: Design of a MWSN node

The SEEDEYE board represents one of the first commercial specimens of Multimedia Wireless Sensor Network. The target of IPERMOB was the implementation of an extremely pervasive system, thus the SEEDEYE board was designed for having a reduced cost and a minimal power consumption while delivering to the system the functionalities required. The IPERMOB decentralized processing approach required that the SEEDEYE could:

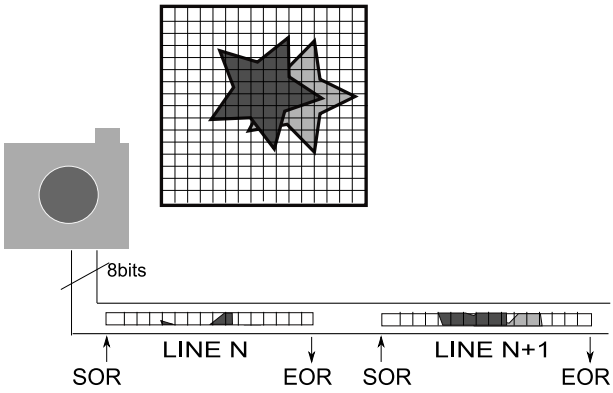
- Acquire images through a camera sensor and store a sufficient number of them for processing.
- Run computer vision algorithms over the stored images for extracting the required information.
- Provide interfaces to the IEEE802.15.4 and IEEE802.11 networks for sharing the features extracted and for providing maintenance functionalities.

1.2.1. SEEDEYE: Design of a MWSN node.

The acquisition and storage of the images significantly constrained the design process. The Microcontroller Unit (MCU), in particular, was selected after the definition of some properties related to the acquisition process. The image resolution and the colour space adopted were the main points addressed during the MCU selection process. The image resolution sets constraints in terms of memory and computational resources demanded. It must be pointed out, that being the principal target of IPERMOB the minimization of the node cost, only commercial, low-cost cameras were considered. To a great extent, the frame size supported by these cameras spanned from QQVGA (160 by 120 pixels) to VGA (640 by 480 pixels). Unfortunately, being designed for standard applications, some aspects of these modules were not optimal for the IPERMOB application scenario. In particular, all the investigated modules output the acquired images on communication buses characterized by high baud-rates, following a schema similar to the one depicted in Figure 2.

Due to the nature of this transmission, two constraints aroused. On the one hand side, MCU ha to handle correctly these fast data transfers. Cameras output pixels in bursts, row by row, with line-rates ranging typically between 5 to 20 Mbps. Unfortunately, the communication baud-rate could not in general reduced below a certain value, imposing a constraint on the minimum frequency of the MCU (i.e. 2-4 times higher than the line rate). On the other one hand, data had either to be processed immediately — on the fly — or stored somewhere for successive processing. Concerning the first option, the transmission of the image is in general too fast for performing processing operations on the received rows. Even exploiting the Direct Memory Access feature available on some MCU for reducing the CPU load, the unused computational power may suffice only for performing simple sums and subtractions on the received image row. The second approach, instead, requires an amount

FIGURE 2. Acquisition scheme of a standard low-cost camera. Pixels composing a row of the image are transmitted in sequence at rates ranging from 5 to 20 Mbps, equivalent to a peak bandwidth of 5 to 20 MB/s. No data are transmitted between EOR and the successive SOR.



of memory proportional to the resolution selected and to the colour space adopted. If the MCU does not provide the required memory an external memory component must be interfaced to the MCU, increasing power consumption and board cost. Table 2 provides a simplified overview of the different memory requirements, presenting as well the typical cost and power consumptions of Static RAM memories. For the configurations marked with a *, MCUs with internal memory sufficient for storing at least one image exist.

The pieces of information presented in Table 2 were not sufficient for selecting the optimal MCU and for deciding whether or not to adopt an external memory. The number of frames that the processing algorithm requires, as well as their resolution, must be known before favouring an implementation over another.

TABLE 1. Cost versus image resolution

TABLE 2. Memory requirements and cost associated to different image resolutions. Cost are considered for small productions.

Resolution	Colour Space	Min. Size [B]	Cost [$]	Consumption [mA]
QQVGA(160x120)	Gray	19200*	0-2	0-2
QQVGA(160x120)	YuV	38400*	0-2	0-2
QVGA(320x240)	Gray	76800*	0-4	0-5
QVGA(320x240)	YuV	153600	2-4	2-5
VGA(640x480)	Gray	307200	4-8	10-20
VGA(640x480)	YuV	614400	4-8	10-20

1.2.2. The image processing.

In parallel with the design of the board, different computer vision algorithms were investigated, analysing their requirements in terms of computational power and memory. The memory demand depends on the image resolution and on the number of frames that must be analysed for extracting the required feature (a.k.a. the algorithm depth). These two aspects combined with the number of operations per pixel can provide the computational cost of the processing algorithm. In the case of the parking monitoring application, the frame resolution maps directly into the node effectiveness. When a higher resolution is exploited, the camera can be placed farther away from the investigated area, increasing the node field of view and consequently the maximum number of slots that can be simultaneously monitored. On other hand side, as shown in Table 2, the amount of memory increases significantly when higher resolutions are considered.

When IPERMOB was conceived, a significant number of traditional image processing tools were not available in a low computational cost flavour. As a matter of example the largely adopted Computer Vision library, OpenCV [**BK08**], did not support low-power MCUS at that time. Starting from these premises, Magrini et al. [**MMN+11**] investigated the feasibility of various computer vision processing approaches, estimating their computational cost and memory occupancy. Concerning the parking area monitoring functionalities, they identified two promising approaches:

- *Frame difference approach.*
  This approach relies on the definition of a zero-level reference image that is compared against the acquired images for identifying the status of the monitored parking slots. As a matter of example, the contrast of the acquired image can be compared to the one of the reference image for implementing a simple *change-detection* based algorithm.
- *Supervised algorithms.*
  Cascade of classifiers can be trained for recognizing vehicles distinctive features, such as car plates or windshields. The number and complexity of the classifiers can be adapted to the amount of resources available, trading accuracy for computational cost. Distributed algorithms can be considered, with a set of nodes implementing the different steps of the processing in order to reduce the single node load.

For both the approaches the memory requirements could be easily inferred. At least two images were required by almost all the investigated algorithms, requiring about 40 Kbyte and 80 Kbyte of memory when QQVGA and QVGA images are respectively considered. Unfortunately, an accurate estimation of the computational cost was not so straightforward, being this aspect deeply related to specific algorithm implementation. For this reason, a conservative approach had to be adopted, favouring MCUs characterized by higher computational capabilities.

Considering all these aspects, an MCU for the SEEDEYE board was finally identified: the PIC32MX-795L512 MCU (from now on, PIC32). The low-power 32 bits PIC32 microcontroller is equipped with 128 KB of internal RAM, has the instruction clock running at up to 80 MHz and can operate external buses operating at up to 40 MHz. Furthermore, it provides an advanced DMA controller that can be used for transferring data sampled at the input pins to any RAM address, without any CPU intervention. On the cons side, this MCU does not present a dedicated FPU or any DSP functionality that, if

required, must be implemented via software, but on the pros side, the power consumption of this MCU can be significantly reduced by exploiting the different power modes available on the microcontroller.

1.2.3. Sharing the information.

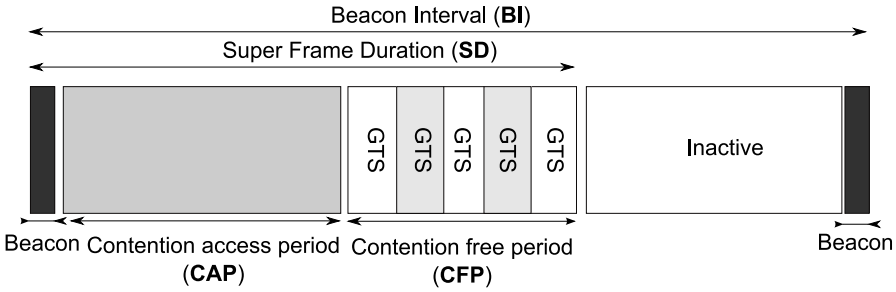
In IPERMOB, a bidirectional path between the control room and the sensing nodes was of fundamental importance. The information extracted by the WSN nodes were transmitted through IEEE802.15.4 links to aggregator nodes that, working as network gateway, forwarded these data to a control room via IEEE802.11 links. Requests and configuration messages from the control room travelled in the opposite direction, with the control room sending them to the WSN nodes, again relying on the aggregator nodes. In both the cases, the gateway has to implement a significant number of features:

- *Packet dispatching between the protocols.*
  IEEE802.11 and IEEE802.15.4 packets largely differ and the gateway node was in charge of the protocol conversion operations.
- *Medium access control for the IEEE802.15.4 interface.*
  In wireless communications, the probing of the channel does not suffice for avoiding transmission collisions and data losses. IEEE802.15.4 standard defines a way for preventing this scenario. A central node is used to schedule the channel accesses, implementing a Time Division Multiplexing Access policy.
- *Fragmentation/defragmentation of large packets.*
  The IEEE802.15.4 standard defines a maximum packet size of 127 bytes while IEEE802.11 networks usually operate with packets of 1500 bytes.

The SEEDEYE board, that was initially designed only with an IEEE802.15.4 transceiver, was extended adding an IEEE802.11 interface. The benefits of this approach were a lower design cost (one board had to be designed instead of two), shorter debug time, higher code reusability and enhanced reconfiguration capabilities of the network. Furthermore, being the SEEDEYE capable of completely power off the IEEE802.11 related components, no power consumption penalties were introduced into the system. A negligible increase in the node cost — largely paid back by the benefits provided — was the only drawback associated to this solution.

The afore-mentioned MCU represented an excellent choice also concerning the wireless interface. In fact, Microchip provides a high power IEEE802.15.4 transceiver, the MRF24J40MB that can be easily connected to any Microchip™microcontroller. This transceiver is largely customizable with its transmitter configurable in terms of amplification gain. In IPERMOB , the internode distance was extremely variable, with some nodes spaced few meters apart and others at up to 50 meters of distance. The MRF24J40MB transceiver — if operated at the maximum power of 100 mW allowed by the ETSI standard — can transmit at up to 100 meters in open field. In the far-from-ideal scenario of the airport, these transceivers managed to provide a significant good Bit Error Ratio for distances up to 50 meters [**PGS+11a**]. Given the good quality of the links, a simple Forward Error Correction scheme was adopted in the IPERMOB experiment, with every node manually configured for operating with a custom transmission gain. It must be pointed out that the possibility of adopting different transmission

FIGURE 3. IEEE 802.15.4 TDMA scheme. After an aligment beacon packet, all the nodes can access the channel (**CAP** phase). In this phase, collisions can occur and retransmissions are therefore requested. During the **CFP** phase, instead, time slots are defined and nodes can transmit only if the master of the channel grants them a slot. Duration of **CAP** and **CFP** can be tuned by the system designer.



gains for the different transceivers represents an excellent way for reducing the nodes interference and the overall power consumption.

Concerning the network topology, the SEEDEYE nodes were grouped into clusters, each of them connected to the wired backbone by gateway nodes. Control room could address directly the gateways using IPv4 addresses while for the sensing nodes an IPv4-to-6LoWPAN addresses mapping was performed by the Gateway. The SEEDEYE nodes used as gateways, runt the LwIP IEEE802.11stack as well as the uWireless IEEE802.15.4 stack.

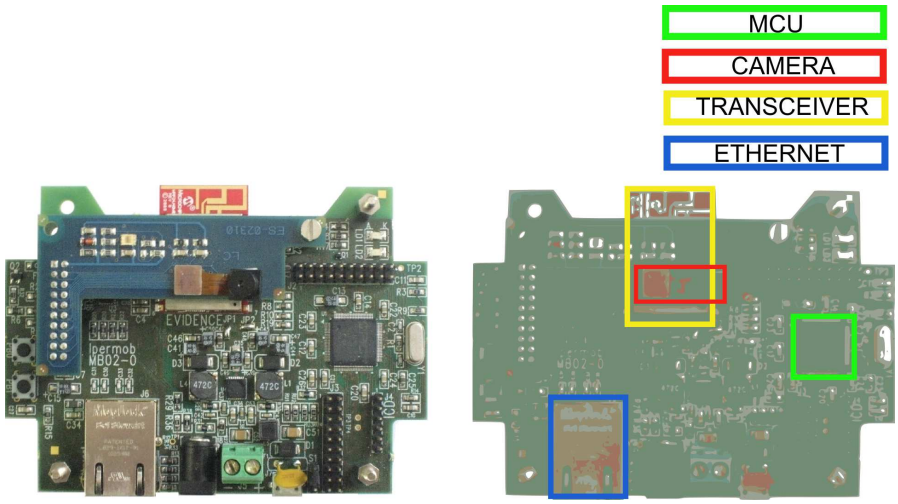
The first one provides, in a low complexity, small footprint fashion, the basic features of the Ethernet stack. An accurate description of it, as well as, the source code can be found in [**www14o**].

The latter, instead, implements all the synchronization, control and request handling features that a IEEE802.15.4 cluster coordinator must provide. This stack, by means of a Time Division Medium Access (TDMA) approach, provides an effective Medium Access Control policy. Figure 3 depicts the TDMA scheme used by this stack. Every SEEDEYE node is entitled to transmit data in a very specific time window, known as Granted Time Slot (**GTS**). The Coordinator/Gateway takes care of transmitting the beacon messages used by the cluster nodes for synchronizing. The mandatory timing reference is internally generated by the SEEDEYE by means of fine grain timers. A more in-depth description of the network stack is given by Nastasi [**Nas12**].

1.2.4. The SEEDEYE board finalized.

With the definition of the MCU, the transceivers (IEEE802.11 and IEEE802.15.4 compliant) and the camera interface it was possible to finalize the design of the board. Starting from the power distribution network, high efficiency switching regulators (90% of efficiency) are used for generating the two main voltage rails, 3.3V and 1.2V. The 1.2V rail, used exclusively by the IEEE802.11 transceiver, could be completely switched off through a MOSFET controlled by the microcontroller. An additional switch has been added for unpowering the camera interface, reducing almost to zero the power consumption related to the camera module. For maximizing the number of camera modules that can be connected

FIGURE 4. The SEEDEYE board.



to the SEEDEYE module, a special approach has been adopted. The SEEDEYE exports a set of signals that can be configured by the MCU for implementing different protocols (SPI, I2C, parallel I/O) as well as the power signals required by the camera module. Different camera modules can be easily designed for interfacing with the SEEDEYE module. While on the one hand side, this approach introduces additional costs, on the other one hand side, it increases the system extendibility, removing as well the problem of the camera obsolescence. The MCU and the IEEE802.15.4 transceiver are connected through an SPI interface, with the latter component optimally placed for maximizing the net transmission power. Additional sensors can be connected to the MCU through a connector: digital sensors (accelerometers, luminosity sensors, GPS) and analogic sensors (temperature sensors, current probes) can be easily interfaced to the SEEDEYE using external modules (daughter boards). Figure 4 shows the final version of the SEEDEYE board.

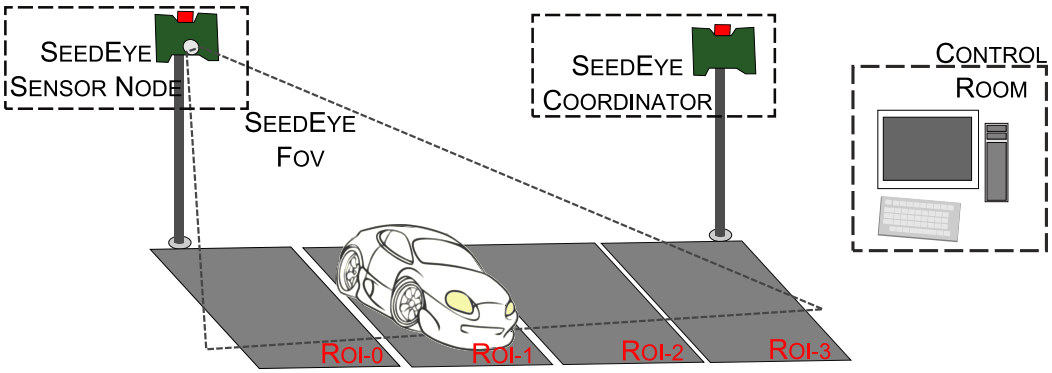
CHAPTER 2

SEEDEYE for ITS applications

THE SEEDEYE board was initially designed for running low complexity Computer Vision algorithms for object recognition and for event identification within the framework of the IPERMOB project. One of the principal applications of the board was the monitoring of the level of utilization of a parking area. In it, the system had to infer the status of various parking spaces - that will be referred equivalently as Region Of Interest (ROI) from now on - by using only the information contained in the image frames acquired by the camera. This information has to be refreshed when the device receives an update request from the so-called control room. The frequency of this request is quite low, with more than one minute of distance between two consecutive queries. The full processing chain is depicted in Figure 1.

Concerning the implementation of this system, being the node power consumption a basilar aspect, large efforts were spent to minimize the complexity of the DSP algorithm adopted. Two different algorithms, in particular, both based on gray-scale images, have been designed. The first one, fully developed during the IPERMOB active phase is characterize by an extremely low computational cost and can provide the maximum power saving. Unfortunately, the achievable accuracy is relatively limited and scenario dependent. The second approach, developed when the IPERMOB was entering the deployment stage provides more promising results but it is characterized by a significantly higher complexity.

FIGURE 1. IPERMOB parking areas monitoring subsystem. SEEDEYE nodes are used for monitoring the status of N parking spaces. Each SEEDEYE analyses different ROIs and identifies the presence or absence of a vehicle. This information is sent through a *Coordinator* to the *Control Room*.



2.1. Frame Difference Implementation

The first algorithm adopted for the IPERMOB car park monitoring system was based on the concept of frame difference. To a great extent, this algorithm computes for each monitored space L, the difference $D_L(t,0)$ between the frame collected at time t, I_t, against a reference frame collected at time 0, I_0, verifying if this difference overpass a given threshold **THR**. Being a significantly high $D_L(t,0)$ likely related to different conditions of the parking space status and by manually characterizing its initial status (for frame I_0), this approach can infer the evolution of the parking space occupancy status. The accuracy of the outcome of this algorithm largely depends on the satisfaction of three conditions:

- *C1:* The reference frame I_0 must be collected when only static objects (trees, sidewalks etc.) are within the FOV of the camera. If this is not the case, the transition from a brightly coloured object to a dark one can be wrongly seen as a status update.
- *C2:* The discrimination threshold (**THR**) that is used for discerning between the empty and busy status, must be correctly defined. Unfortunately, no automatic methods for conducting this task have been identified.
- *C3:* Scene illumination and camera noise, associated to frame I_t are either sufficiently close to the ones associated to frame I_0 or their effects are properly compensated. Variations in these parameters, in fact, reduce the correlation between variations in the frame and changes of the parking space occupancy status.

While *C1* and *C2* have to be addressed by the human operator, *C3* can be generally satisfied feeding the algorithm with other additional pieces of information. Good results can be obtained exploiting the average value, $u_L(t)$ and the variance $v_L(t)$ of the image pixels. The former parameter can be computed for the t-th frames through the formula:

$$u_L(t) = \frac{1}{(W*H)} * \sum_{I=0}^{W} \sum_{J=0}^{H} i_{I,J,t} \tag{1}$$

while the variance of the image can be obtained via:

$$v_L(t) = \frac{1}{(W*H)} * \sum_{I=0}^{W} \sum_{J=0}^{H} (i_{I,J,t} - u_t)^2 \tag{2}$$

For both formulas, $i_{x,y,t}$ is the digitalized value of the pixel at position x,y taken from the t-th frame. The "difference" between the reference frame and the t-th frame is computed using the formula:

$$D_F(t,0) = \frac{1}{(W*H)} * \sum_{I=0}^{W} \sum_{J=0}^{H} \left(\sqrt{v_t} * \left\| u_t - i_{I,J,t} \right\| - \sqrt{v_0} * \left\| u_0 - i_{I,J,0} \right\| \right) \tag{3}$$

The full algorithm for extracting the parking space status can be summarized as:

(1) Store the reference frame that satisfies the condition *C1*. This frame will be referred as frame 0. Compute u_0 and v_0.
(2) Gather the k frame, I_k, compute first $u_L(k)$ and $v_L(k)$ and then $D_L(k,0)$. Store $u_L(k)$ and $v_L(k)$ values.

(3) After having empirically identified a value for the **THR** parameter, identify a candidate value, $\hat{S}(t)$ through the equation:

$$\begin{cases} \hat{S}(t) = 1, & if\ D_L(k,0) > \mathbf{THR} \\ \hat{S}(t) = 0, & otherwise \end{cases} \tag{4}$$

(4) Adopt a low-pass filtering on the values of $\hat{S(t)}$ for removing all the unwanted oscillations. A simple two elements filtering can be adopted for defining $S(t)$, the outcome of the algorithm:

$$\begin{cases} S(t) = \hat{S}(t), & if\ \hat{S}(t) = \hat{S}(t-1) \\ S(t) = S(t-1), & otherwise \end{cases} \tag{5}$$

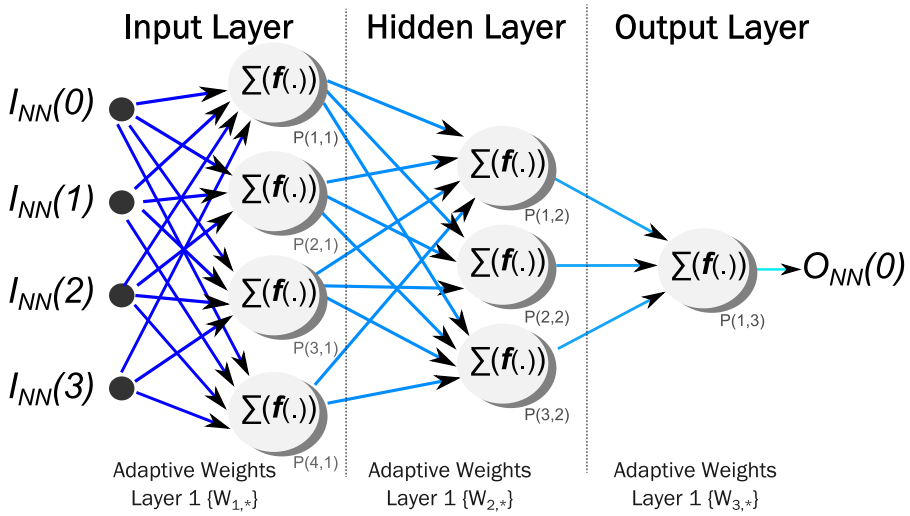
(5) If no background update requests come from the human operator, jump back to step 2, otherwise execute step 1.

This algorithm is characterized by an extremely reduced computational cost and requires only two images I_t and I_0 to be stored into the memory. On the cons side, several limitations make this approach not reliable and potentially tedious for the human operator. *C1* imposes the emptiness of the parking spaces during the phase 1. This condition may be difficult to achieve for parking areas continuously operative. *C2* can be difficult to satisfy, with the human operator forced to frequently adjust the **THR** value. Furthermore, this implementation does not contain any sort of intelligence, and unfortunately all the efforts spent by the human operator do not contribute to an enhancement of the system accuracy on the long run. In order to overcome these limitations, a more robust and powerful approach had to be considered. Machine learning algorithms, thanks to their higher degree of adaptability and to their learning capability, were considered a good solution and deeply investigated. In the proceeding a promising car park monitoring system running on SEEDEYE boards, based on Neural Networks will be presented.

2.2. Supervised Machine Learning Implementation

The main limitation of the previously introduced algorithm is the weakness of the feedback mechanism used by the system. After the initial setup of the parking space statuses, the only automatic correction applied relies on the value of the image luminosity. While this approach can easily tolerate fast transients, it generally fails to convergence to the right solution when slow-varying, local properties change, as for the case of shadows entering the parking space area investigated (the ROI). The system designer can try to cope with environmental changes, only modifying the **THR** selection threshold. Additional parameters could help to better cope with these modifications but they will likely lead to a more problem specific formulation, reducing the future reusability of the solution. These limitations could be overcome using an approach that automatically infers and updates the transformation from the provided inputs (the acquired frames) and the expected outputs (the parking space statuses). Neural Networks (NN) were considered as an effective way to achieve this result and investigated.

FIGURE 2. MLP network structure.



NNs are a computational method that mimics the structure of human brain, describing the main entities, neurons and synapses, in a mathematical formulation. They are used in economy, for predicting market trends, in robotic, for designing adaptive controllers, and in signal processing, for identifying patterns inside a signal (i.e. image or audio). In their initial formulation [**MP43**], NN were considered of limited interest, but they become an extremely valuable tool after important modifications of the original model [**Ros58**, **RHW86**]. Differently from standard computational paradigm, they do not rely on exact algorithms for solving the problem, instead, they, similarly to human beings, learn by examples. Considering the signal processing domain, NN are an extremely powerful tool for identifying patterns. In the image processing domain, neural networks can be used for recognizing specific objects inside an image or a sequence of them. In the ITS scenario, and in particular in the car parking monitoring system, this pattern recognition capability is extremely valuable. Among all the possible implementation of the Neural Networks computational paradigm, due to their reduced computational cost and large usage in literature, Multi Layer Perceptron (MLP) networks were considered the best choice and implemented on the SEEDEYE .

MLPs appear to be broadly used: they are successfully exploited for image recognition [**BCA12**], for image processing [**HC99**] and for voice recognition [**GSR+09**] . MLP networks implement a series of simple mathematical operations (addition, multiplication and thresholding) in a cascaded fashion, through the so-called *perceptrons*. A perceptron, $P_{i,j}$ receives a vector of inputs, which are preliminary scaled through a vector of coefficients (*weights*, W), sums them into a single value that is finally transformed through a not linear function, the activation function, $f()$. For mimicking the brain connectivity, perceptrons are logically grouped in *layers*. The perceptrons composing each layer, $P_{*,j}$, receive the same inputs but their weights vectors, W_i, are independently generated. Each layer generates an output vector that is either passed to the successive layer or outputted as network outcome. Therefore,

a chained approach is implemented with NN inputs, in general, passing through two or three layers before becoming the NN output, as depicted in Figure 2.

The network weights can be automatically computed through different algorithms. Due to its iterative nature this process is referred as network training phase. In it, real inputs I_T are fed into the network through the so-called *input layer* and the outcome, O_{NN}, of the *output layer* is compared against a-priori known outputs, O_T. The I_T and O_T vectors, that together form the training set, can be seen as the input and the output of a black-box that the MLP Neural Network has to mimic. Iteratively, the network will be slightly modified, in order to produce outputs as close as possible to O_T.

To achieve this, for each training step, the initially random network weights and connections are modified for minimizing the so-called network error. This error is represented by the difference between the network outputs O_{NN} and the expected outcomes O_T. Different algorithms with different convergence properties can be exploited, with the Gradient Descent and the Resilient Backpropagation used in the largest part of the cases. For the interested reader, an exhaustive description of the MLP neural networks and training algorithms can be found in Haykin [**Hay08**].

Unfortunately, the number of layers, the amount of perceptrons per layer and the adopted activation functions must be defined by the system designer via an iterative approach. Candidate networks are trained first and then fed with the inputs from the *validation set*, I_V. I_V vectors are equivalent to I_T ones but they must represent minimally correlated set of data. If this correlation is small enough, the validation set can be exploited for verifying how the network responds to never seen before data.

2.2.1. Preprocessing function.

To a great extent, the training of Neural Networks appears to be faster and more effective when the network input vector, I_{NN}, is characterized by a low level of redundancy. Being images extremely redundant in terms of information content, pre-processing functions are largely used for transforming the images into low-redundancy network inputs. For identifying an optimal pre-processing algorithm, various standard image processing algorithms were analysed, investigating, in particular, the amount of computational power and memory required by them. A large number of examined algorithms, for example SIFT [**Low99**] and SURF [**BETVG08**] — while capable of extracting highly valuable features from the image — had to be discarded for excessive requirements in terms of memory/computational power. Fortunately, at least three different parameters can be extracted from the images using the resources available on the SEEDEYE board:

- *The image luminosity*.
  Luminosity can be computed as the mean value and variance of a grey-scale version of the image.
- *The image histogram*.
  To compute the histogram, the image colour space is initially split in **N**, equally spaced, bins. In other words, each bin covers part of the colour space, with the set of them covering the full space. If a pixel value is contained within a specific bin a counter associated to that bin will be incremented by one. As a side note, for grey-scale images, the result can be seen as a

density distribution of the different luminosity levels. The number of bins, **N**, represents the granularity of this density distribution.

- *The edges density*.
  Starting from the Canny algorithm [**Can86**, **Der88**], the edges of all the objects contained in the acquired image can be extracted. This algorithm produces a binary map with "1"s identifying the objects borders. By summing the mask values and dividing by the image size, a sort of edge density can be obtained. Worth noticing that the introduction of any object into the monitored parking space area will likely modify the amount of edges and thus the edge density.

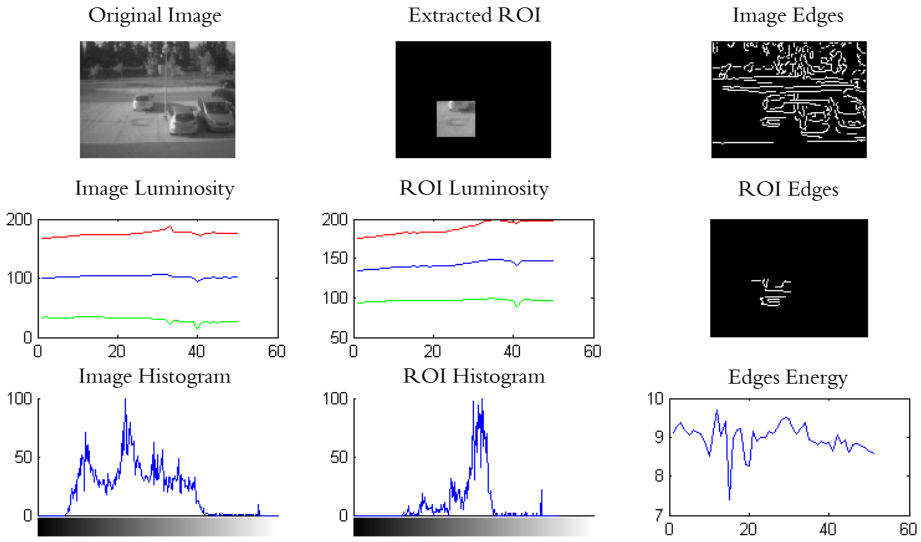
It must be pointed out that the memory and power required for extracting these parameters grow linearly with the number of pixels composing the image. Concerning this point, substantial improvements can be achieved running these algorithms only for the pixel composing the ROIs whenever possible. Additionally, also the input vector size — or equivalently the pre-processing outcome vector size — impacts on the amount of memory and computational power required for training and running the neural network. Being the system in an early design stage, quantitative approaches were difficult to identify and maintain and thus the research of the optimal algorithm was conducted through a qualitative analysis. The candidate algorithms were fed with image datasets in which variations on parking spaces statuses occurred. Their outputs were plotted and, in this way, it was possible to visually evaluate their behaviours in case of empty or busy parking space (Figure 3).

Through this approach it was possible to identify some of the principal advantages and limitations of the three algorithms:

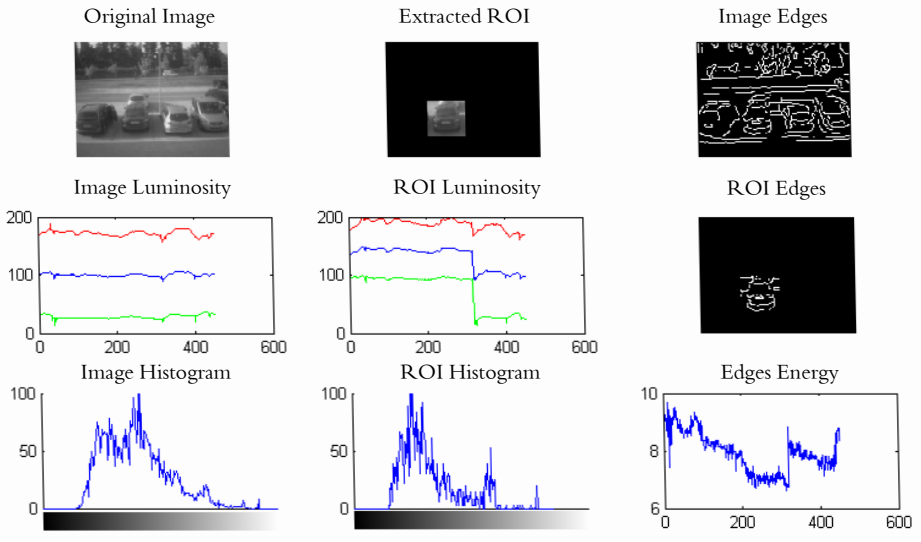
- The luminosity value computed for the complete image has a limited correlation with the status of the parking space. The value computed for the ROI, instead, presents significant variations associated to the space condition (Figure 3 second row, central column). Unfortunately, ambient light variations cannot be easily discerned from objects entering the ROI.
- As for the previous point, the image histogram proves to be a good metric for the status of the parking space but only when computed over the selected ROI. A correlation between histogram distribution and presence of objects inside the ROI is clearly visible. To a great extent, ground appears to be more homogeneous than cars with associated histograms more dense and characterized by lower variance (Figure 3, third row, central column). Compared to luminosity value, histogram looks less affected by light variations but still shadows inside the ROI may be interpreted as cars, leading to wrong results.
- Edge density varies accordingly to the presence or absence of vehicles in the selected ROI (Figure 3, second row, third column). This parameter presents a useful independence from ambient light but it may be less reliable when the ground is not homogeneous and uniform.

From these premises, it results that the image luminosity cannot be broadly applied as NN input for ITS systems (i.e. light intensity cannot be controlled in outdoor scenarios). Both the edge density and the image histogram, instead, present a better tolerance to luminosity variations and thus can be considered as valid pre-processing transformations for the system addressed. Given the low power

FIGURE 3. NN preprocessing selection process. Outputs of the three candidate algorithms when no cars are inside the ROI (a) and when the parking space is busy (b). First row contains the original grey-scale image (320x240 pixels), the ROI subset and the edges extracted from the original image. In the second row, the first two plots from the left contain the trends of the luminosity value computed for the full image and for the ROI, while the plot on the right contains the edges extracted from the ROI. In the third row the image and ROI normalized histograms and edges density trend are plotted.



(a) No cars inside the ROI



(b) Cars inside the ROI

nature of the WSN system targeted, the histogram calculation was preferred for its lower computational cost (approximately one order of magnitude less costly).

2.2.2. Network Structure.

For identifying the optimal network structure, the following approach was adopted:

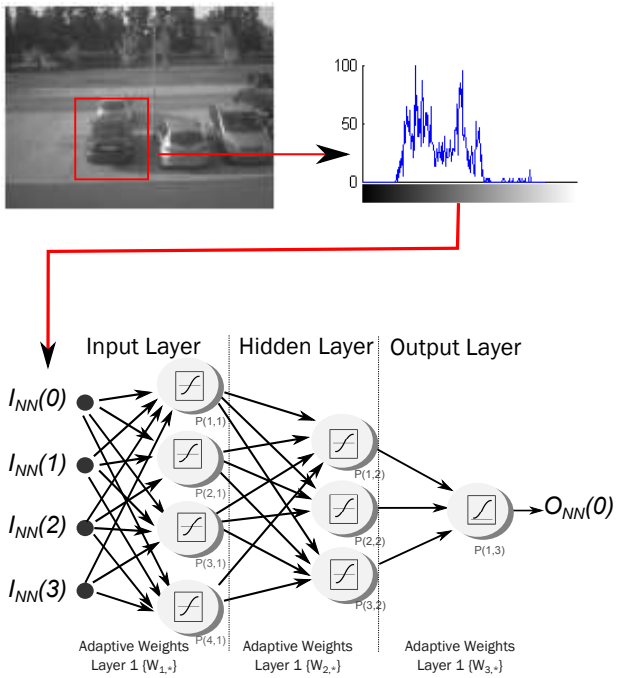
(1) Create a sufficiently wide set of images taken with a single, fixed-position camera in different daytimes with various weather conditions. Long acquisition campaign should provide a sufficiently heterogeneous set. The final dataset will be composed of **NUM_FRAMES** frames $\{F\}$.
(2) Define for all the ROIs $\{R\}$ their coordinates in the first valid frame, F_0. The **NUM_ROIs** variable will contain the number of ROIs defined.
(3) **for** J : 1 to **NUM_FRAMES**-1 **loop**
for K : 1 to **NUM_ROIs loop**
Manually define the status of the ROI (parking space empty/busy)
(4) Split the frame set $\{F\}$ in two subsets: the training set F_T and the validation set F_V.
(5) Create the training set, R_T, as a subset of $\{R\}$.
(6) Associate to the validation set all the remaining ROIs, R_V.
(7) Compute the histogram for the F_T and F_V sets, using the portion of image defined by R_T and R_V, respectively. The outputs of this process will be the vectors of network inputs I_T and I_V, respectively from the training and the validation sets. O_T and O_V vectors will be used for storing the manually defined outputs associated to the two generated input vectors.
(8) Define the network topology, number of layers, activation function as well as the number of perceptrons per layer.
(9) Train the network using I_T and O_T.
(10) Run the trained network using I_V as input.
(11) Compute the network accuracy as:

$$Q = 1 - \frac{1}{\mathbf{N_{FRAMES}}} * \sum_{I=0}^{N} (O_V(I) - NN_O(I)) \tag{6}$$

(12) Reiterate from step 8, up to when all the defined configurations have been tested.

Figure 4 depicts the core logic of the algorithm, providing, as well, a graphical insight of the different network configurations tested. The Mathworks ™ Matlab framework was largely exploited for testing different network configuration and for analysing the quality of their outcomes. The creation and analysis of different Neural Network topologies was largely simplified by means of the Matlab Neural Network Toolbox [**www14l**]. More in detail, the research process was conducted using an acquisition trace composed of 2950 frames (**NUM_FRAMES** = 2950) with 4 ROIs manually defined (**NUM_ROIs** = 4). A small training set was generated extracting 10 frames from the full acquisition set, while the remaining frames were used as validation set. The two R_T and R_V sets uses different, not-overlapping parking spaces, in this way the correlation between the training set and the validation set was minimized. It must be noted that the algorithm will require a specific training for each camera, training that can be easily done during the commissioning phase. Two/three layers structures were

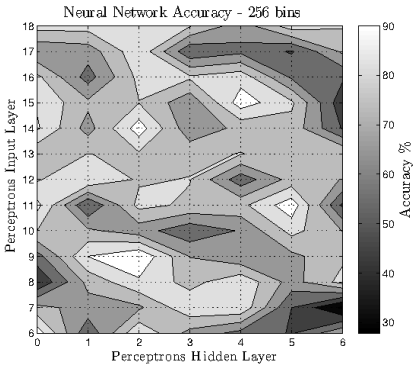
FIGURE 4. The system setup.



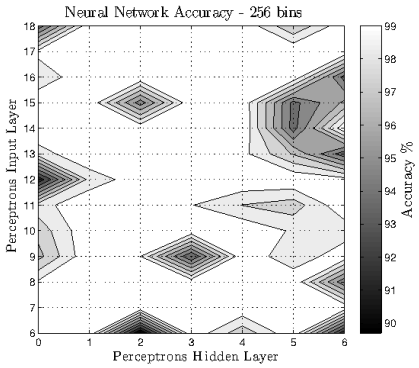
investigated, with **N0** perceptrons used for the layer 0 (the *input layer*) and **N1** ones composing the optional layer 1 (referred in literature as *hidden layer*). The last layer (known as *output layer*) has to produce a single value outcome that represents the status of the parking space analysed, therefore it is always composed of a single perceptron. Another parameter that was varied is the number of bins, **N**, of the computed histogram. As previously anticipated, the cost for running and training the network directly depends on the network input vector size and consequently on the **N** value. With a coherent formulation the input vectors can be defined as $I_{T<N>}$, $I_{V<N>}$ with **N** representing the granularity of the adopted histogram.

Figure 5 depicts the results achieved adopting histogram resolutions of 256, 64 and 16 bins, using the Gradient Descent and the Resilient Backpropagation training algorithms with different network structures. For the Gradient Descent algorithm the initial networks weights were randomly extracted from the set [0, 1] for maximizing the algorithm convergence speed [**GMW81**]. For the Backpropagation method, instead, the approach proposed by Riedmiller [**RB93**] was adopted. The accuracy levels obtained running the networks trained with the Resilient Backpropagation (Figure 5, right column) are significantly higher and with lower variance than with the Gradient Descent algorithm (Figure 5, left column). Concerning the topology, networks appear to be more accurate when the number of neurons is kept limited, with optimal configurations found for values of **N0** in [10-18] and **N1** in [0,4].

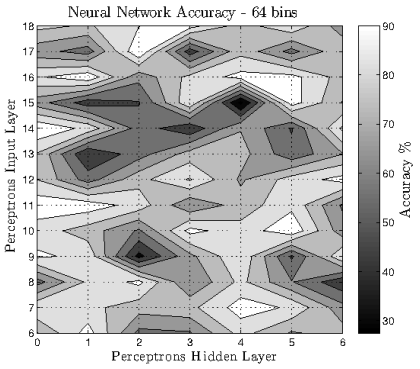
FIGURE 5. Accuracy achieved by the proposed system. Training set inputs are computed on 10 images using different histogram resolutions. Two different training methods, the Gradient Descent and the Resilient Backpropagation are investigated. Figures (a), (c) and (e) show the results achieved using the Gradient Descent algorithm, while (b), (d) and (f) present the accuracy achieved with the Resilient Backpropagation algorithm. Each plot shows the accuracy reached by networks with different combinations of **N0** (ordinate) and **N1** (abscise). Color bars placed on images right side provide a mapping between color and accuracy level reached by the specific network configuration.



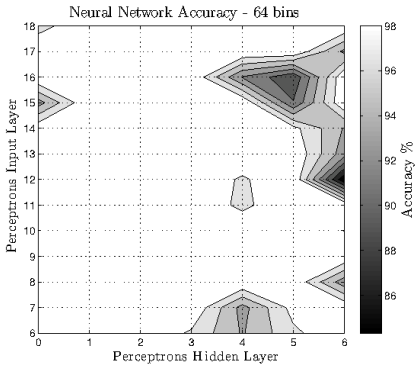
(a) 256 bins histogram, GD



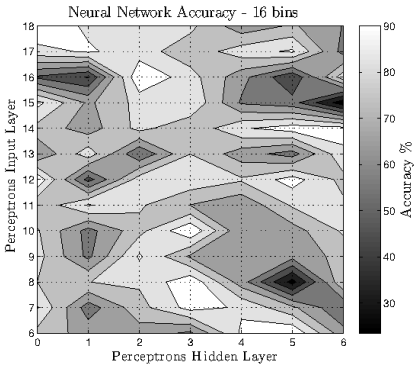
(b) 256 bins histogram, RP



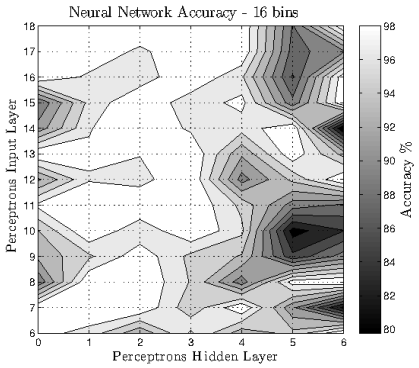
(c) 64 bins histogram, GD



(d) 64 bins histogram, RP



(e) 16 bins histogram, GD



(f) 16 bins histogram, RP

2.2.3. Porting on the SEEDEYE.

The identified system had to be implemented balancing performance and configurability in order to be effective as well as extendible. The histogram computation routine and the set of neural network functions (i.e. the ones for generating the network, for training and for running it) were developed maximizing both the aspects.

For the neural network, the design process leaded to a MLP Neural Network library optimized for low-power microcontrollers. The network topology could be configured at *compile-time* or at *run-time*, allowing a better utilization of memory resources and a faster initialization, in the first case, or more adaptability using the latter flavour. In particular, the *run-time* initialization simplifies the assignment of the network weights, permitting "over the air" update of them. The NN weights can be computed by a standard elaborator and then transmitted to the SEEDEYE nodes through the IEEE802.15.4 interface, relying on a Gateway node. The benefits of this approach are:

- Execution times are significantly reduced and errors can be more easily spotted;
- Simplified GUI can used for the training process, making the network configuration a painless process;
- Various training functions, eventually too complex for embedded devices, can be compared and tested on the real system.

TABLE 1. Histogram execution time for different resolutions and ROI areas. Measurement error < 0.01 ms.

	128 bins	**96 bins**	**64 bins**	**48 bins**	**32 bins**
160x120	91.55 ms	90.35 ms	89.85 ms	89.08 ms	89.01 ms
100x100	50.06 ms	49.12 ms	48.43 ms	47.87 ms	47.35 ms
75x75	29.67 ms	28.78 ms	28.04 ms	27.57 ms	27.22 ms
50x50	15.08 ms	14.25 ms	13.47 ms	13.05 ms	12.67 ms
35x35	9.06 ms	8.28 ms	7.50 ms	7.10 ms	6.72 ms

From these premises, more effort was spent on optimizing the network execution instead of speeding up the network training. The execution time of the two main processing blocks is listed for different configurations in Table 1 for the histogram computation routine and in Table 2 for the NN output computation.

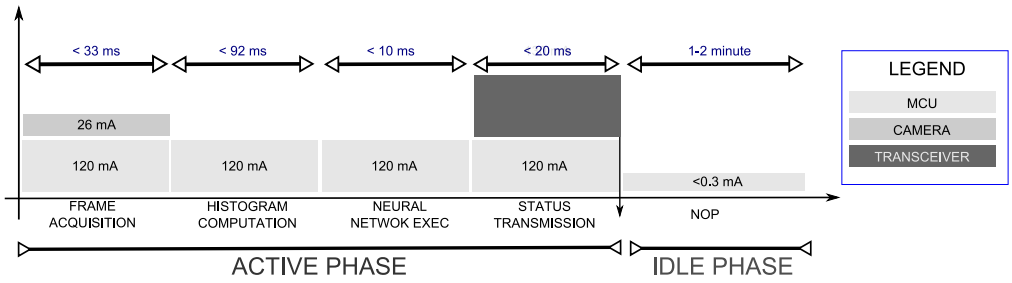
Considering, as example, a square ROI of 75 by 75 pixels and a histogram granularity of 96 bins, the SEEDEYE will produce the desired network outcome in 3.86 ms + 28.78 ms = 32.62 ms. Knowing that a new update requests from the Control Room should arrive every minute (i.e. 60*1000 msec), the refresh rate results being $60000/32.62 = 1840$ times longer than the time required for computing the new values. In other words, the CPU has an utilization factor of less than 0.1%.

TABLE 2. Network execution time for various network structures. Tansig is used as activation function in the input and in the hidden layers, while logsig is adopted in the output layer. Measurement error < 0.01 ms.

Input size	N_0	N_1	N_2	Exec. Time
96	18	4	1	3.86 ms
96	18	0	1	2.87 ms
64	18	4	1	2.86 ms
64	18	0	1	2.59 ms
48	18	4	1	2.41 ms
48	18	0	1	2.15 ms
32	18	4	1	1.83 ms
32	18	0	1	1.57 ms

As an important result, the system consumption can be largely reduced bringing the PIC32 MCU in a low consumption state (Sleep Mode) during the inactive phase. Figure 6 well depicts this scenario, giving a rough estimation of the power consumption during the different phases.

FIGURE 6. Processing structure with associated power consumption.



CHAPTER 3

SEEDEYE for streaming applications

AFTER the positive results achieved within the IPERMOB project, other multimedia related application started to be deployed on top of the SEEDEYE board. A series of them, addressed the streaming of audio or video contents in WSNs, a research field that is considered extremely promising.

Audio and video streaming applications can provide extensive benefits in emergency scenarios, providing useful information regarding survivors' location in damaged buildings or in collapsed caves. In addition, they can simply be used for monitoring wild life in forests or for surveillance applications. Unfortunately, the adoption of these technologies is limited by the lack of low complexity compression algorithms, without them, in fact, the limited bandwidth available on WSNs does not suffice for effectively streaming multimedia signals. Compression algorithms, in order to operate on WSNs, must compress the signal stream bits using the minimal amount of power as possible.

In the proceeding, some of the existing works addressing this challenging topic will be introduced, focusing on the ones designed on top of the SEEDEYE board. For the sake of clarity, video and audio streaming will be described separately.

3.1. Video streaming

Considering the video streaming, it must be pointed out that compression schemes used in normal elaborators, such as the H264 or MPEG4 codecs [**HCS07**], come with a computational cost that current WSN nodes cannot stand. Fortunately, the computational power required for implementing the video streaming can be reduced in different ways. First, the usage of fixed point representation of the data, instead of the commonly used floating point one, can determine a tenfold reduction of the execution time. An interesting work [**MRX08**] addresses the widespread JPEG2000 standard, focusing, in particular, on the power consumption reduction achievable moving from the initial floating representation to a slightly less accurate fixed point one. Other researches, instead, investigate less computational costly implementations of existing compression schemes. To this category belongs the analysis conducted by Paniga et al. [**PBR+11**], where a hybrid DPCM/DCT coding scheme is proposed. The scheme resembles the JPEG encoding scheme but it is characterized by a significantly lower computational cost.

To a great extent these approaches exploits only intra-frame information for reducing the image size, ignoring the correlation existing between contiguous frames. A complimentary approach can be

adopted whenever the difference between two consecutive frames is limited for a significant large number of frames. In this scenario, the bandwidth requirement can be significantly reduced by transmitting only the variations instead of the full frame.

This approach is largely exploited by Salvadori et al. [**SPP+12**] that propose a video streaming application that achieves significant compression ratios relying on a change-detection approach. Their algorithm analyses the inter-frames correlation in order to identify image portions that must be updated (i.e. to be transmitted), considering the remaining part of the image as unvaried (i.e. not to be transmitted). From a closer perspective, the algorithms can be decomposed in four steps:

(1) *Image extraction*.
The extracted image is stored in memory as *foreground frame*. Accordingly to the value of an internal variable, known as *refresh counter*, the system will proceed to step 2 or 3. The *refresh counter* is initialized with a value **N** that represents the inter-distance, in terms of number of frames, between two full image transmissions. This update is used to compensate slow-dynamic scene variations, such as luminosity variations. When the *refresh counter*, that is decremented for every frame collected, becomes zero, the algorithm executes the step 2, otherwise it jumps to step 3.

(2) *Full transmission*.
A background refresh operation has been required and two operations are performed. Locally, on the WSN node, the *foreground frame* replaces the so-called *background frame*. This latter frame represents the reference value that will be used for compressing the video stream, as described in steps 3. Additionally, the WSN node transmits the new value of the *background frame*, together with the image size, to the receiver, where these elements will be used for the stream decompression, as for step 4.

(3) *Partial transmission, blob identification*.
When the refresh operation is not required, the subset of the image that has changed is identified. This area is inferred by differencing the *background frame* against the *foreground frame* and by thresholding the generated values. This process produces a binary mask that explicitly shows the group of pixels that has changed, allowing the identification of an enclosing rectangular *bounding box* defined by the four parameters, **H**, **W**, **x** and **y**. The first two represent the size of the rectangle, while the last two describe its position relative to the top left corner of the frame. The subset of the *foreground frame* contained in the *bounding box*, indicated by the authors as *blob*, can be finally sent to the other node together with the four afore-mentioned parameters.

(4) *Decompression*.
On the receiver side the full frame (or a subset of it) is reconstructed exploiting the **H**,**W**, **x** and **y** parameters. It must be pointed out, that for the full frame the values of **x** and **y** have been set to (0,0) while **H** and **W** represent the height and width of the image, respectively. The receiver can identify a *background frame* update, simply using a local version of the *refresh counter* or by evaluating these conditions:

$a)\quad x = y = 0$

$b)\quad (W = \textbf{image_width})\, and\, (H = \textbf{image_height})$

If the counter reaches zero or both *a*) and *b*) are satisfied, the system will update the stored version of the *background frame*, reset the *refresh counter* and output the new background frame as the *decompressed frame*. Otherwise, the algorithm will generate the decompressed frame substituting the bounding box portion of the *background frame* with the received *blob*. The stored *background frame* will be not modified in this case and refresh counter will be decremented.

The authors evaluate the efficiency of the algorithm against two different validation sets. Both of them were collected in acquisitions campaigns conducted during the IPERMOB activity, with the first one collected in a parking area and the second one acquired installing the camera on an urban road side. Considering their amount of variations, the former trace can be considered a *low motion scenario*, while the latter trace — that is characterized by higher variability — is addressed as *high motion scenario*. The algorithm has been tested using raw images but it could have been applied on top of traditional compression schemes, such as JPEG, with no significant variations. Due to the thresholding operation applied in step 3, the algorithm belongs to the category of lossy compression schemes. The degradation of the reconstructed image J as a function of the initial image (I) quality, can be computed using the peak signal-to-noise ratio (**PSNR**(I,J)) value, defined as:

$$\mathbf{PSNR}(I,J) = 10\log(MAX(I)^2/\mathbf{MSE}(I,J)) \tag{7}$$

where $MAX(I)$ is the highest pixel value of the original image and the mean square error (**MSE**(I,J)) is defined as:

$$\mathbf{MSE}(I,J) = \frac{1}{(W*H)} * \sum_{i=0}^{H-1}\sum_{j=0}^{W-1}[I(i,j)-K(i,j)]^2 \tag{8}$$

Table 1 and Table 2 depict the results achieved varying the values of **N** for the low motion and high motion validation traces. Similar **PSNR** values have been achieved using far more computational costly compression schemes as by Paniga et al. [**PBR+11**]. In must be noted that the parameter **N** provides an easy way to trade video quality for bandwidth, with negligible impacts on the total system computational cost. The performance of this approach can be improved substituting the fixed *refresh period* with an adaptive one. In this case, the **MSE**(I,J) between the foreground frame, J, and the background frame, I, — computed on the transmitter side — can be used for triggering the refresh operation, using a simple condition, such as:

$$\mathbf{MSE}(I,J) >= \mathbf{T} \qquad \textbf{refresh}$$
$$\mathbf{MSE}(I,J) < \mathbf{T} \qquad \textbf{wait}$$

where **T** is an application specific constant. Compared to the fixed refresh period approach, the adaptive approach can potentially reduce the bandwidth requirement while preserving or even enhancing the video quality.

The bandwidth reduction achieved via the presented system can be exploited for two not mutually exclusive finalities: first for transmitting more frames per seconds, second for implementing data protection mechanism, such as forward error correction (FEC) techniques. FEC is an extremely valuable tool that can be used for coping with the error-prone nature of the wireless channel. It must be noted

TABLE 1. Low Motion Scenario. Relation between refresh period **N**, bandwidth occupancy and reconstructed image **PSNR**.

N	**Bandwidth Occ. [kbps]**	**PSNR**
1	153.60	inf
10	19.63	38.05
20	13.02	37.42
30	10.02	36.74
40	9.68	36.60
50	9.33	36.11
60	9.23	35.84
70	9.01	35.92
80	9.02	35.84
90	9.06	35.36

TABLE 2. High Motion Scenario. Relation between refresh period **N**, bandwidth occupancy and reconstructed image **PSNR**.

N	**Bandwidth Occ. [kbps]**	**PSNR** [dB]
1	153.60	inf
10	51.02	30.93
20	48.86	30.67
30	48.02	29.80
40	51.84	29.70
50	47.67	29.72
60	50.54	29.67
70	56.53	29.68
80	55.53	29.32
90	56.06	29.22

that in real applications, the quality of the video stream depends also on the number of uncorrected errors introduced by the wireless transmission.

FEC schemes introduce properly chosen redundant bits to correct channel errors (i.e. bit inversions of the original trace). The number of errors that can be corrected is related to the number of redundant bits introduced. A large number of FEC schemes are described in literature [**Moo05**] and each of them comes with a different correction capability and computational cost. Higher correction

capability means that the same number of redundancy bits can be used for correcting more errors. Unfortunately, the low power, rare memory nature of WSN nodes, limits the number of available schemes.

Targeting the SEEDEYE board, Petracca et al. [**PGS+11b**] propose a low complexity FEC scheme based upon the class of error correcting codes referred as BCH codes [**BRC60**].

In the literature, BCH codes are defined by four important parameters:

- n, that represents the number of bits generated by the coding process;
- k, the number of bits that enters the BCH coder, or in other words, the number of bits to protect;
- t, the number of bits that can be corrected, aka the correction capability of the code.
- $R_c = k/n$, the code rate, that represents the efficiency of the code adopted.

The authors analyse different BCH codes (characterized by various n, k and t combinations) researching the optimal trade-off between transmission overhead and error correction capability. In their study, a video stream is stripped in k-bits fragments, coded using a $BCH(n,k,t)$ code and finally embedded in the data payload of IEEE802.15.4 packets. The multimedia stream is composed of a series of frames collected within the IPERMOB project in a relatively dynamic environment. On the receiver side two possibilities arise, if the packet header has been corrupted by errors, the packet is dropped and must be concealed, otherwise the payload is decoded and the original fragment of length k is decoded back. The afore-mentioned concealment process implies the filling of the gap left behind by the lost packet. In their studies the authors show that the copy-concealment represents an effective way for coping with lost fragments. As the name suggests, copy-concealment technique replace the lost fragment with the equivalent frame (in terms of position on the image) associated to the previous reconstructed frame, exploiting the time locality of video sequences.

TABLE 3. Reconstructed image quality versus bandwidth overhead penalty. The wireless channel has an average **BER** of $1.69 * 10^{-3}$.

FEC adopted	R_c	**Overhead [%]**	**Err. Rec. [%]**	**PSNR [dB]**
None	1	0	0	35.88
$BCH(255,247,1)$	0.968	19.31	3.54	36.62
$BCH(255,239,2)$	0.937	23.14	20.38	36.73
$BCH(255,231,3)$	0.906	27.36	20.38	36.73
$BCH(127,120,1)$	0.944	15.30	4.67	36.67
$BCH(127,113,2)$	0.889	22.51	24.82	37.83
$BCH(127,106,3)$	0.834	30.61	50.50	39.34
$BCH(63,57,1)$	0.904	17.94	5.71	36.77
$BCH(63,51,2)$	0.809	31.90	27.29	38.34
$BCH(63,45,3)$	0.714	40.74	53.62	39.60

The analysis of the performance of the system, instead of relying on a model of the channel, makes use of real traces collected in the IPERMOB test-bed. These traces have been generated by transmitting known patterns from a *sender* board to a *receiver* board. The latter, knowing the expected sequence, could easily infer all the bit errors introduced by the wireless channel, estimating as well the bit error rate (**BER**) of the channel. The **PSNR** of the reconstructed image has been computed as in equation 7 and the results achieved for the different BCH code variations are listed in Table 3. With a relatively small overhead of 20% - 30%, a significant number of errors can be corrected with this approach, significantly enhancing the quality of the received stream.

An implementation of this algorithm on top of a SEEDEYE board have been investigated and the computational cost required is well within the capabilities of the device. From the analysis conducted, this approach could be effectively used to enhance the streaming quality, even in extremely noisy environments.

3.2. Audio streaming

Audio streaming on MWSN does not differ significantly from video streaming. The same constraint on computational resources exists, bandwidth remains limited and the communication channel introduces errors that degrade the final quality of the stream. However, compared to video streaming, a larger number of studies has been addressing low-complexity implementations of compression/decompression schemes. Cellular phones, portable music players and Voice over IP (VoIP) services have fostered the researches on this topic. From the former group, a significant number of compression/decompression algorithms could be considered "WSN-friendly".

It must pointed out that the selection of the optimal algorithm must be driven by its computational cost as well as by the targeted reconstructed audio quality. Concerning the latter aspect, for the voice transmission, a largely adopted quality metric is the Mean Opinion Score (MOS) [**www96**]. MOS is computed in a qualitative way, through definition of a quality scale (0,5) with 5 representing the highest quality possible. Reconstructed signal is graded by a sufficiently large set of human listeners

TABLE 4. List of the commonly used compression/decompression algorithms for voice transmission. The acronym WMOPS stands for Weighted Million Operations Per Seconds. For the wideband codecs, unique values for the Mean Opinion Score (MOS) have been found.

Coding Scheme	Freq. Range	Coding rate [kbps]	RAM [Kword]	Complexity	MOS
ITU-T G.711	Narrowband	64	0.01	0.35 MIPS	4.41
ITU-T G.726	Narrowband	16/24/ 32/40	0.15	12 MIPS	2.22, 3.51, 4.24, 4.37
ITU-T G.728	Narrowband	16	20.38	36 MIPS	4.24
ITU-T G.722	Wideband	48/56/64	1	10 MIPS	-
ITU-T G.722.1	Wideband	24/32	5.5	10 WMOPS	-

FIGURE 1. Structure of the IEEE802.15.4 packet adopted.



and the average of their grading represents the MOS value. To clarify the concept, Table 4 lists, some ITU standards [**www09**, **www14j**] for voice encoding — and their associated MOS values —. It is worth noticing that the limited computational power required by these approaches is compatible by the SEEDEYE device.

Even though the complexity of some of the algorithms listed in Table 4 is compatible with the WSN available resources, only few researches have been considering the adoption of these well-established algorithms in their systems. Among the group of compression-free implementation, Mangharam et al. [**MRRS06**] present a monitoring system for a coal-mine that supports bidirectional audio streams. Due to the high requirements in terms of bandwidth, sensors can transmit the collected audio in a scheduled, TDMA-based, fashion. Other similar works are from Li et al. [**LXSL09**] and Petracca et al. [**PLR+09**] where two different audio quality enhancement techniques are investigated.

A more bandwidth-aware approach has been investigated in [**PGPP12**], where a low complexity compression algorithm has been combined to a filtering operation for minimizing the bandwidth requirement and to a data protection technique for enhancing the transmission quality.

More in detail, the compression algorithm adopted is the low-cost ITU-T G.711 codec, for which a free of charge C implementation is provided by ITU. G.711 relies on logarithmic transformation for reducing the bandwidth of the audio signal down to 64 kbps using a negligible amount of computational power. In their proposal, Petracca et al, to further decrease the required bandwidth, perform a filtering operation on the signal, transmitting only the valuable pieces of information. With the audio stream firstly split in *audio packets* of 10 ms duration and then further divided in $\mathbf{N_{Seg}}$ *audio segments*, the decision of what must be transmitted is taken by estimating the contribution of the *audio segments* to the reconstructed signal. An effective way for establishing the contribution of an *audio segments* is represented by its energy, E_i that can be computed through the formula:

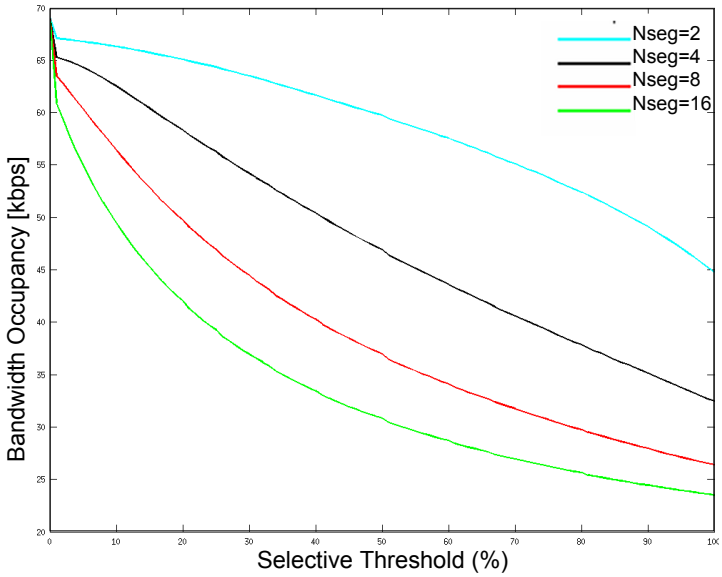
$$E_i = \sum_{j=1}^{M} (x_j^2) \tag{9}$$

with $\{x\}$ the set of M discrete samples composing the audio segment. After having identified E_{max}, the maximum value for E_i within the packet, only the packets respecting the following condition will be transmitted:

$$100 * (E_i / E_{max}) >= \mathbf{S_{tr}} \tag{10}$$

The selection threshold, $\mathbf{S_{tr}}$, can be used for trading bandwidth for signal quality, making the implementation more adaptive to changes. Selected segments are encoded using the G-711 algorithm and packed for transmission. The final packet (i.e. the IEEE802.15.4 payload) will contain a header composed of two fields, **C_mask** and **P_mask**, as well as the set of compressed fragments. The so-called

FIGURE 2. Required bandwidth vs Selective Threshold values for different $\mathbf{N_{seg}}$.



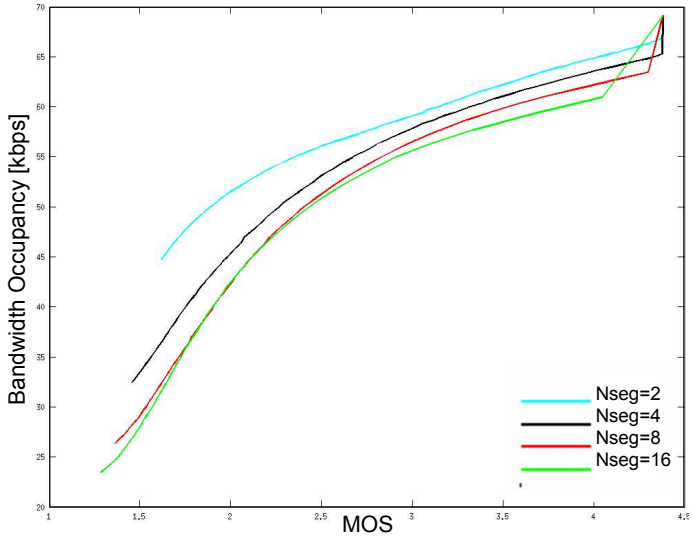
current mask, C_mask, provides a mapping between the encoded fragments, F_I, and their final position within the reconstructed audio packet. The reconstruction process will read this 2 byte field for determining the position of the decoded fragments within the packet. Fragments that were discarded due to their irrelevance are concealed, through a zero-concealment technique. In this approach, zeros replace the not transmitted segments. The **P_mask** contains similar pieces of information but it has been introduced for increasing the system tolerance to packet losses. Whenever a generic packet I is lost, the **P_mask** can be used for reconstructing part of its content. To achieve this result, the IEEE.802.15.4 has been structured as in Figure 1.

After the current encoded fragments F_{I+1}, if the available payload of 100 bytes has not been fully utilized, the F_I most important fragments belonging to the previous audio packet are encoded and embedded into the packet. The P_{mask} and the F_I segments contained in the I+1-th packet will be used to reconstruct the content of the I-th audio packet. Even with a single fragment recovered from the lost packet, being this fragment the most important one, the quality of the reconstructed signal can be highly enhanced. The negligible drawback of this approach is represented by the introduction of a fixed delay of one audio packet, i.e. 10 ms.

The implementation is tested in two different scenarios feeding the system with speech traces belonging to the NTT Multi-lingual Speech Database [**www14m**]. In the first set of tests, the channel is considered error-free and the average bandwidth required for streaming the signal is shown in Figure 2 where different combinations of $\mathbf{S_{tr}}$ and $\mathbf{N_{seg}}$ are considered. Concerning the quality of the reconstructed signal, Figure 3 shows the MOS value as a function of the average bandwidth.

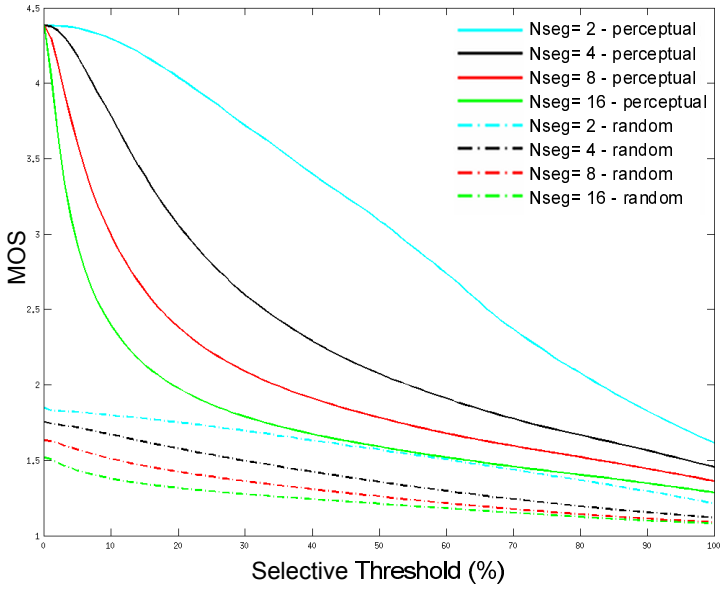
A more realistic scenario, considers instead the error-prone nature of the channel, investigating the quality of the protection scheme adopted. Packets are randomly dropped as it occurs in a real scenario when errors affect the header of the received packet. The amount of packets to be dropped has been

FIGURE 3. Mean Opinion Score value vs required bandwidth for different $\mathbf{N_{seg}}$.



set equal to the Packet Loss Rate (PLR) measured using a transmitter power of 0dBm with the receiver placed at 10 mt of distance in an indoor environment. If no protection mechanism is put in place, the final PLR of 5.5% significantly affects the MOS value, dropping the MOS value from 4.38 (error-free channel) to 3.03 (error-prone channel). When instead the protection mechanism is adopted the quality

FIGURE 4. MOS vs selective threshold for different $\mathbf{N_{seg}}$.



of the reconstructed signal increases. Figure 4 shows the MOS enhancement that can be achieved using this protection scheme.

The combination of protection scheme and signal filtering provide excellent results in terms of bandwidth reduction and reconstructed signal quality, implementing an easy mechanism for trading bandwidth for quality. When lower audio quality is tolerated, $\mathbf{S_{tr}}$ can be set to higher values, significantly reducing the bandwidth request. When instead, higher quality is required the protection scheme will contribute to compensate for the channel imperfection, enhancing the final perceived quality.

Part 2

Throughput constrained systems

CHAPTER 4

A DSP for Ultra-Fast Optical Communications

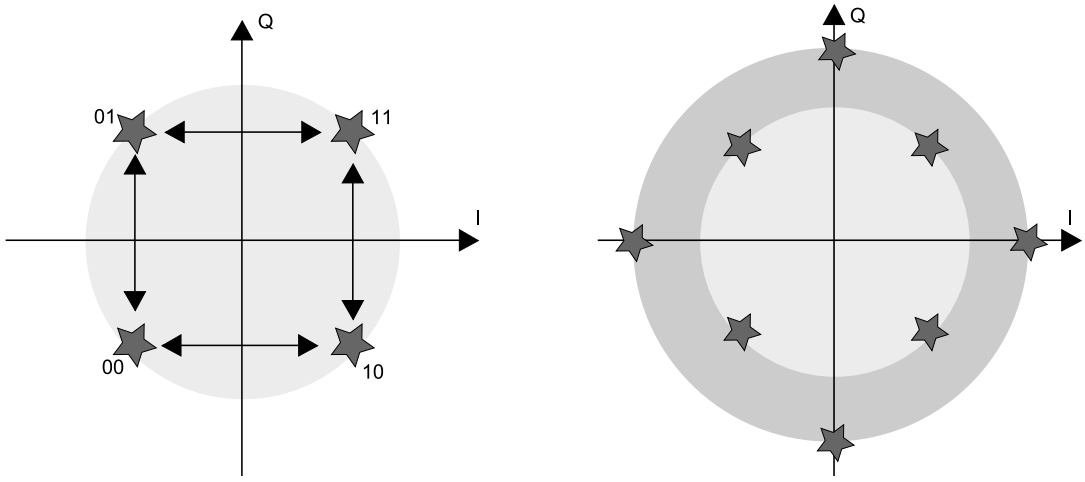
NOWADAYS telecommunication networks mainly rely on optical fibres for transferring data within the continents and among them. Roughly 70 Exabyte of global IP traffic is generated monthly and reasonable forecasts set this value to 120 Exabyte for the year 2017 [**www14i**]. This continuous request of more bandwidth has fostered the research on optical communication, leading to an impressive reduction of the cost required for transmitting a single bit.

Interfaces for the 100 Gigabit Ethernet (100G) have been standardized [**www10**] and commercial products start to be available on the market [**www14h**]. At the present time, the IEEE 802.3 working group is defining various physical implementations of the 100G standard, exploiting optical fibres as well as copper cables. Concerning the former transmission medium, the standard proposes 100G solutions based on four optical links operating at 25.78125 Gbps. Data stream is encoded with a 64b66b scheme [**st600**], producing the targeted 100 Gbps bit-rate. While minor modifications have been made to the MAC layer, this extension of the standard will introduce significant novelties on the network physical layer (PHY). From the latest draft, all the 100G optical-based flavours of the new IEEE802.3 standard will make use of single-channel Polarization Division Multiplexed Quadrature Phase Shift Keying (PDM-QPSK) techniques [**www01**, **CZJ11**]. This modulation scheme represents an excellent step towards even faster interfaces, like the 400G and the 1T (1 Terabit/s).

In fact, a foreseen approach for augmenting the channel bandwidth relies on higher-order modulation formats. The QPSK modulation used in the PDM-QPSK communication scheme has order 4, i.e. encodes the stream with a constellation of 4 points (Figure 1.(a)). Unfortunately, when adopting PSK modulation, the maximum order that can be realistically exploited is 16: more dense constellations, in fact, are affected by errors too difficult to compensate. This limitation has been overcome by adapting solutions already used in the wireless communications domain. In particular, excellent results have been achieved moving from the PSK modulation the to Quadrature Amplitude Modulation, QAM [**BB99**]. **N**-QAM (Figure 1.(b)) modulation of high-order (i.e. 128) provides in fact sufficient noise margin to allow ultra-fast communications [**Nak09**]. With these higher order modulations, for a given bit-rate, the spectral efficiency — i.e. the maximum throughput divided by the bandwidth in hertz of the channel — increases and the symbol rate decreases, leading to a reduction of the electronic processing frequency.

By exploiting a coherent detection scheme [**www12c**], the adoption of these high-order modulation formats is simplified and furthermore, the receiver does not introduce non-linearities that could impair the quality of the received signal. The use of coherent detection techniques, combined with advanced Digital Signal Processing (DSP) algorithms is considered the most efficient solution towards the development of future architectures operating at 400 Gbps (400G) and beyond. QPSK-based solutions

FIGURE 1. Standard Modulation Scheme. A Quadrature PSK (QPSK) is depicted in (a) while (b) shows a 8-QAM. It is worth noticing that the constellation associated to a QPSK and to a 4-QAM — graphically represented by the inner circle of (b) — are equivalent.



(a) Example of QPSK constellation

(b) Example of 8-QAM constellation

for the 100G can be upgraded for 400G links adopting a four times higher modulation scheme, such as the 16QAM. While the symbol rate will remain stable, the DSP logic will increase in complexity in order to cope with the lower SNR tolerance. In fact, higher order modulation schemes come with a reduced SNR — due to the diminished distance among constellation points — and thus a more difficult and error-prone reconstruction process. Independently from the modulation scheme adopted, DSP logic must compensate for channel distortion and detrimental effects, such as the group-velocity dispersion, GVD [**www14b**], and polarization mode dispersion, PMD [**GM05**].

With the current technology, due to the extremely high symbol rate of the transmission, all the DSP operations must be performed in a parallel fashion for providing real-time results. The most effective way for achieving these results exploits FPGAs as core processing unit: no other technology can provide comparable parallelization and configuration capabilities. Unfortunately, the design of complex DSP algorithms on FPGAs requires a significant amount of time, with several limitations on the debugging process.

To overcome these limitations, a model-based design approach can be exploited. In traditional designs, a waterfall approach is generally used. In this approach, after the requirements definition phase, designers start with the actual implementation of the system, alternating design and debug phases up to the final solution. In model-based designs, instead, the requirements are the starting point for the creation of an abstract model of the system. This model will be continuously improved and when considered sufficiently exhaustive and accurate, tools will be used for converting it into a real implementation of the system. Considering the design of DSP algorithms for FPGA as an example, the former approach will mainly focus on the coding of HDL components and low-level test-benches, while the

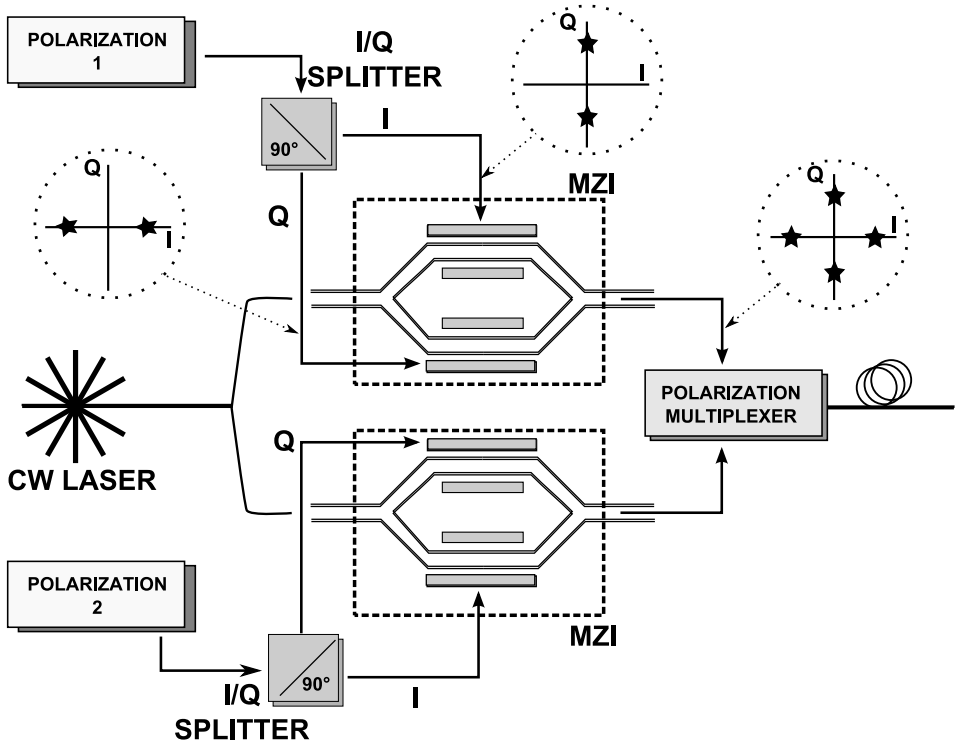
latter will focus on the creation of an abstracted model that describes the DSP algorithms. Automatic code generation tools will then be used for generating the targeted software/firmware implementation of the system. It is worth noticing that can be configured for addressing specific programming languages and architectures fostering code portability (modifications to the model are not necessary) and debug effectiveness (high level models are generally easier to the debug). On the cons side, some of these tools are still far from providing software/firmware as good as the human written one. When the system requirements are not completely defined and significant modifications/upgrades of the system are foreseen, the not-optimality of the code will be largely compensated by the shortening of the design process duration.

In the proceeding, the design of a FPGA-based DPM-QPSK receiver for 100G communications will be presented, focusing in the initial part on the DSP components. The model-based approach adopted will be discussed later on, presenting in the final part the solutions deployed and the results achieved. Being the design of an effective DSP systems the focus of this Part, the structure of the optical link (transmitter and receiver properties, optical channel characterization) will be introduced in a not-exhaustive way. For the interested reader more details can be found in [**Tay04**, **Kaz89**].

4.1. DSP implementation of an Ultra-Fast Optical Receiver

At the present days, few works are addressing single fiber transmissions with data-rates above 100G. Among them, the largest part is still in the design stage, with systems far from being concrete.

FIGURE 2. Transmitter structure for coherent detection.



The Ultra-Fast all-optical group group from the TECIP institute, starting from the work conducted by Colavolpe et al. [**CFFP09**] produced one of the first specimen of single fiber 100G+ optical link. In their system, two independent data streams are sent on two orthogonal States Of Polarizations, SOP [**www12b**], and reconstructed via a coherent detection scheme. By adopting QPSK (initially) and QAM modulation (later on), the system has achieved an extremely high transmission spectral efficiency, representing a fundamental step towards 400G+ optical transmissions. To be broadly adopted, this system required the design of an FPGA-based DSP core. After a brief description of the full optical transmitter/receiver, a multi-FPGA implementation of the receiver DSP logic will be presented.

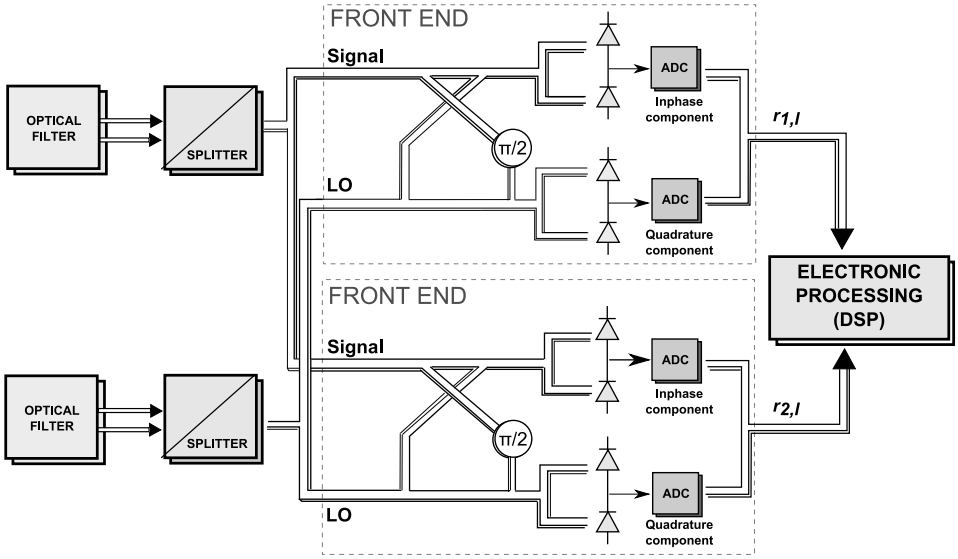
4.1.1. The structure of the transmitter.

In the analysed system, two independent streams of data are initially modulated and later on transmitted through the two different optical polarizations of the fibre. More in detail, for each polarization, a Mach-Zender Interferometer (MZI) is used for imposing a QPSK modulation to the data streams. The MZI [**www12a**] modulates the In-Phase (I) and Quadrature (Q) components of the incoming signals using a Differential Phase-Shift Keying (DPSK) modulation. The I/Q components are merged into a Quadrature Phase-Shift Keying (QPSK) representation by an optical coupler as depicted in Figure 2. MZIs modulates a continuous wave laser — i.e. a laser operated at fixed voltage — operating as external modulators [**Agr10**]. Finally the two streams (now modulated) are merged by an optical Polarization Beam Combiner (PBC) and transmitted along the fibre.

4.1.2. The structure of the receiver.

The receiver, depicted in Figure 3, comprises two main components:

FIGURE 3. Receiver structure for coherent detection.



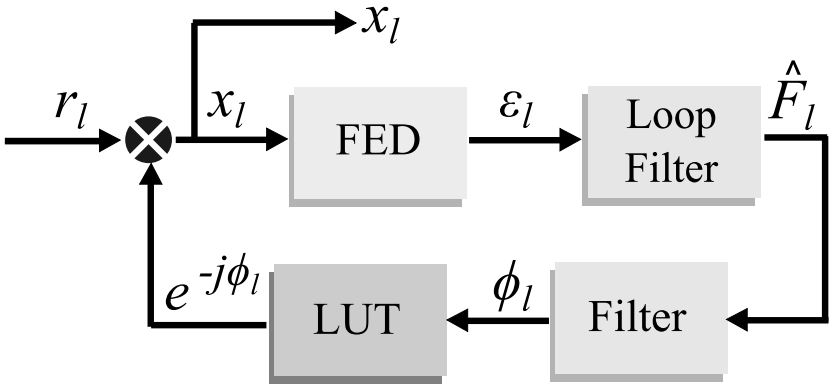
- *The Analog Front End.*
  An opto-electronic (O/E) front end performs the initial demodulation of the signal and the successive optical to electrical conversion. The signal is initially filtered when it is still in the optical domain, then the two orthogonal optical polarizations are split through a Polarization Beam Splitter (PBS). They are successively combined with the optical field of a local oscillator laser (LO) in 2x4 90° copies and detected with four (two by two) photo-detectors. With the assumption that only a relatively low frequency offset exists between the incoming signal and the LO laser, a "cheap" free-running LO laser can be used at the receiver. The fine frequency recovery that then results necessary can be performed by the digital logic, with an Automatic Frequency Compensation block (AFC). At this step of the elaboration, the signal is in the electrical domain and it can be digitized by means of Analog-to-Digital Converters (ADCs).
- *The Digital Back End.*
  The two sets of In-Phase (I) and Quadrature (Q) components outputted by the ADCs are then passed to the digital part where they are combined as a two-dimensional column vector r_l. Whenever the sampling rate satisfies the Nyquist condition, the received signal can be reconstructed correctly. In this scenario, b samples per symbol interval T will be extracted from the signal, resulting in a sampling interval of $T_c = \frac{T}{b}$. The digitalized vector r_l can be used for reconstructing the original transmitted bit stream after proper compensation of the channel effects has taken place. Two main blocks compose this digital signal processing block: the first contains the frequency estimation and compensation features (AFC), while the second implements an adaptive two dimensional fractionally spaced FeedForward Equalizer (FFE).

4.1.3. Implementation of the system.

Various test-fields have already proven the effectiveness of this optical communication scheme. Concerning the conversion of the prototypes into commercial products, while the technology related to the transmitter side is considered mature enough, for some portions of the receiver additional implementation efforts are required. In fact, to the best of our knowledge, the DSP logic exploited by the receiver has been implemented only in a suboptimal, not real-time fashion. Due to the limitations of the traditional computing scheme, the data can be reconstructed only partially, meaning that only the X% of the traffic can be restored. To fully reconstruct the data stream, the DSP has to be implemented in a parallelized fashion, with various instances running in parallel over different portion of the input signal. The instances outcomes can be reordered for reconstructing the original stream introducing a network negligible latency penalty.

FPGAs were considered the only technology capable of providing sufficient parallel processing capabilities, accommodating the special requirements of the system. Starting from an existing batch implementation of the DSP core, designed in Matlab/Fortran for running in a single-threaded fashion, an effective multi-instances system has been designed and tested on a real system. The various components designed and the approach adopted will be presented in the proceeding after a brief description of the algorithms exploited.

FIGURE 4. Automatic Frequency Compensation (AFC).



4.1.4. Frequency estimation and compensation.

As previously said, the reconstruction process imposes a requirement on the frequency offset existing between incoming data and local oscillator (LO). When the data stream is continuous, frequency estimation and compensation can be implemented with a closed-loop algorithm. This algorithm, referred as Automatic Frequency Compensation (AFC), is shown in Figure 4.

Defining T_c as the sampling time and denoting by $\hat{F}_l$ the frequency estimation at time $t = l * T_c$, the AFC input, r_l, can be frequency compensated by the formula:

$$x_l = r_l * e^{-j\phi_l} \tag{11}$$

where ϕ_l can be obtained via:

$$\phi_{l+1} = \phi_l + 2\pi\hat{F}_l T_c \tag{12}$$

As a side note, the phasor $e^{-j\phi_l}$ can be easily obtained through a fixed look-up table (LUT), as shown in Figure 4. The signal x_l goes as well to the frequency error detector (FED) block for computing the error signal ϵ_l. The frequency estimation uses this error according to the recursion:

$$\hat{F}_l = \hat{F}_{l-1} + \zeta\epsilon_l \tag{13}$$

where ζ is a proper step-size. Several classical FEDs can be chosen according to the desired performance/complexity. In [**CFFP09**] the delay&multiply (DM) FED and a FED derived from the maximum likelihood (ML) criterion [**MD97**] have been considered.

4.1.5. FeedForward Equalizer (FFE).

With the frequency of the LO correctly compensated, the non-linear attenuation of the channel can be corrected, restoring the original stream. The large part of this compensation is achieved through an adaptive Feed-Forward Equalizer. A Feed-Forward Equalizer (FFE) exploits the information contained in the signal waveform for correcting the errors introduced by the channel. For the developed system,

an adaptive FFE — i.e. a FIR filter with variable taps — has been addressed, exploiting a so-called two-dimensional fractionally-spaced FFE. The discrete two-dimensional input signal $x_{\eta k+n}$ pass through a two-dimensional filter, characterized by impulse response C_l, for $l = -Lc+1, ..., -1, 0$. Consequently, the output of the FFE will be:

$$y_{\eta k+n} = \sum_{l=-(L_c-1)}^{0} C_l x_{\eta k+n-l} \tag{14}$$

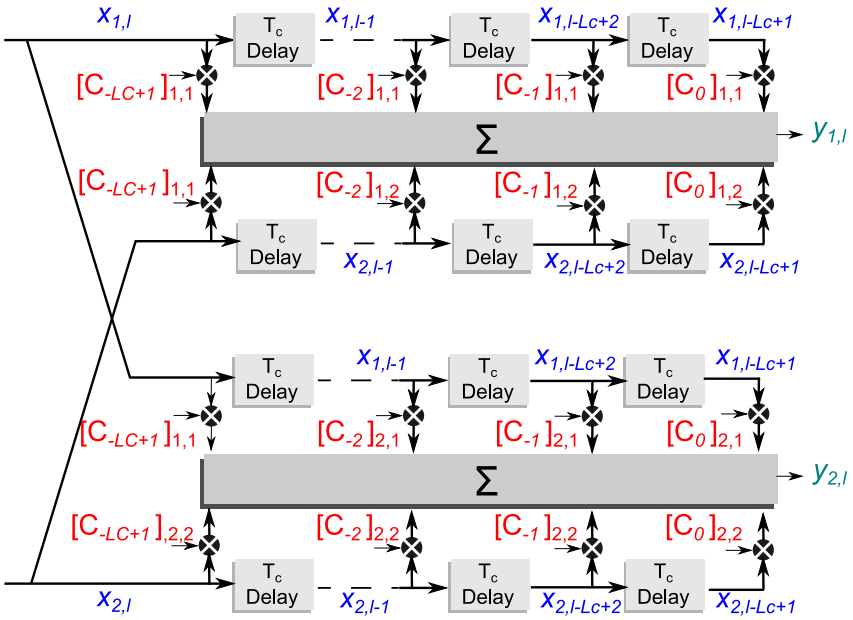
This filtering operation provides a perfect compensation for GVD and PMD, given the AFC correctness and the knowledge of the channel impulse response, Q_l. If the afore-mentioned conditions hold, this FFE scheme can also compensates for a constant phase shift that could be introduced by transmit and receive lasers. The FFE structure is shown in Figure 5 with $[C_l]_{k,n}$ indicating the (k,n) entry of C_l.

Unfortunately, in a realistic scenario, Q_l is unknown and it must be extracted from the received stream, with the filter coefficients continuously adapted for tracking the channel variations.

Implementations of this process have been addressed in various works [**CR99**, **SG99**, **PP87**]. The former solution has been identified as the most promising and adopted in the deployed scheme. This solution relies on the so-called decision-directed approach for updating the coefficients.

The statistical properties of the transmitted bits can be exploited for inferring the correctness of the reconstruction process. Considering a 4QAM — QPSK modulation the probability of decoding one of the four possible symbols should be equal to 0.25 on the long run. This homogeneous distribution of symbols is largely imposed through specific encodings in order to operate more effectively the channel.

FIGURE 5. Feed Forward Equalizer (FFE).



Unfortunately, two ambiguities will remain. The former affects all the signal components and depends by the intrinsic rotation symmetry angle ϕ of the employed alphabet. As an example, $\phi = \frac{\pi}{2}$ for QPSK and QAM alphabets. The latter ambiguity comes from the possible exchange of the two polarizations. A solution for the first problem exploits a differential encoding for the two streams. The second kind of ambiguity, instead, can be solved by inserting some known (pilot) symbols in the transmitted sequence at the beginning of the stream.

CHAPTER 5

Field Programmable Gate Technology

BEFORE detailing the implementation of the Digital Signal Processing logic for the 100+G optical receiver, it is worth introducing some general concepts related to the Field Programmable Gate Array (FPGA) technology. To a great extent, an FPGA is a programmable chip composed by a series of user configurable, highly-integrated electronic components. While, the ancestors of modern FPGAs were characterized by only two blocks (clocking network and logic blocks), modern FPGAs contain additional features such as fast transmitter/receivers (transceivers), embedded memories, PCI-express controllers and lately ARM cores.

Due to the continuous reduction of the technology process, density of these chips has grown steadily, providing benefits in terms of power consumption and chip cost. However, it must be pointed out, that FPGAs, if compared to traditional ICs, have been always characterized by higher power consumption and by lower performance. Even considering next generation devices [**www14a**, **www14f**], — produced using state of the art technology processes (14 and 16 nm respectively) — these two aspects will not change. As a matter of fact, performance and consumption penalties are intrinsically introduced by the reconfigurable nature of FPGAs. In fact, internal FPGA components can be configured and connected in an large set of combinations, and consequently clocks, data and control signals must be propagated through all these components.

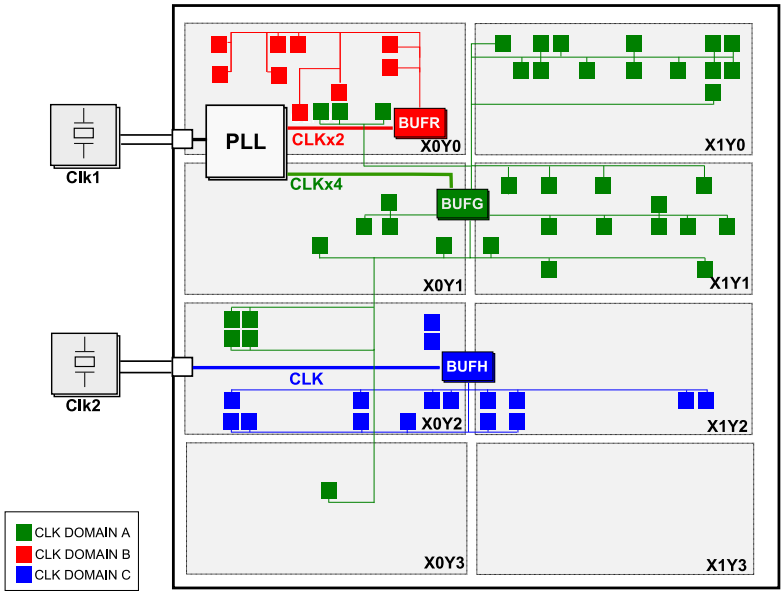
While an in-depth description of the components present in a modern FPGA is beyond the scope of this chapter, in the proceeding, the blocks exploited in the design of the DSP logic will be briefly introduced. Since the final system has been designed and tested on a Xilinx Virtex-6 FPGA, the architecture of this family of devices will be considered as a reference. In the final part of this chapter, the evaluation board used for validating the system will be described.

5.1. Clock network.

The internal blocks of an FPGA, except for limited exceptions, require at least one clock to operate. These clocks cannot be generated using only FPGA resources, at least one reference clock signal must be externally generated — by means of a quartz or a clock generator/jitter cleaner — and provided to the FPGA pins. When inside the FPGA, these clock signals can be buffered and used as they are or they can be internally modified — in terms of frequency and phase — to accommodate different needs.

The latter process relies on internal Phase Locked Loop (PLL) blocks that can be configured to generate faster or slower clocks, synchronous (in terms of phase) to the original clock sources and eventually characterized by Low ElectroMagnetic Emissions (i.e. Spread Spectrum clocks [**HFB94**]). From now on, for the sake of clarity, groups of resources clocked by the same clock source will be referred in the proceeding as *clock domains*.

FIGURE 1. Simplified clocking scheme of a Virtex6. As colours suggest, internal logic can be clocked by different clock sources (i.e. clock domains). It must be noted that in the same area, resources can be clocked by different clock sources. The depicted device contains 8 clocking regions, indicated as **X0Y0** ... **X1Y3**



Concerning the internal distribution of the clocks, two different type of connections can be selected: clock buffers or local interconnections. Clock buffers are internal connections (paths) characterized by minimal latency and low jitter that can distribute the clock along the chip. While most of them can distribute the clock only inside a limited portion of the chip — a so-called *clocking region* — a limited number of them, referred as *global buffers*, can propagate a clock signal throughout all the chip. When these resources are not used, clock signals are locally routed (local interconnections) as normal signals and latency/jitter on the clock distribution network significantly increase. A simplified clocking scheme of a Virtex6 is shown in Figure 1, for the interested reader more details related to the Virtex-6 clocking scheme can be found in [**www14e**].

It is worth noticing, that designers working with FPGAs can define multiple clocking domains operating at different frequencies. By adopting multiple clock domains, the following system aspects can be improved:

- *The system performance.*
  An important aspect related to FPGAs is the lack of a fixed operating frequency, FPGA vendors, in fact, only provide a maximum operating frequency for the various elements composing the chip. Unfortunately this value only represent an optimal value that is extremely difficult to reach in complex designs. As a matter of example, the same chip configured with two different configurations (referred in literature as *firmwares*) can run at 200 Mhz or at

20 Mhz. The so-called *slowest path* determines the maximum frequency for the specific implementation. It is represented by the time that takes for transferring data between two connected sequential elements (more in the proceeding). The system can run at a maximum clock frequency that is represented by the inverse of this propagation time. In a large number of cases, by splitting the design into multiple clock domains, the critical path can be clocked by a slower clock, while the other parts of the design can run at an higher clock rate, improving the final system throughput.

- *The resource utilization.*
  Some so-called *hard block components*, such as multipliers, are in general a precious resource in DSP related applications. Fortunately, these blocks can operate at extremely fast clock rates, usually 2 or 3 times higher than Configurable Logic Blocks (defined in next section). For this reason, designers largely rely on the *resource sharing* approach. A limited resource, such as a multiplier, can be shared for performing different operations, reducing its inactive time and avoiding the usage of other equivalent blocks.
- *The power consumption.*
  Power consumption can be split into static and dynamic power consumption. The static power consumption of an FPGA chip is device dependant and, for a given device, cannot be reduced by the system designer. Concerning instead the dynamic consumption of the chip, this latter can be significantly reduced without impairing the performance, by properly clocking the various components of the chip. In fact, over-clocking a group of components represents a waste of energy that can be limited by adopting a slower (i.e. closer to optimality) clock for that particular area. Data can be transferred between the newly generated *clock domains* by means of internal memories or synchronization signals. Eventually, when a given area is not required, the clock used within that domain can be stopped, significantly reducing the overall power consumption and thermal dissipation.

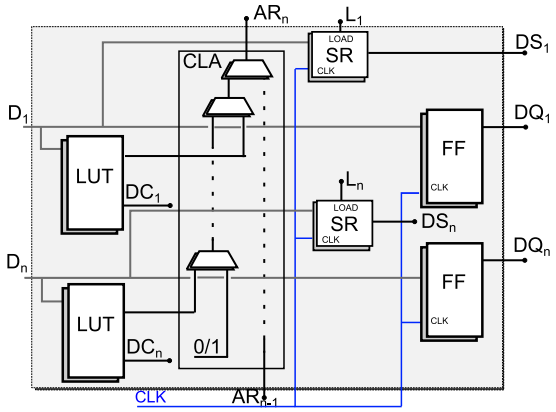
5.2. Configurable Logic Blocks.

Modern FPGAs contain huge amounts of logic blocks, spread all around the chip. Logic blocks — referred by Xilinx as Configurable Logic Blocks (CLB) — can be exploited for designing sequential as well as combinatorial logic. The output of the former will change synchronously to clock transitions, while for combinatorial blocks, signals change independently to the clock signal.

Sequential logic is largely used for implementing memory blocks, shift registers, and in general for synchronising signals to a specific clock domain. As will be discussed later, the same functionalities can be achieved by means of internal memory blocks. Due to this overlapping of functionalities, designers have to carefully select one resource or the other accordingly to the design characteristics. This concept will be clarified more in the *Internal Memories* Section.

Concerning the design of combinatorial logic, instead, CLB's internal Look Up Tables (LUT) can be used for implementing boolean transformations as well as signals multiplexing. CLBs also contain multiplexers as well as high-speed carry logic for arithmetic functions (i.e. used for implementing addition and substraction operations). Figure 2 provides a simplified overview of a Xilinx CLB™block.

FIGURE 2. Simplified structure of a Virtex6 Configurable Logic Block. Shift Registers (SR) and Flip Flops (FF) are the main sequential components of the block, while Look Up Tables (LUT) and the Carry Logic for Arithmetic operations (CLA) are the principal combinatorial ones.



It is important to notice that a significant usage of the CLB resources available leads, in general, to the degradation of the maximum system performance (due to the unavoidable creation of long connections between spread logic blocks). Considering the Virtex-6 chips, the maximum clock rate for CLBs is 600 Mhz but for average sized designs, it will be difficult for the designer to overpass the 200 Mhz clock rate without relying on internal memories or DSP blocks.

5.3. Internal memories.

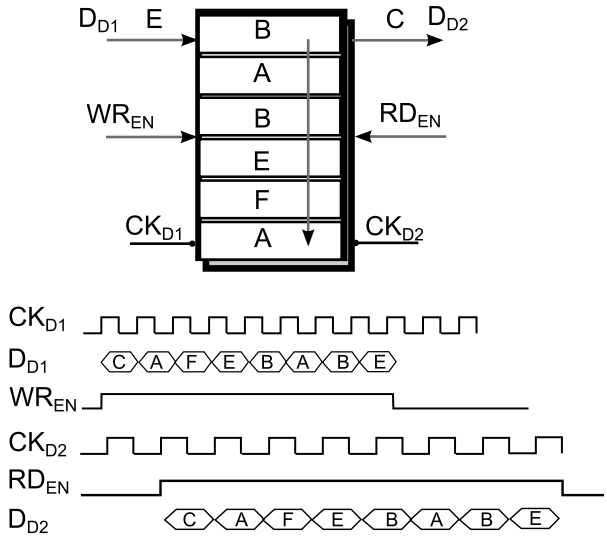
While the clocking network and the CLB (or their equivalents) were already available in the first FPGA specimens, modern FPGAs come with additional *hard blocks* to foster advanced applications. Hard blocks are specialized parts of the FPGA that provide to the system some specific functionalities in an extremely optimized fashion. They have relatively small silica footprints, can run at fast clock rates and can embed analog functionalities. Memories, are an excellent example of hard blocks.

Even though small memories can be implemented relying on CLBs resources, when depth and width sizes increase significantly, the amount of CLBs resources required can overpass the available ones. Instead, by using the hard block memories available on the chip, the same amount of data can be stored in a small area with significant benefits in terms of maximum achievable clock rate. In fact due to their density, data-paths tend to be shorter and thus data can be transmitted from sequential block to the other via a short and fast link. Additional features, such as asymmetric aspect ratio (read and write words can have a different number of bits) can be extremely useful during the system design.

Furthermore, it must be pointed out that modern hard block memories are implemented as dual clock memories — i.e. the clock used for reading the memory may be different from the one used for writing into it — providing an effective way for transferring data between different clock domains, without incurring in metastability [**www14d**].

Figure 3 depicts the usage of hard-block memories for interfacing different clock domains in a reliable way.

FIGURE 3. *Hard-block memory* used for clock domain crossing. Data are produced in the clock domain $\mathbf{CK_{D1}}$ and consumed by logic running at $\mathbf{CK_{D2}}$. All the non-deterministic behaviours (meta-stabilities) are removed by using the memory as a dual clock Fifo In First Out (FIFO) element.



5.4. Digital Signal Processor blocks.

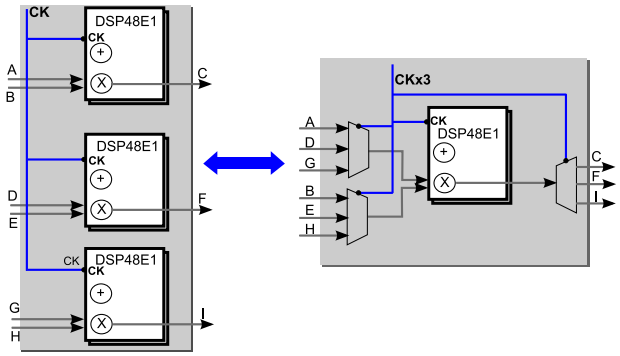
As it will appear more clear in the proceeding, the Digital Signal Processing logic used for reconstructing the data stream contains a large number of fixed point multiplications and sums. These two operations would require a massive amount of resources if implemented using CLB resources and the performance will be extremely unsatisfactory. Fortunately, Virtex-6 FPGAs come with a relatively high number of DSP48E1 hard-blocks [**www11b**]. DSP48E1s can be seen as small arithmetic cores that can be exploited for performing multiplications, additions, multiply and accumulate (MAC) operations as well as for patterns detection. Due to the high level of optimization of these blocks, the maximum clock at which they can be operated is extremely high (for Virtex-6 roughly 400 Mhz). Similar operations, implemented only with CLB blocks will likely run at one tenth of these clock rates.

Their fast nature can be exploited for increasing the overall system throughput, via the so-called *resource sharing* approach.

As a matter of example, considering the flow depicted in Figure 4, instead of using three different DSP48E1 blocks for performing the three independent operations running at clock **CK**, the same result can be achieved by using only one DSP48E1 clocked with a three times faster clock (**CKx3**) and by multiplexing/demultiplexing its inputs/outputs.

Additional benefits in terms of resources utilization can be achieved when multiple operations are brought back to a supported atomic operation of the DSP48E1 module, such as for the standard filtering formula:

FIGURE 4. *Hard-block DSP48E1* sharing. When applicable, the resource sharing approach can lead to a better utilization of the available resources.



$$\hat{x}(t) = x(t) + x(t-1) * \mathbf{C} \tag{15}$$

that can be implemented using a single DSP48E1 block, through the supported Multiple-and-ACcumulate (MAC) atomic operation.

5.5. Available resources.

Every FPGA device comes with a fixed amount of clock resources, CLBs, memory blocks and DSP48E1 elements. Considering modern FPGAs the amount of CLBs they provide is largely sufficient for fitting almost any complex design. Unfortunately, it commonly happens that the amount of the other types of resources does not completely cover the request of the design. In the latter case, the design is generally investigated for minimising the utilization of that specific resource and if it still does not fit into a single FPGA can be split into multiple FPGAs.

Concerning the design of the DSP for ultra-fast optical receiver, since the beginning, the amount of DSP48E1 blocks required was identified as critical. A large amount of complex multiplications and sums is required by the FFE block and by the coefficients update block. Splitting the design into multiple FPGA can provide a working system, but in order to reduce the number of FPGAs required, large efforts were spent on optimizing the design. Table 1 lists the amount of resources available on the XC6VLX240T Virtex-6 chip used for testing the design.

TABLE 1. Amount of resources available on the Virtex-6 XC6VLX240T chip

Resource Type	Available Quantity
PLL (MMCM)	10
CLBs	37,680
BlockRAMs (18 Kb)	832
DSP48E1	768

5.6. Development device

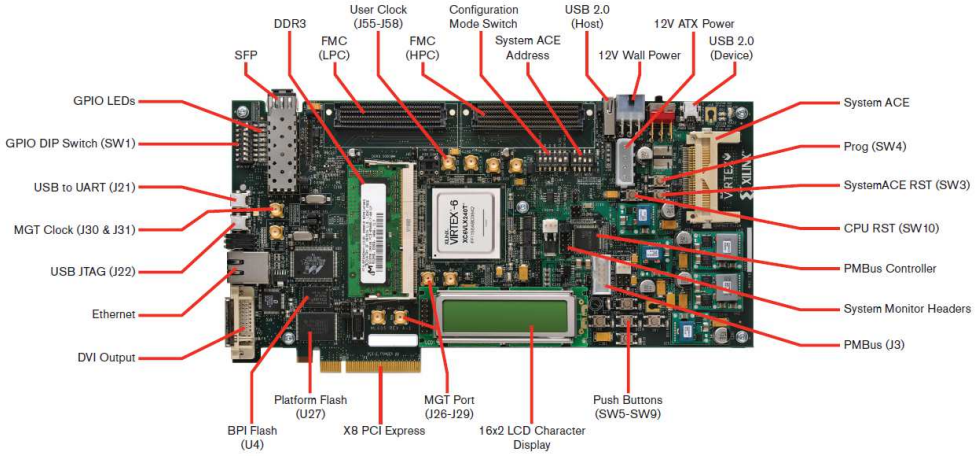
Designing solutions for FPGAs requires a significant amount of efforts and experience. Even though verification tools are provided to the designer, the final validation of the system is generally conducted on hardware. This latter can be either the final device or an evaluation board that contains the same type of FPGA chip. FPGA vendors provide evaluation boards in which their chips are connected to additional components, such as Ethernet/USB adapters, LCD/LEDs, clock generators etcetera.

When this research activity was conducted, the ML605 Evaluation Kit [**www11a**] represented the Xilinx state-of-the-art technology. This evaluation board comes with:

- A Virtex-6 XC6VLX240T FPGA chip.
- A USB-JTAG programming interface.
- A DDR3 memory module (1GB)
- An Ethernet adapter
- PCI-Express Connector
- Two Full Mezzanine Card (FMC) connectors
- Leds, Buttons and Switches

Being the device fully supported by the Simulink™ environment, the debug and validation process was largely simplified as will be described in the proceeding. Figure 5 shows the ML605 evaluation board, highlighting its main components.

FIGURE 5. Xilinx ML605 evaluation board. Courtesy of Xilinx.



CHAPTER 6

A model for the DSP core

THE design of the logic for the optical receiver started with the creation of a functional model of the system. As afore-mentioned, a model based approach could provide substantial benefits to the system designers, easing in particular future extensions/upgrades of the system as well as the debug process.

Among the model-based design tools, Mathworks Simulink™ starts to gain popularity, thanks to its automatic code generation feature [**Lam12**,**ACO12**]. Concerning the design of the receiver, Simulink™ could be used for obtaining the HDL implementation of the targeted system via the HDL coder toolbox [**www14c**].

The generation of the model was conducted in a stepwise fashion with in-depth verification of all the designed subcomponents. Table 6 lists the various components deployed, discerning between the blocks implemented using only Simulink™ language constructs and the ones containing calls to external procedures (Matlab and/or Fortran).

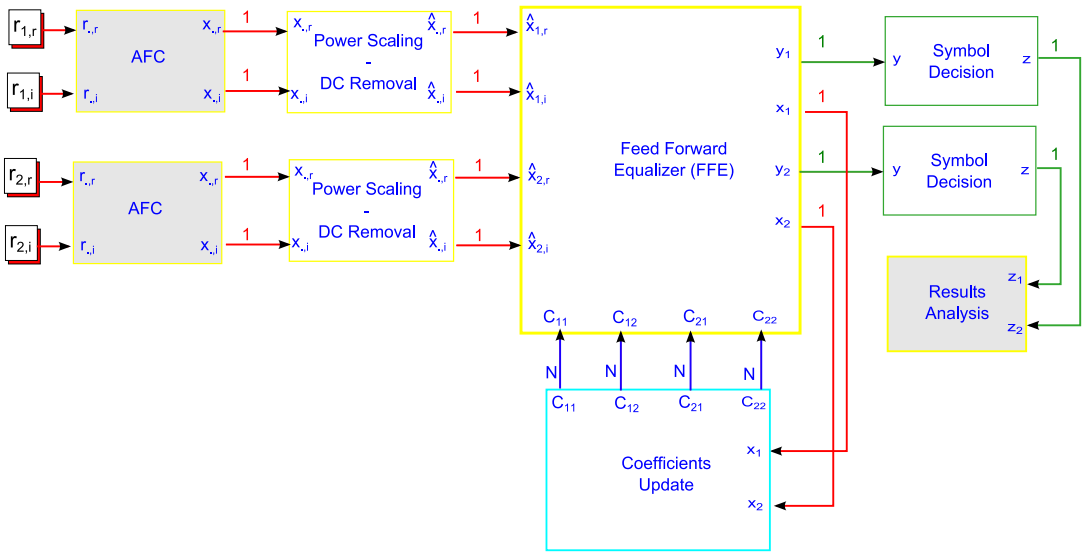
Matlab routine	**Simulink™ block**
Input conversion	Power normalization
	DC removal
AFC	Equalizer
	Symbol Decision
Results analysis	Coefficients Update

To a great extent, the design and verification of these components was eased by two aspects. Firstly, real datasets, collected with a fast sampling oscilloscope, were provided. Secondly, a largely tested and validated batch implementation of the DSP logic was already available. The latter comes as a set of Fortran/Matlab procedures designed for standard CPUs (single-thread implementation). Thanks to the datasets, it was possible to compare the behaviour of the existing batch implementation against the under-design model in a realistic scenario. Furthermore, Simulink™ can call and run external code, providing to the designer, the possibility of generating mixed Simulink/Matlab-Fortran-C implementation. For the DSP receiver calls to the batch Fortran implementation were used as place-holders for not yet deployed components.

For enhancing code reuse and extendibility, the designed components were grouped into a library. The full system implementation is shown in Figure 1.

The various blocks will be detailed in the proceeding, focusing, in particular, on their cost in terms of FPGA resources.

FIGURE 1. DSP top level view. The system is composed of various I/O blocks (entering arrows: inputs; exiting arrows: outputs) connected by buses of various width (the number on the connecting lines represents the vectors size). Different clock domains are depicted by means of different colours for the signal and for the I/O block. Greyed blocks are implemented in a batch mode and can not be converted to HDL code by the HDL coder tool.



6.1. Inputs generation.

Simulink™ is a discrete time simulator that can be used to efficiently generate system models and to simulate their dynamic behaviour. The DSP logic under design can be seen as a complex, time-varying control system, characterized by a significant large number of internal signals. Concerning the latter aspect, the possibility of verifying their evolution in time has been considered extremely valuable since the initial design stages. In particular, the debug process has been significantly eased by the possibility of probing the internal signals.

It must be noted, that being Simulink™ a time-based simulator, every signals must by associated to a time reference. In other words, every vector signal is handled internally by Simulink™ as a $[data, time]$ matrix. Fortunately enough, only the model inputs have to be time-referenced: for all the internal signals, in fact, the tool takes care of propagating the temporal reference.

Being the input of the DSP the output of the ADC block (as depicted in Figure 3 at page 42), data traces collected by means of a fast scope (50 Gsamples/s) could be used in place of all the acquisition chain without impairing the validation process.

The data digitalized by the scope — more in the specific the I/Q pair of the two polarizations — had to be converted into four time-referenced vectors. To be used inside the Simulink™ environment, input traces must be packed into a structured record:

```
struct Input {
        time                    // time vector;
        signals.values          // the sampled input vector of size Nx1;
        signals.dimensions      // the number of vectors composing values, therefore, 1.
};
```

A Matlab script has been designed for performing this conversion.

6.2. Automatic Frequency Compensation (AFC)

As introduced in the previous Chapter, the Automatic Frequency Compensation represents an effective way for compensating mismatches between received clock and local laser oscillator frequencies. While small mismatches are compensated by the FFE, larger ones must be corrected by the AFC.

Concerning the implementation of this block, after the initial definition of its main structure, its design was halted due to the lack of a valid method for assessing the correctness of the generated outputs. In fact, due to the high quality of the LO used in the test-beds, the frequency offset remains negligible and well within the correction capabilities of the FFE block for all the samples collected. Considering this limitation, the batch version of the AFC was preserved and wrapped into a Matlab script for being inserted into the model, while the real-time version was considered a feature of secondary importance.

FIGURE 2. AFC Scheme. Incoming data, $r_{K,\eta}$ are frequency compensated by rotating them for a corrective phase ϕ that is obtained from an analysis of the statistical properties of the the block output $x_{K,\eta}$. In order to maximize the performance of this block, the frequency estimated value, F, is computed over a down-sampled set of inputs (i.e. taking one sample every N.)

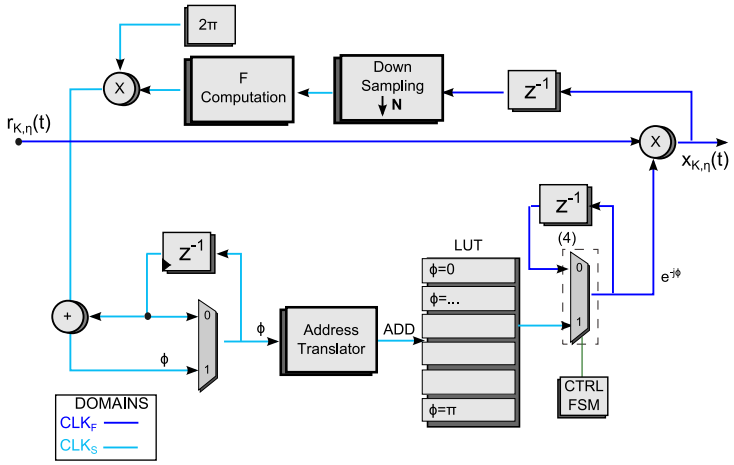


Figure 2 provides a graphical insight of the initial — not optimized — real-time implementation of this block. Preliminary tests have shown that this implementation provides correct results when the LO frequency variation per unit of time is relatively small and when the down-sampling ratio N is not excessively high. It is worth noticing that two clock domains ($\mathbf{CLK_F}$ and $\mathbf{CLK_S}$) are used for separating the frequency offset compensation part from the offset calculation block. In this way the less DSP48E1 blocks are required and, at the same time, the achievable throughput is significantly increased.

6.3. Power normalization and DC removal blocks

Fluctuations on the transmitter laser power as well as minimal variations of the attenuation factor of the fibre can deteriorate the convergence properties of the FFE block. While the former effect can be minimized by improving the power stage of the laser (i.e. by adopting more costly units), the latter depends on ambient temperature variations and it is extremely difficult to prevent even when special fibre coatings are used.

Fortunately enough, a normalization of the input signal represents an effective solution to both the problems and can be implemented inside the DSP core. Feed with normalized inputs the FFE block will not have to compensate for signal power variations and will converge more promptly to the optimal coefficients. Defining the digitalized I/Q components of the K polarization as $x_{K,r}(t)$ and $x_{K,i}(t)$ respectively, for $K = 1,2$ their normalized counterparts, $\tilde{x}_{1,i}(t)$, $\tilde{x}_{1,i}(t)$, $\tilde{x}_{2,r}(t)$ and $\tilde{x}_{2,i}(t)$ can be obtained via formula 16:

$$\hat{x}_{K,\eta}(t) = x_{K,\eta}(t) * \mathbf{PN}_{K,\eta}(t) \tag{16}$$

for $K = \{1,2\}$ and $\eta = \{r,i\}$. $\mathbf{PN}_{K,\eta}(t)$ can be computed over a subset of the samples using the formula:

FIGURE 3. Power Normalization Scheme. Various sub-blocks can be identified. (1) the accumulation FIFO, (2) the mean square value computation block, (3) the computation of $\mathbf{PN}_{K,\eta}(t)$ and finally (4) the update of the $\mathbf{PN}_{K,\eta}(.)$ value.

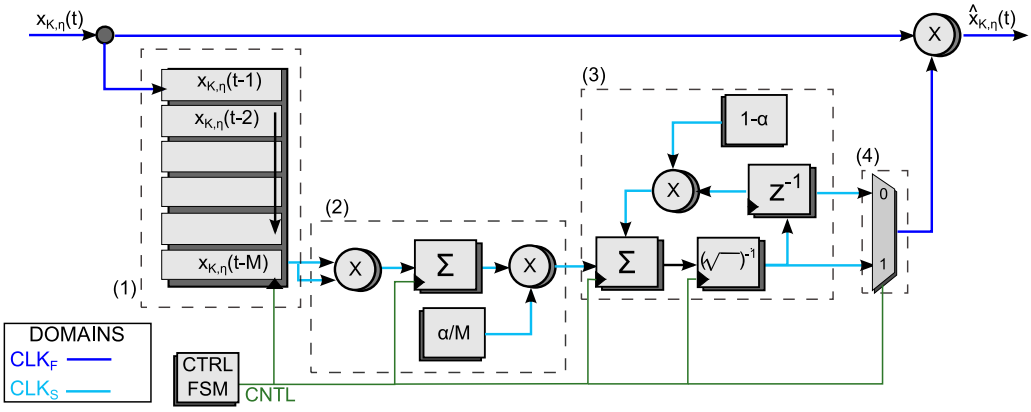
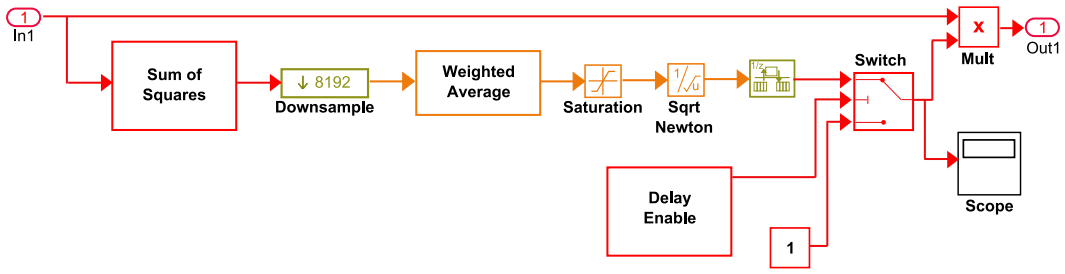


FIGURE 4. Power Normalization Scheme, Simulink implementation



$$\mathbf{PN}_{K,\eta}(t) = \frac{1}{\sqrt{\alpha * \mu_{x^2_{K,\eta}(q:q+M)} + (1-\alpha) * \mathbf{PN}_{K,\eta}(t-Q)}} \tag{17}$$

Formula 17 exploits the slow dynamic of the signal power for reducing the average computational cost of this block. In fact, with $Q >> M$ only M samples out of Q have to be stored and processed to compute $\mathbf{PN}_{K,\eta}(t)$. Two different clock domains can be exploited: a relatively faster one, $\mathbf{CLK_F}$, will clock the logic implementing Formula 16, while a slower one, $\mathbf{CLK_S}$, will be used by the block that updates the $\mathbf{PN}_{K,\eta}(t)$ value as in Formula 17. As introduced in the previous Chapter, it can be easily seen how this approach will lead to a better utilization of the available FPGA resources, without any negative impact on the system throughput.

More in details, the samples, are stored in a First In First Out (FIFO) memory that writes new data at $\mathbf{CLK_F}$ frequency. When $\mathbf{M}$ samples are collected, writing is stopped and data can be extracted, using $\mathbf{CLK_S}$ as read clock. Thanks to this approach, the computation of $\mathbf{PN}_{K,\eta}(t)$ can be split into several clock cycles, leading to a more FPGA efficient implementation. When after $\mathbf{K}$ clock cycles a new $\mathbf{PN}_{K,\eta}(t)$ is produced it can be used instead of $\mathbf{PN}_{K,\eta}(t-Q)$ without impairing the quality of the output. Figure 3 provides a graphical insight of this block. It is worth noticing that by using the Newton approximation [**HP98**] the inverse square root can be computed in a iterative fashion, characterized by growing accuracy. Again, being possible to use multiple clock cycles for computing this value, high accuracy can be achieved without affecting the performance of the system (i.e. without introducing a *critical path*).

Additional improvements in the convergence speed of the FFE coefficients can be achieved by removing from the power normalized signals $\tilde{x}_{K,\eta}(t)$ the continuous (DC) component. The DC-free signals, $\hat{x}_{K,\eta}(t)$, can be obtained via Formula 18.

$$\hat{x}_{K,\eta}(t) = (\tilde{x}_{K,\eta}(t) - \mu_{\hat{x}_{K,\eta}(t-N:t)}) * \mathbf{DC}_K(t) \tag{18}$$

where $\mathbf{DC}_K(t)$ can be computed over a limited number of samples, as for the normalization factor, using the formula:

$$\mathbf{DC}_K(t) = \frac{1}{\sqrt{\alpha * [\mu_{[\tilde{x}_{K,r}(t-N:t)+\tilde{x}_{K,i}(t-N:t)]^2} + \mu^2_{\tilde{x}_{K,r}(t-N:t)} + \mu^2_{\tilde{x}_{K,i}(t-N:t)}] + (1-\alpha) * \mathbf{DC}_K(t-Q)}} \tag{19}$$

Power normalization and DC removal have been implemented in Simulink using the standard building blocks. Except for the probing tool (the *scope* blocks), all the other blocks adopted are supported by the automatic HDL code generation toolbox. Therefore, an HDL implementation of the system can be automatically generated by Simulink™, largely simplifying the coding and debug processes. The afore-mentioned *scope* blocks have been extensively used for monitoring internal signals and, being ignored by the code generation tool, do not impair the performance of the final system.

The Simulink™ blocks for the power normalization is shown in Figure 4. For the sake of conciseness, the DC removal block, due to the high level of similarity with the power normalization block, is not shown in this dissertation.

6.4. FFE block

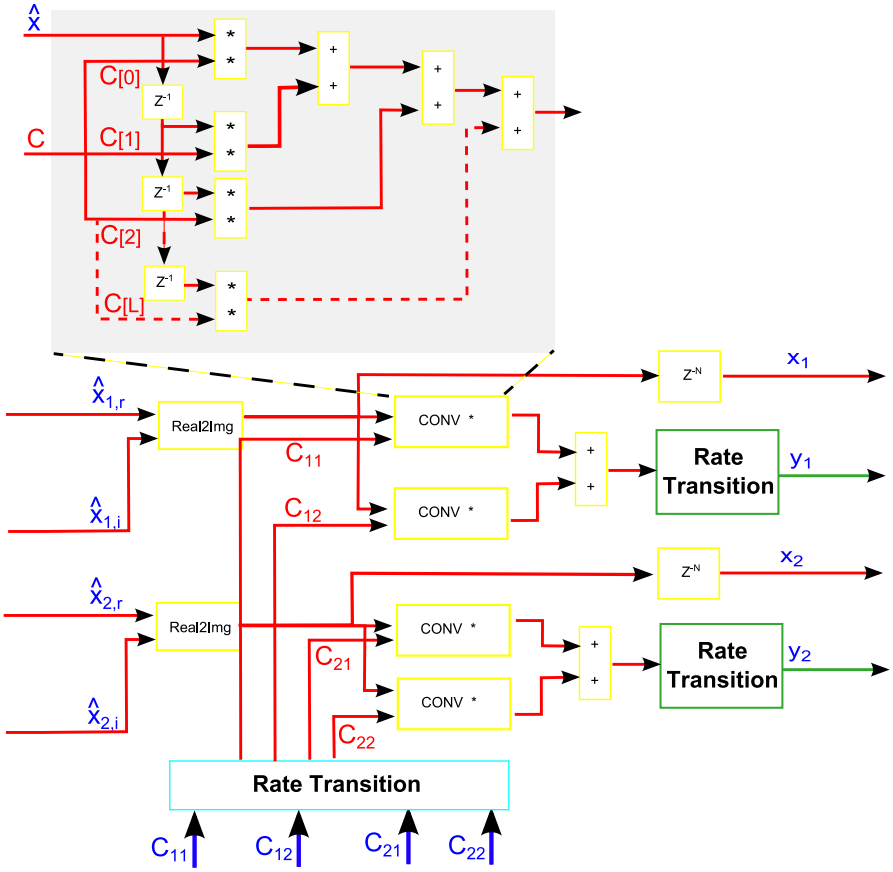
The Feed Forward Equalizer convolves the received data stream with the inverse of the channel response for compensating the channel degrading effects. The inverse of the channel response, that is generated by the coefficients update block, is represented by a matrix of size 4x**N**, with **N** the number of filter taps (aka the filter order). Designers should select **N** accordingly to the properties of the channel, for long-haul, low quality fibres more taps (i.e. a bigger value for **N**) are in general required. For all the traces collected in the testbeds, **N**=5 proved to be sufficient.

Concerning the implementation on FPGA, due to the Convolution Theorem, the signal samples can be convoluted with the filter coefficients either in the time or in the frequency domain. While in terms of results the two transformations are equivalent, their implementation is extremely different.

Starting with the time-domain transformation, the filter coefficients — received from the coefficients update block — can be used accordingly to Formula 6.6 of Page 64 for reconstructing the transmitted data stream. It is worth noticing that in this implementation, the number of multipliers/adders increases linearly with **N**, as shown in Figure 5. The final number of DSP48E1 blocks will depend by the number of multipliers/adders required as well as by their accuracy in terms of bits. The two operands of the DSP48E1 block can have, in fact, a maximum resolution of 25 and 18 bits, thus, whenever one of the inputs overpass this resolution multiple DSP48E1 must be instantiated. Concerning the latency of this block, even when pipeline stages are introduced for improving the timing performance of the system, the number of clock cycles required is extremely limited.

The cost, in terms of DSP48E1 blocks, of the frequency-based implementation instead is not only a function of **N**. In this formulation, for the convolution another system parameter affects the final cost of the block: **K** the depth of the Fast Fourier Transform (FFT) operation. In fact, the depth of the FFT operation affects the amount of resources requested by the block in terms of memory and DSP48E1 blocks. More in details, 4 FFTs and 4 Inverse FFTs (IFFTs) have to be instantiated for transforming the complex-based signal vectors and the coefficient vectors from the time domain time to the frequency domain and vice-versa. The amount of resources required for implementing these transformations depends by **K** and by the type of FFT implementation selected. As a matter of example, for $\mathbf{K} = 1024$, the Xilinx FFT IPCore [**www11c**] — when the implementation with the smallest latency (2167 clock cycles) is addressed — requires 32 DSP48E1 and 8 BlockRAM blocks. The same configuration for $\mathbf{K} = 4096$ needs 78 DSP48E1 and 30 BlockRAM units. By selecting a slower implementation (in terms of processing latency), the number of DSP48E1 can be brought down to 3 for $\mathbf{K} = 1024$ but the latency

FIGURE 5. Equalizer: time domain convolution. Each convolution requires **N** multiplications and sums that can be implemented as **N** Multiply-ACcumulate (MAC) operations

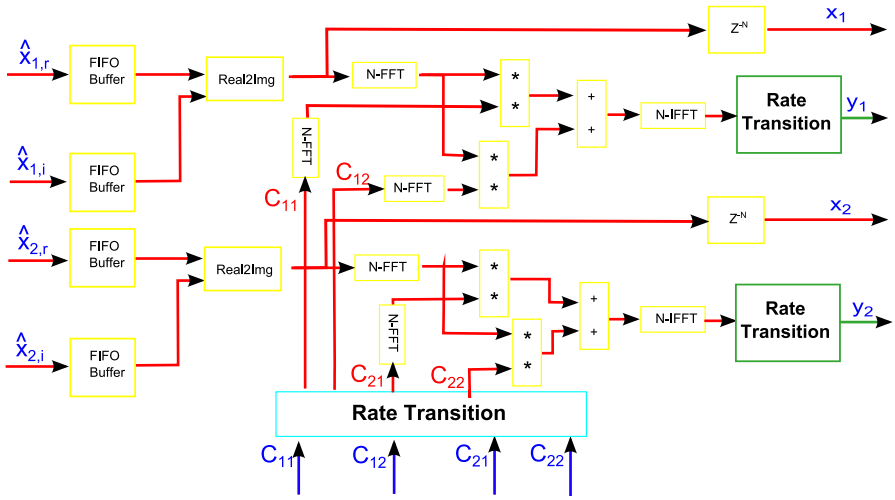


grows sixfold (12453 clock cycles). It is worth noticing that this extremely high latency comes from the nature of the FFT implementation: data are in fact transformed one by one and a new output is presented to the user every clock cycle after the initial latency.

Additionally, 4 complex multiplications are required to convolve the frequency based signals with the filter coefficients. Figure 6 depicts the main components of this implementation.

For deciding whether to use the time or the frequency domain formulation, the smallest values for the **N** and **K** parameters had to be estimated. The estimation is mandatory: **N** and **K** are channel dependent and they cannot be obtained via a closed-formula. Fortunately enough, an iterative approach that relies on the deployed DSP system can be exploited. The optimal value of **N** can be obtained feeding the model with real data traces and by analysing the magnitude of the various filter coefficients. While all the coefficients will be likely different from zero, it may exist an index **L** such that $\mathbf{L} < \mathbf{N}$ and $|C_i| < \mathbf{THR}$ for $i > \mathbf{L}$ and **THR** sufficiently small. If **L** exists it can be safely considered as the number of coefficients (taps) required by the filter.

FIGURE 6. Equalizer: frequency domain convolution. Blocks labelled **N**-FFT and **N**-IFFT are respectively, the Fast Fourier Transform and its Inverse of depth **N**, respectively.



A similar approach can be used for selecting an optimal value for **K**. It must be pointed out that **K**, due to the inner structure of the FFT implementation, is strictly a power of 2. By running the model using different values of **K** and by examining the quality of the outputs against the golden-reference results, a **K** value that provides the required trade-off performance/cost can be identified.

When both **N** and **K** are known, the costs related to the two different implementations can be computed and compared.

6.5. Symbol decision

The equalized samples $y_1(t)$ and $y_2(t)$ represent a "channel compensated" version of the transmitted stream. From them it is possible to reconstruct the sequence of transmitted symbols adopting a symbol-by-symbol asynchronous detection strategy. In the implemented system, various approaches have been converted into Simulink™ models and investigated. Among them, the approach proposed by Schober [**SG99**] provides the best trade-off between performance and computational cost. Symbols can be reconstructed via the following formula:

$$\hat{d}_{i,k} = \arg\max_{d_{i,k}} \left[\left| y_{i,k\eta} d^*_{i,k} + g_{i,k} \right| - \frac{\left| d_{i,k} \right|^2}{2} \right] \tag{20}$$

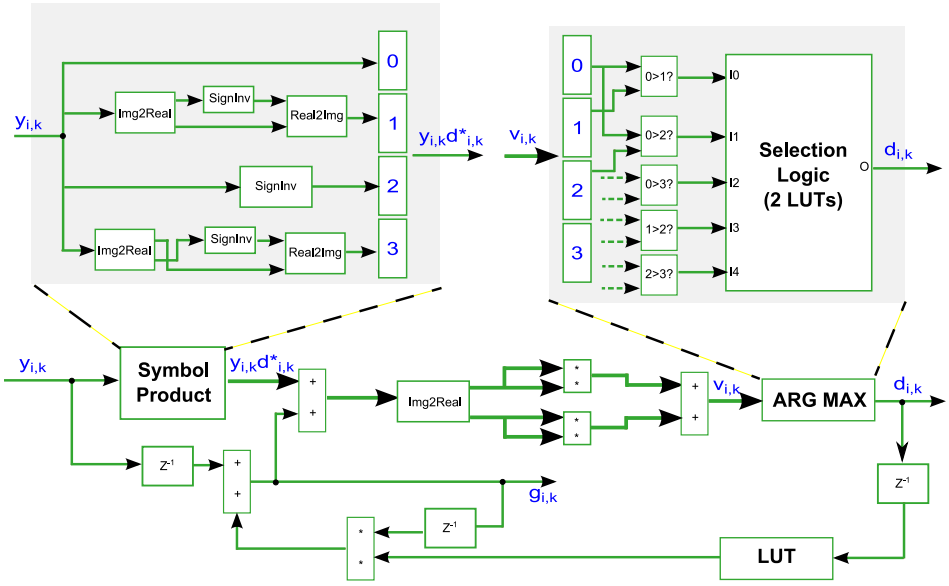
where $d^*_{i,\cdot}$ is the one of the N constellation symbols obtained from $e^{-i\frac{2\pi}{N}}$ and $g_{i,k}$ is given by:

$$g_{i,k} = \beta g_{i,k-1} + y_{i,(k-1)\eta} \hat{d}^*_{i,k-1} \tag{21}$$

In order to reduce the computational cost of this block $d^*_{i,.}$ has been expanded for $\mathbf{N} = 4$ (QPSK modulation), avoiding the computation of the complex exponential thus significantly reducing the number of multiplications required. Concerning the argmax operation, this has been explicitly encoded as a block of combinatorial logic. Figure 7 presents the final implementation of this block.

While the cost in terms of resources of this implementation is relatively small, the feedback used for computing $g_{i,k}$ represents, as it will be shown later, an extremely difficult to optimize *slow path*.

FIGURE 7. Symbol decision Simulink™ implementation. The SignInv block relies on the signed representation of the signals for inverting the sign of its input value. Concerning the Selection Logic block, it is represented by a group of LookUp Tables (LUTs) configured for outputting the symbol index [0,1,2,3] using five inequality conditions as inputs.



6.6. Coefficients update

As already stated, the equalization of the received data stream represents an effective way for compensating all the degrading effects introduced by optical link, but to operate correctly it requires the knowledge of the channel response. Being this latter time-varying and link dependant, iterative approaches are adopted for obtaining an estimation of it. As a matter of fact, various works propose different techniques for computing and updating the coefficients used by the FFE.

Among the possible implementation of the update formula, the so-called *symbol-time adjustment* [**PM08**] has been adopted in the deployed system. The coefficients are updated using the following formula:

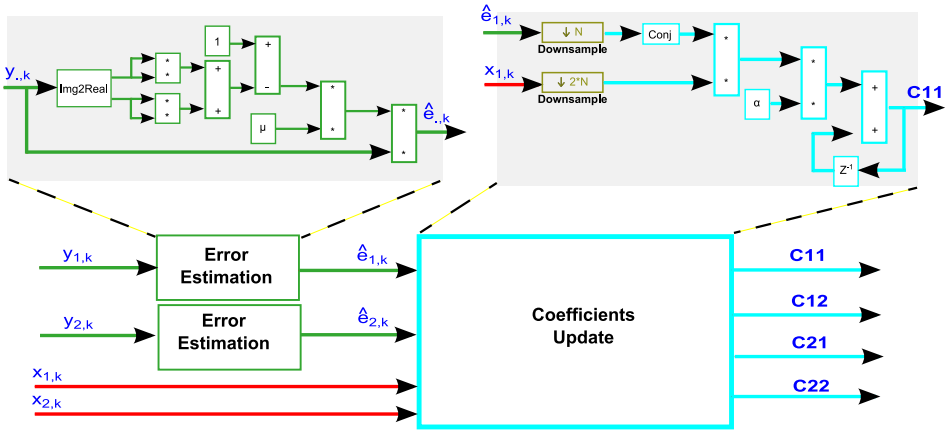
$$\hat{C}_l^{(k+1)} = \hat{C}_l^{(k)} - \alpha \hat{e}_k x_{k\eta-l}^H \tag{22}$$

with $x_{k\eta-l}^H$ the conjugated value of $x_{k\eta-l}$, α a proper step-size [0,1] and the error $\hat{e}_k$, computed via:

$$\hat{e}_k = \mu * (1 - \left| y_{k\eta - l} \right|)^2 * (y_{k\eta - l})) \tag{23}$$

with again μ a design parameter step size in the range [0,1]. A Simulink™ implementation of the update formula has been implemented and tested, varying as well, the update frequency of the coefficients. Figure 8 shows the implemented block.

FIGURE 8. Coefficients update block. Two main parts can be identified, the estimation of the current reconstruction error and the computation of the new coefficients.



6.7. Fixed Point conversion

All the internal operations contained in the normalization logic, in the FFE, in the coefficients update and in the symbol decision logic were initially based on a 32 bits single precision floating point representation. While FPGA Vendors provide floating point libraries (largely based on the IEEE-754 standard), the use of floating point logic is largely discouraged being the fixed point implementation more efficient in terms of resource utilization and throughput.

When converting a floating point implementation into a fixed point one the designer must cope with new limitations/constraints. The definition of the right amount of bits used for representing a signal may not suffice and signals may need to be preconditioned/postconditioned in order to minimize the impact of underflows and overflows. Simulink™ can be used for simulating the impact of different fixed point representations. For all the signals, in fact, the fractional and integer components can be parametrized in terms of bits exploited. Running — as it was done for the DSP receiver — an external configuration script, the designer can test different set of representations in order to minimize the logic used and the number of DSP48E1 required. When the results are satisfactory, HDL coder will generate the HDL code accordingly.

Concerning the DSP receiver, the available testbed traces were used as inputs to the system and, block by block, the signals were converted from the floating point representation to a fixed point one. In case of overflows or underflows, the tool was raising error messages, pointing to the specific part of the design that suffered the event.

Exploiting a custom made analysis routine it was possible to identify the performance degradation introduced by the fixed point representation. The impact of using different representations for the input vectors can be measured in terms of variation from the golden reference results, using for example the normalized quadratic error:

$$\mathbf{NQE} = \frac{|{Z_{ref}}^2 - {Z_{fp}}^2|}{\mathbf{N} * {Z_{ref}}^2} \tag{24}$$

where **N** represents the number of samples compared, Z_{ref} and Z_{fp} are the DSP output under comparison, respectively generated using floating point and fixed point accuracies.

A small extract of these analysis is depicted in Table 1 where different AFC inputs representations are investigated, measuring, in particular, the final impact on the FFE output. It must be pointed out that the final implementation, due to the fully parametric nature, requires limited efforts for being extended. When the representation of the DSP inputs changes, the modification is propagated through all the design and the representation of all the internal signals is updated consequently.

TABLE 1. Performance penalty introduced by reducing the representation dynamic. The **NQE** is measured varying the representation dynamic of the AFC inputs (from 7.0 to 3.0)

Number Bits	**NQE** Time-FFE	**NQE** Freq-FFE
7	1.45e-06	1.88e-06
6	1.40e-06	1.87e-06
5	2.37e-06	3.04e-06
4	1.97e-05	2.21e-05
3	8.50e-04	1.08e-03

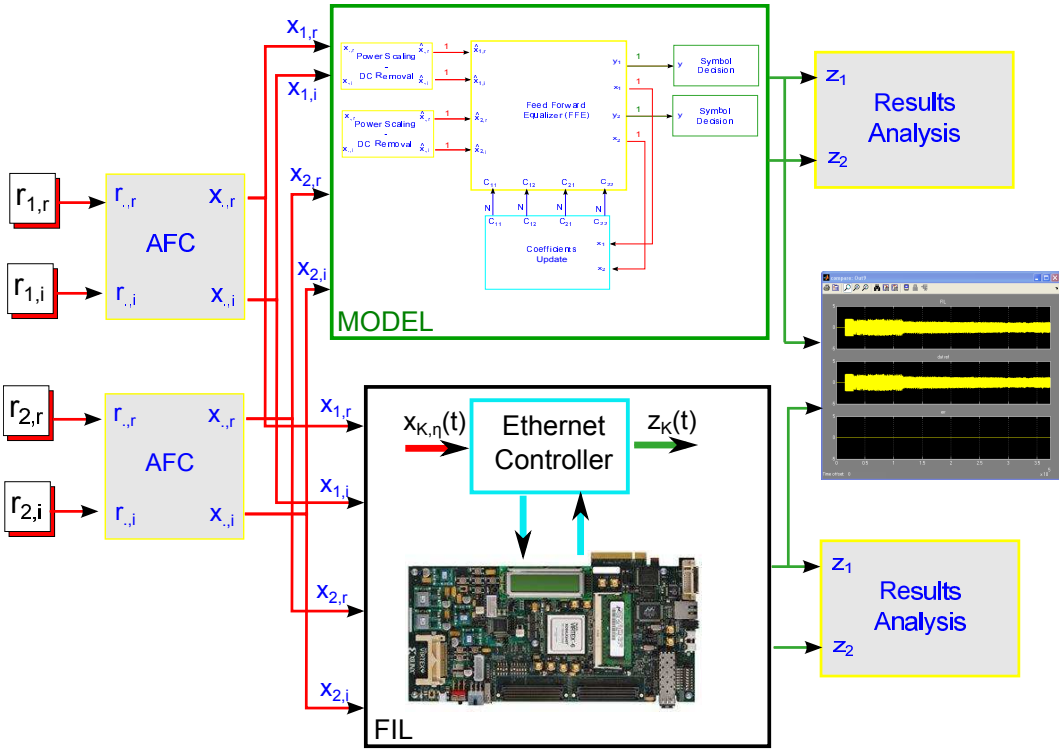
CHAPTER 7

System implementation and validation

The model described in the previous chapter satisfies all the requirements imposed by the Simulink HDL Coder tool and thus can be automatically converted to HDL code. HDL coder can be configured to generate Verilog or VHDL code and to automatically apply standard optimization techniques on the generated code. Being fully integrated with Simulink™, debug and validation phases result simplified as it will be pointed out in the proceeding.

Among the code optimization provided, HDL coder can automatically identify candidates for *resource sharing* and can improve the timing performance of the system by inserting pipeline registers

FIGURE 1. FPGA in the Loop. The validation of the automatically generated firmware has been conducted using an ML605 evaluation board [**www11a**]. The original model and the firmware generated by HDL Coder receive the same input signals and their outputs can be compared for validation purposes.



into the system. The optimizations will be back-annotated into the initial model, providing the designer with a way for verifying their correctness in simulation.

The generated implementation can be furthermore validated using the so-called *FPGA in the Loop* (FIL) feature of the tool. In this case, the generated firmware is extended by the addition of an Ethernet communication interface. A connected FPGA evaluation board is programmed by Simulink™ (relying on Xilinx commands) and data are shared between PC and FPGA via the Ethernet interface. The model of the system will be substituted by a black-box but, still, its ports can be probed for verifying the correctness of the model-to-code conversion. Figure 1 depicts this validation mode.

Furthermore, the tool provides an early estimation of the amount of resources required, easing the optimization phase. The final performance and utilization of the FPGA instead are generated running the proprietary Xilinx tools. Next section will provide some utilization figures related to this implementation.

7.1. Converted model: preliminary results

When the results provided by the *FPGA in the Loop* were considered satisfactory, the system size was investigated. Table 7.1 lists the resource estimations generated by Simulink™ for each block, while Table 7.1 summarizes the Xilinx resources utilization report of the firmware generated targeting the ML605 FPGA.

TABLE 1. Simulink™ estimated number of resources used. FFE implemented in the time domain with filter length equal to 5.

Block(s)	Multipliers	Adders	Registers	RAMs	Multiplexers
Equalizer	80	156	120	0	4
Symbol decision	88	158	1282	0	42
Coefficients update	32	110	308	0	40

TABLE 2. Xilinx generated utilization figures for the same implementation. XC6VLX240T chip is the target FPGA.

Resource Type	Used	Available	Utilization	Bottleneck?
Slice Registers	8334	301,400	2%	No
Occupied Slices	4331	37,680	11%	No
Number of BUFG	5	32	15%	No
Number of DSP48E1	234	768	30%	Yes

For completely characterizing the implemented system the maximum clock speed of the design must be investigated as well. The Xilinx ISE tool used by Simulink™for generating the FPGA programming file (aka, the bitstream) provides this information. At the end of the so-called Place and Route

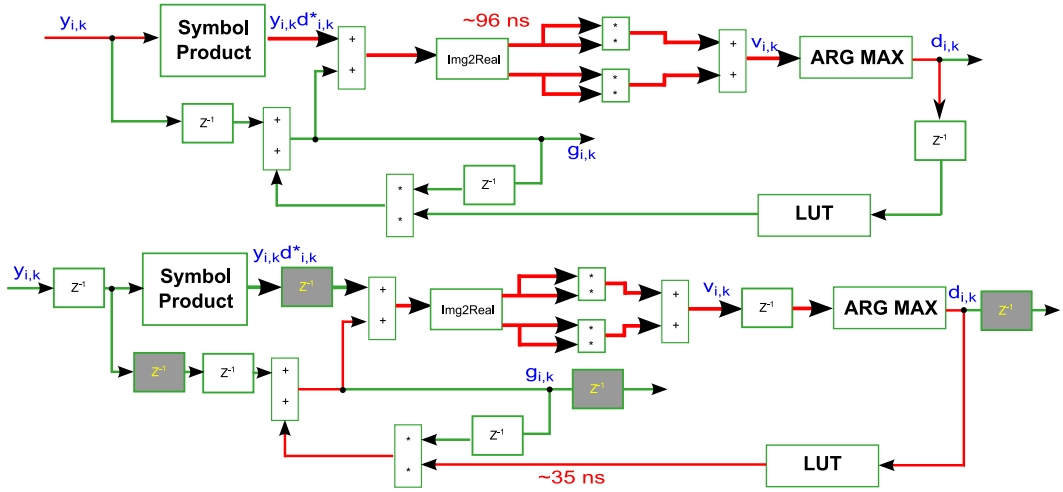
phase, in fact, it produces a table with all the timing constraints (the so-called *timing report*), specifying for each of them if they were satisfied or not.

Timing constraints are in a certain way similar to compiler directives: they provide a set of targets that the Xilinx tools should try to reach. More in details, they define the frequency of the input clocks and the relation between the internally generated clocks. For the same design, an increase of the input clock frequencies, in fact, leads to an higher system throughput. Timing constraints must be carefully selected, from a given value on, in fact, the tool will fail to find a configuration that satisfies the set of constraints. Failing paths (connections that do not respect one or more constraints) then will appear. Data transmitted along these paths cannot be considered reliable, meta-stabilities will likely occur. If performance are not sufficient, constraints have to be tuned and eventually the design modified for shortening these paths. The introduction of registers (so-called pipelining) can split these critical paths in shorter ones, eventually improving the maximum clock reachable by the implementation. As a drawback, the pipelining process introduces additional latency into the system, latency that may be not acceptable in certain circumstances.

Concerning the DSP receiver, the Xilinx tools identified as roughly less than 25 Mhz the maximum frequency of the input clock. It is worth noticing that all the internal clocks are generated from the input clock and thus it represents the only timing constraint of the system. Considering that standard designs can run on that specific FPGA chip at up to 200 Mhz, results were considered unsatisfactory and the root of this inefficiency was investigated.

The HDL coder came in handy also for this task: the tool in fact can parse the timing report generated by Xilinx, highlighting on the model the paths that are failing. As a matter of fact, the Symbol

FIGURE 2. Manual optimizations of the model. Pipeline registers (in gray) are introduced for shortening the critical path. Maximum clock rate passes from 20.8 Mhz (1000/(96 ns)∗**CLK_DIV**) to 57.1 Mhz (1000/(35 ns)*__CLK_DIV__) with **CLK_DIV** equal 2.



Decision block appeared to be the root of this problem. The introduction of a pipeline register represented a partial solution to the problem, bringing the maximum clock speed above 55 Mhz as depicted in Figure 2. Unfortunately, additional improvements will require the introduction of other pipelining stages that have proven to impair the correctness of this block. It appeared that, improvements in terms of clock speed could only be achieved using a different — not yet identified — Symbol Decision algorithm.

The implemented design can process up to 50 MSamples/sec correctly compensating the channel non-linearities and reconstructing the transmitted symbols sequence. Considering a samples rate of 50Gsamples/sec this result is still far from being exploitable (the number of FPGAs required will be excessively high). Fortunately, the FPGA chips are still far from being fully utilized and an additional step has been performed for increasing the system throughput. More details will be given in next section.

7.2. A Multi-Instances implementation

Table 7.1 has provided the amount of resources required for implementing the DSP receiver logic. For the sake of clarity, from now, a single DSP block (as described in the previous Chapter) will be referred as an *instance*. Given the amount of resources left unused in the chip, an approach based on multiple instances was considered. The idea was simple, every instance is fed with a subset of the input stream, producing a subset of the output stream that is reorganized before being outputted. By using **M** instances, if the system clock frequency is not affected the final throughput of the system will be **M** times higher than for the single instance approach.

In order to verify the feasibility of this approach, few aspects were investigated:

- *Input Stream partitioning*. The stream of samples must be correctly distributed among the various instances. Accordingly to the type of FFE implemented, two approaches must be taken. When the FFE is computed using the time-based convolution the input stream can be split into not-overlapping sets of samples and distributed to the various instances. Instead, if the frequency-based approach is exploited, the generated subsets must be partially overlapping for avoiding the introduction of artifacts due to the FFT operations. Not-exhaustive tests have shown that by using an overlapping window of **K**/5 (i.e. one fifth of the FFT size) samples, the system provides correct results.
- *FFE Coefficients update*. The FFE taps can be updated in two different ways: in a local fashion, implementing **M** version of the update block or in a two-tiers way, generating the update values using the contributions coming from all the instances. The first approach, while simpler to be implemented, is far from being optimal. The convergence speed, in fact, is now **M** times slower and in some occasions, the system had fail to converge to stable tap values. The second approach overcomes all these limitations and requires a limited amount of additional logic for being implemented. The coefficients update block, instead of updating directly the coefficients, proposes new values that are averaged between the **M** instances. The averaged values are then used for globally updating the coefficients. With this approach, the convergence speed has been less affected: even when an implementation with **M**=12 was investigated, the convergence speed was reduced of less than 50% .

- *Output stream reconstruction*. The outputs of the instances have to be sorted and eventually filtered for obtaining the correct sequence. The sorting process simply relies on Dual Port Memories, using a Double-Buffer approach for avoiding data losses. Each instance writes in the memory in a given address space and when this memory is filled (i.e. all the instances have filled their memory space) the data is extracted from the FIFO. While data are read, a second memory is written, performing a similar process. The process repeats ceaselessly. It is important to notice that when the frequency-based FFE is used the results generated from the overlapping window must be written only once for avoiding errors.

Figure 3 depicts the multi-instances implementation of the DSP receiver. The modified system has been validated relying again on the FPGA in the Loop feature provided by Simulink™. Given the amount of resources provided by the ML605 evaluation board used, up to three instances were instantiated. The final utilization of the chip is shown in Table 7.2. Concerning the throughput of this implementation, the maximum achievable clock rate by the system remains almost constant (roughly 54 MHz) and the final throughput becomes slightly more than 150 Msamples/s. Additional tests have been conducted implementing systems with up to 24 instances. Simulations have shown no significant results degradations or anomalies. Larger **M** values were not tested due to the excessive simulation time required (order of days).

Considering a full data rate implementation of the system, FPGAs characterized by more resources have been investigated. Table 7.2 lists some of them, highlighting the maximum throughput achievable. As a conclusive remark, it is worth pointing out that the automatically generated code is far from

FIGURE 3. Multiple instances implementation. The stream of samples coming from the ADC is split using FIFOs in **M** subsets. Coefficients are updated in a centralized fashion in order to guarantee the convergence speed of the coefficients.

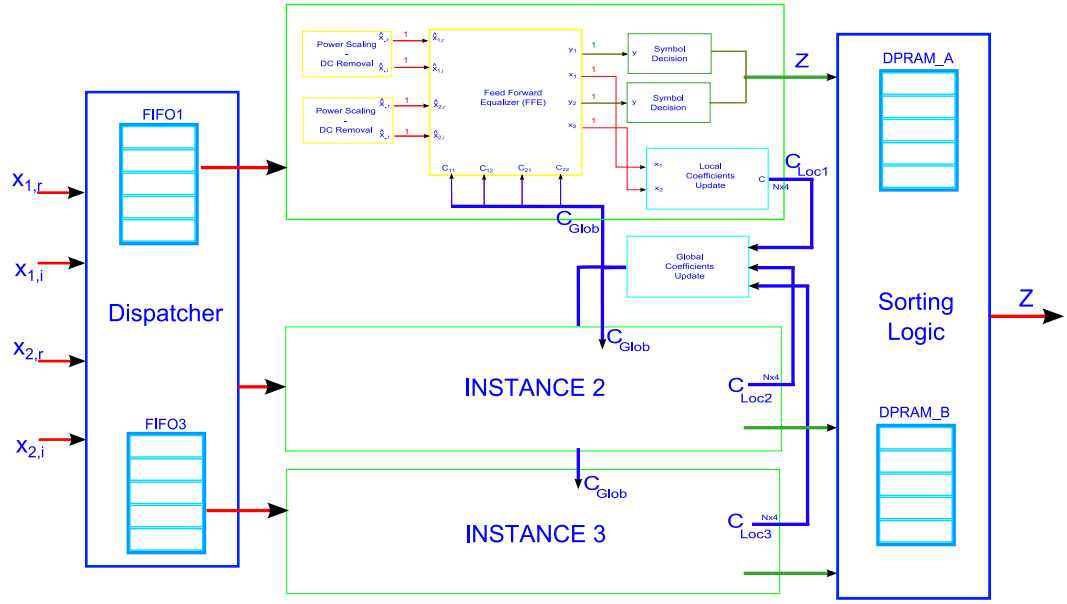


TABLE 3. Output of the Xilinx Place and Route tool for a system implementing three instances.

Resource Type	Used	Available	Utilization	Bottleneck?
Slice Registers	24461	301,400	8%	No
Occupied Slices	11479	37,680	30%	No
Number of BUFG	5	32	15%	No
Number of DSP48E1	742	768	96%	Yes

being optimal. Whenever an implementation is selected, the manual coding of some design blocks can significantly increase the maximum throughput achievable, by reducing the amount of resources required.

TABLE 4. Maximum throughput achievable using different Xilinx FPGA chips. Virtex-7 devices provide higher performance in terms of maximum clock rate than Virtex-6 ones.

FPGA Chip	Number of DSP48E1	Clock	Throughput
Virtex-6 XCE6VLX240T	768	>50 Mhz	150 Msamples/s
Virtex-6 XC6VSX475T	2016	>50 Mhz	450 Msamples/s
Virtex-7 XC7VX550T	2880	>65 Mhz	750 Msamples/s
Virtex-7 XC7VX980T	3600	>65 Mhz	900 Msamples/s

Part 3

Latency constrained systems

CHAPTER 8

Digital Signal Processing in High Energy Physics Experiments

IN a broad sense latency can be seen as the amount of time required to traverse a system. In the digital processing domain, latency indicates the processing time required to convert the provided inputs into the expected outputs. In a significant number of DSP-based applications, results are valid and correct only if they are provided to the user within a specific time window (i.e. the data processing has a deterministic latency).

High Energy Physics (HEP) experiments represent an extremely interesting and challenging example of latency constrained systems. As a matter of fact, almost all the HEP experiments investigate extremely rare physics events and, to reach sufficient confidence levels on their results, they impose the collection of massive amount of data. In large experiments, such as the ones conducted using the Large Hadron Collider (LHC) particles accelerator at the European Center for Nuclear Research (CERN), the experiments detectors can easily generate Terabits of data per second. For ATLAS and CMS — two of the four main experiments at CERN — collected data must traverse various processing levels before being stored in computer farms. Every level must decide in a fixed amount of time whether the actual physics event contains relevant pieces of information or not. In particular, hard real time constraints exist in the initial levels of the processing chain, where data contained in the detector buffers are expeditiously overwritten by new ones. Any processing deadline violation may impact the accuracy of the system, reducing the level of confidence on the generated results and thus increasing the time required to confirm the value of the investigated matter property.

In the proceeding, the research activity conducted for the upgrade of the ATLAS Level-1 Trigger will presented, providing a concrete example of latency constrained signal processing systems.

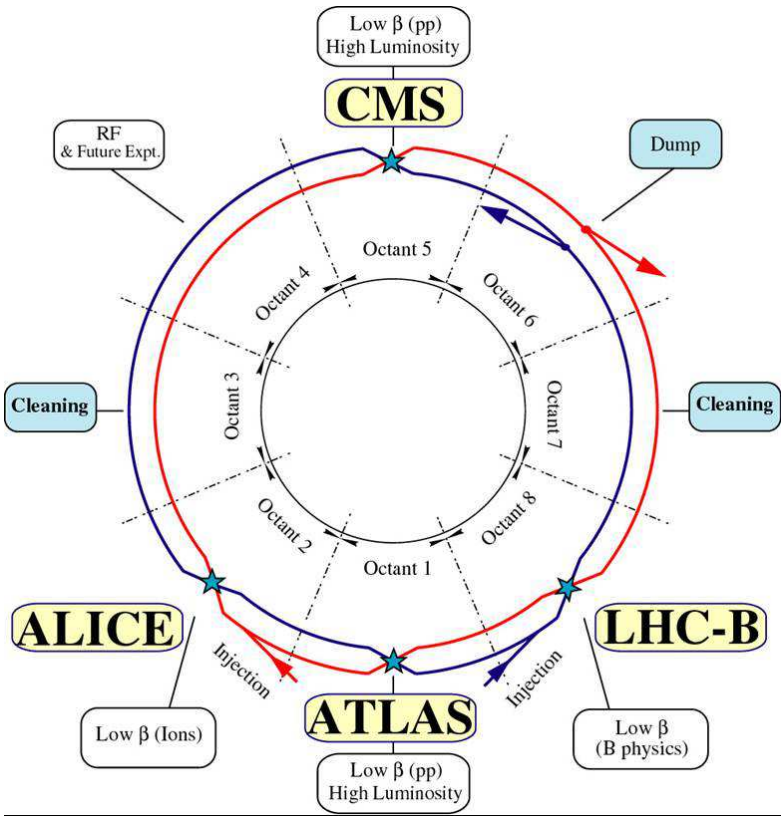
8.1. The LHC and the ATLAS Experiment

Nowadays, the Large Hadron Collider (LHC) accelerator represents the largest experiment ever built. Situated at CERN, it comprises different facilities in France and in Switzerland, with sophisticated machineries located inside and on top of a circular tunnel of 26.7 km of length, built 100 meters under the ground level.

Four artificial caverns contain the particle detectors used by the four main experiments: ATLAS [**Col99**], CMS [**Col94**], ALICE [**Col98**] and LHCb [**Col95**]. The ATLAS and CMS experiments (lately responsible for the discovery of the Higgs Boson) are referred as general purpose experiments, while the LHCb experiment is dedicated to B Physics and the ALICE experiment investigates heavy ion collisions.

Inside the LHC, every 25 ns two bunches of charged particles (protons or ions) are accelerated at up to the 99.9999991% of the speed of light and sent through the two directions of the circular

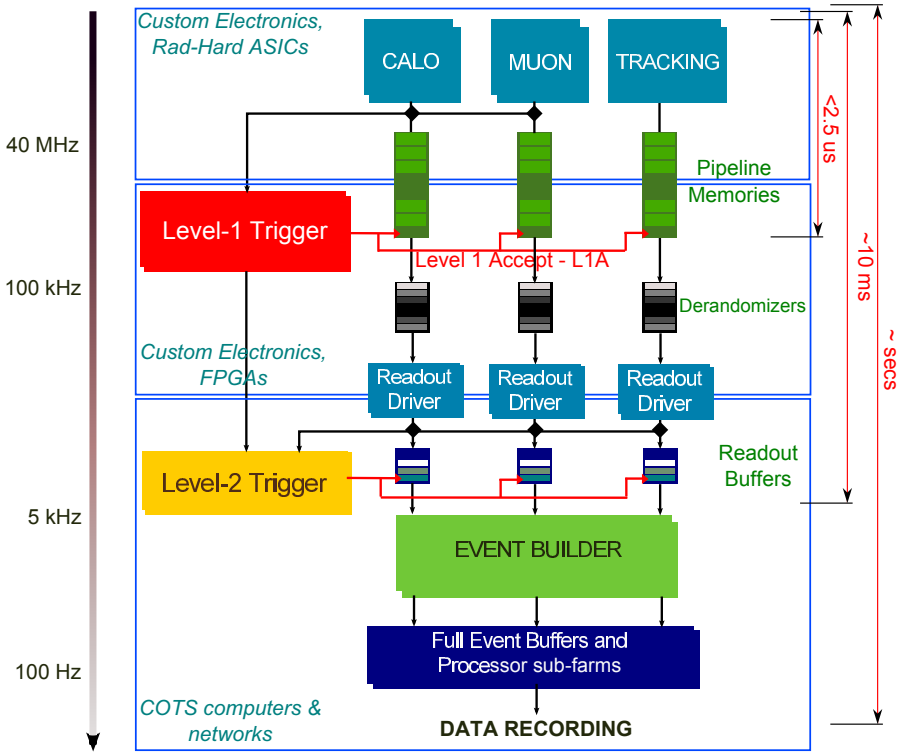
FIGURE 1. The Large Hadron Collider accelerator and the ALICE, ATLAS, CMS, LHCb experiments. *Courtesy of CERN.*



tunnel inside magnetic beam pipes. Collisions between particle beams are occurring inside the four experiment detectors, where different typologies of sensors are used to investigate different properties of the matter.

Focusing on the ATLAS experiment, the detector is composed of different sensors that are in charge of measuring the energies carried by the particles and of identifying the momenta of muons. Energies are measured by the Liquid Argon Calorimeter and by the Tile Calorimeter while the momenta of muons are characterized by the Thin Gap Chambers, the Resistive Plate Chambers, the Monitored Drift Tubes and finally by the Cathode Strip Chambers. These sensors readings are processed by so-called front-end electronics: custom devices (mainly ASICs and radiation-hard ICs) that have been designed for operating in this extremely radioactive environment. Energy readings are exploited by the calorimeter trigger processors to generate electron/photon, tau/hadron, and jet multiplicities and transverse energy information, while muons momenta are used by the muon triggers to produce the muon multiplicities. The so-generated multiplicities are used by the Central Trigger Processor (CTP) for discerning events of interest, implementing the first stage of the event filtering process (the so-called L1 Trigger). Whenever the CTP identifies an event of interest, it asserts a special signal, referred as Level-1 trigger accept (L1A), that is connected to the detector front-end memories via the so-called

FIGURE 2. The ATLAS Experiment, on-line data processing structure. Processing latency of the various components are depicted on the right hand side, while event rate is indicated on the left hand side.

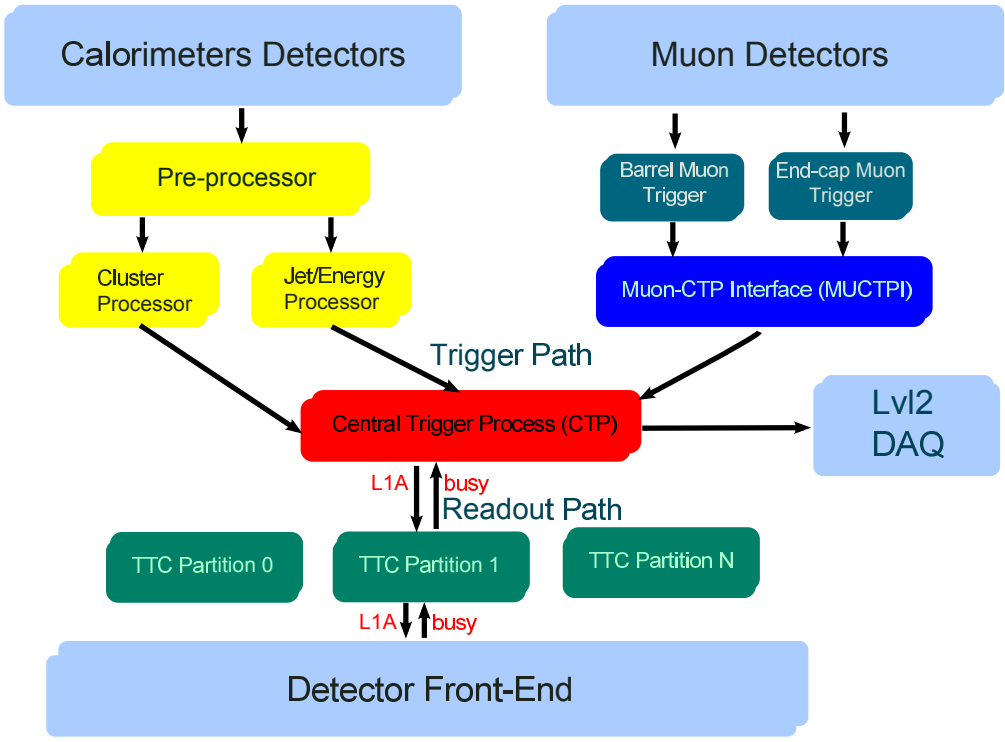


Time, Trigger and Control (TTC) link. The L1A signal initiates the readout of the memories that, due to their limited size, must be read before being overwritten by new readings. The L1 Trigger reduces the extremely high event rate (40 MHz) down to a much lower one (down to 100 kHz), sustainable by the Level-2 and DAQ systems. By means of commercial off the shelf components, these systems reduce the event rate furthermore (100 Hz) before sending them to the storage farms, where they will be accessed by the off-line processing software.

The full processing chain is depicted in Figure 2. It is worth noticing that the front-end electronics and the L1 Trigger are the most critical components in terms of latency. As a matter of fact, their design had to take into account the latency of links and of the digital processing logic in order to produce the targeted performance. More details on the front-end electronics can be found in [**Col96a**, **Col96b**, **Col96c**, **Col97**].

Next sections will present the activity conducted on the L1 Trigger and in particular on the Central Trigger Processor, focusing on the upgrade operations conducted during the first long shutdown (LS1) of the LHC.

FIGURE 3. The ATLAS Level-1 Trigger block diagram.



8.2. The Central Trigger Processor

The Level-1 trigger is a synchronous system that operates at the LHC Bunch Crossing (BC) frequency of 40.08 MHz. It reduces the event rate down to 100 kHz using the information on clusters and global transverse energy in the calorimeters and on tracks in dedicated muon trigger detectors for discarding event of no interest. See figure 3 for an overview of the ATLAS Level-1 trigger system.

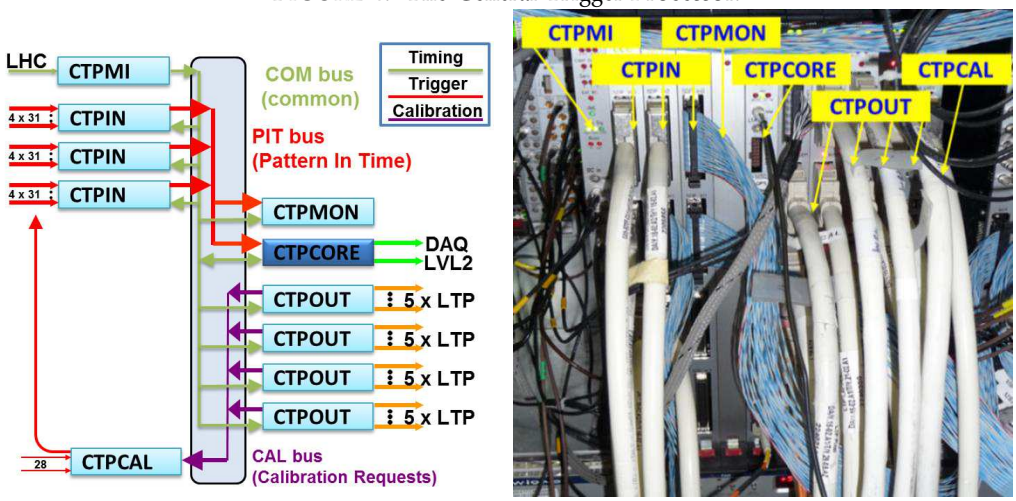
The largest part of the signal processing is carried out by the Central Trigger Processor (CTP) that primarily makes the final Level-1 trigger accept (L1A) decision using the following pieces of information:

- Electron/photon, tau/hadron, and jet multiplicities;
- Transverse energy information from the calorimeter trigger processors;
- Multiplicities from the muon triggers;
- Luminosity detectors, minimum bias scintillators and beam pick-ups.

The L1A decision must be formed within 5 Bunch Crossings (BC) to avoid Front-End buffers being overwritten by new data. In other words, the L1A signal must be irremissibly generated by the CTP logic in less than 125 ns.

The trigger items used to generate the L1A can eventually be masked and pre-scaled by the CTP as a function of the LHC bunch structure. Furthermore, the CTP can introduce preventive dead-time, effectively protecting the front-end buffers of the experiment from becoming full.

FIGURE 4. The Central Trigger Processor.



(a) Diagram of the CTP (b) CTP: installation in USA15

The CTP is in charge of distributing timing signals (received from the LHC), together with the L1A, through the so-called Trigger, Timing, and Control (TTC) network. It also sends trigger summary information to the Level-2 trigger (LVL2) and data acquisition (DAQ) systems. In addition, the CTP implements monitoring functionalities, such as trigger counters, as well as, dead-time and busy information for luminosity and background monitoring. A more detailed analysis can be found in [ea08].

The CTP is host in a 9U VME64x [www98] chassis where three dedicated backplanes connect eleven custom designed modules of six different types. The architecture of this module is depicted in Figure 4. The main elements composing this system are:

- *The machine interface module (CTPMI).*
  This unit receives the timing signals from the LHC and distributes them over one of the dedicated backplanes (referred as common, COM).
- *The input module (CTPIN).*
  Three CTPIN modules receive the trigger input signals that are synchronized, aligned, selected and finally sent over the Pattern In Time (PIT) backplane to the monitoring and core modules.
- *The core module (CTPCORE).*
  The CTPCORE is in charge of generating the Level-1 accept signal (L1A) according to the programmable trigger menu. Furthermore this module has to transmit trigger summary information to the Level-2 trigger and the DAQ system.
- *The monitoring module (CTPMON).*
  The CTPMON monitors the the PIT signals in a bunch-by-bunch fashion.
- *The output module (CTPOUT).*
  Four CTPOUT modules are in charge of distributing the trigger and timing signals to the

sub-detector TTC partitions. They also receive calibration requests and busy signals from the sub-detectors.

- *The calibration module (CTPCAL).*
  The CTPCAL module multiplexes the calibration request signals sent over the calibration backplane (CAL). Additionally, this block performs level-conversion over a set of trigger inputs.

8.3. CTP upgrade: motivations

During the 2013-2014 LHC shutdown machines and cables will be consolidated and some components will be upgraded for coping with the foreseen increase in the event energy (from 8 TeV to 14 TeV). An extensive description of the planned upgrades can be found in [**col11**].

Concerning the Level-1 trigger various upgrades are required, with a significant part of them involving the CTP. Modifications to the CTP are necessary to interface a new topological processor (L1Topo, see [**Sim12**]) — introduced for improving multi-object selection — and to handle a larger number of trigger inputs and trigger items.

TABLE 1. Level-1 Trigger. Resource utilization summary.

Resource	Used	Available
CTPIN input cables	9	12
CTPIN input signals	212	372
CTPIN integrating monitoring counters	138	768
PIT bus lines	160	160
CTPCORE trigger items	241	256
CTPCORE bunch group masks	8	8
CTPCORE per-bunch trigger item counters	12	12
CTPOUT cables to TTC partitions	20	20
CTPMON per-bunch monitoring counters	88	160

Considering the current CTP, this system is close to full utilization, as shown in Table 1. As a matter of fact, all of the PIT bus lines and almost all of the trigger items are currently used. The upgrade addresses these limitations, increasing the number of trigger inputs and the number of trigger items.

In addition, after the upgrade, the CTP will provide partitioning of the L1A generation for detector commissioning, an improved group masking and bunch-by-bunch trigger item monitoring, and more outputs for sub-detector TTC partitions.

Even after the upgrade, the total round-trip latency will not change: 2.5 μs of which about 125-175 ns (5-7 BC) allocated to the CTP. Concerning the L1Topo module, it will transmit the topological information to the CTP (in the specific to the CTPCORE) via two alternative links: an electrical (lower

latency) and optical ones (higher throughput). The former will potentially be adopted in the future if its latency will prove to be compatible with the allocated headroom.

The CTP upgrade imposes the redesign of the CTPCORE, the CTPOUT, and the COM backplane. After this upgrade, these components will likely remain unchanged up to the next major upgrade, foreseen for 2022/24. In the proceeding the CTPCORE upgrade process will be presented.

CHAPTER 9

The Upgrade of the Central Trigger Processor Core Module

AS introduced in the previous Chapter, the Central Trigger Processor (CTP) must be upgraded for operating at the higher energy and luminosities targeted for the post-LS1 [**col11**]. Compared to the actual CTP, the upgraded system will provide twice the number of trigger inputs (from 160 to 320) that will be transmitted using the existing PIT bus backplane operated in double-data-rate (DDR) mode. Additional inputs will be received via 96 LVDS inputs (operated in DDR mode) and via 12 serial optical links. These additional signals will be used by the upgraded CTPCORE, the CTPCORE+, for enhancing the extending trigger selection capabilities.

Compared to existing CTPCORE, the CTPCORE+ will have twice the number of trigger items for the programmable trigger menu, thus increasing the existing quota from 256 to 512. Three trigger generation partitions will be included, with each of them having a different selection of trigger items and its own dead-time handling. Only the primary or physics partition will provide output to the Level-2 trigger and DAQ, while the two remaining secondary partitions will be used for detector commissioning, calibration and other tasks. Monitoring functionalities will be enhanced, with 16 bunch groups for bunch-by-bunch masking of the trigger items (cf. 8 today) and 256 bunch-by-bunch counters for trigger item monitoring (cf. 12 today).

As for the existing CTP, the Level-1 Accept (L1A) signal generation will remain latency constrained, with only a small increment in the total latency tolerated (from 5 to 7 BC).

In the proceeding, the upgraded CTPCORE (namely the CTPCORE+) will be described, focusing on the design of the hardware and firmware components.

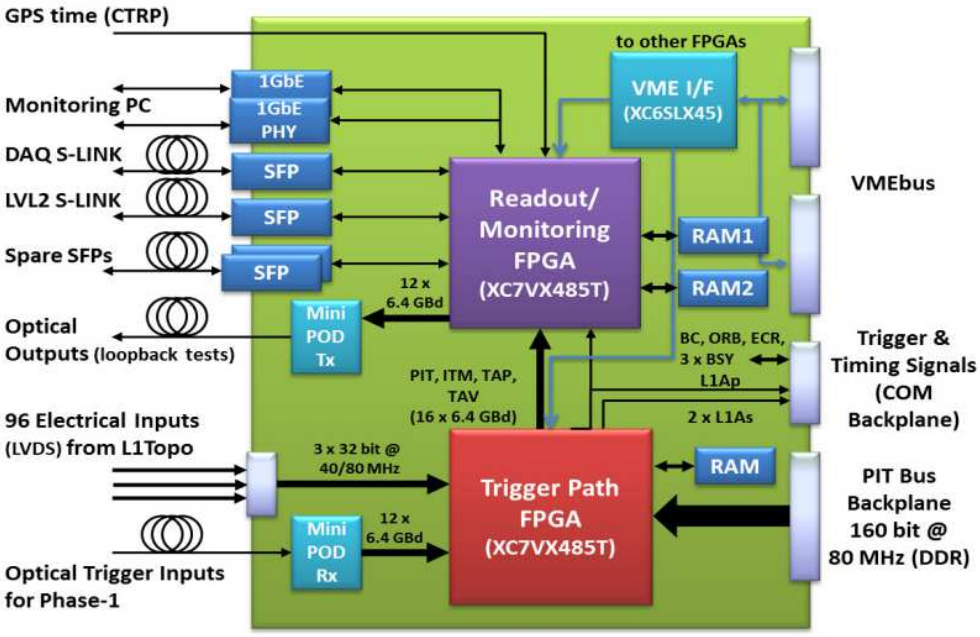
9.1. The CTPCORE+ module

The CTPCORE+ board must implement complex digital processing operations under strict hard real-time constraints. To implement the foreseen functionality, the CTPCORE+ module must provide large processing capabilities as well as massive chip-to-chip connectivity. As shown in Figure 1, the main components of this board are two large Xilinx Virtex-7 FPGAs and a smaller Xilinx Spartan-6.

Virtex-7 chips are used by the so-called Trigger Path FPGA (TRG) to implement the trigger path logic and by the Readout and monitoring FPGA (RDT) to provide the readout and monitoring functionalities. The same chip model, the XC7VX485T, is mainly used by both TRG and RDT.

The XC7VX485T is a complex device featuring 300 k 6-input look-up tables (LUT), over 1000 RAM blocks of 36 kbit each and 20 multi-gigabit transceivers (MGT), thereby providing ample resources to implement the required functionality. The Spartan-6 chip instead is used to interface the processing FPGAs to the VME bus for configuration and control.

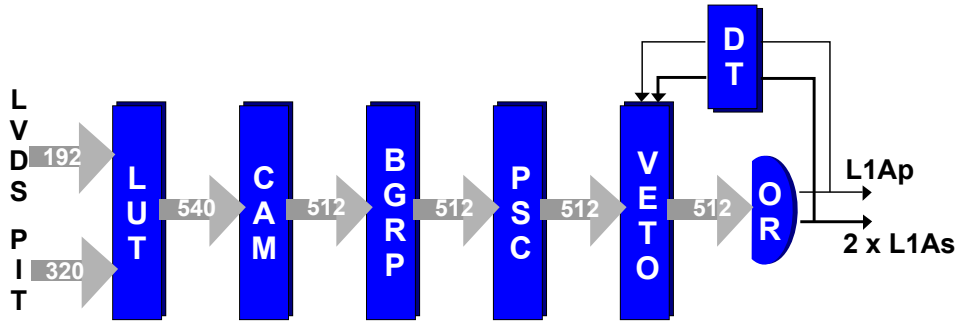
FIGURE 1. CTPCORE+ board, block diagram



The TRG FPGA implements all of the latency critical real-time data path functionality. It receives up to 320 trigger inputs from the CTPIN modules through 160 PIT bus lines operating at 80 MHz. Additional 96 LVDS electrical inputs from a connector on the front-panel of module will interface the module to the topological trigger processor. Finally trigger inputs can be optionally received through 12 serial optical links operating at 6.4 GBaud using a 12-way ribbon fiber receiver module. However the serialization and deserialization implemented by the Multi-Gigabit Transceivers introduces a latency penalty of about 3 BCs that needs to be considered. High-density optical receiver modules (Avago MiniPOD) provide a compact solution for implementing the optical-electrical conversion. These modules can be placed close to the FPGA, thereby reducing the required line length for the critical high-speed signals and easing potential signal integrity issues.

Concerning the functionalities, the trigger path FPGA generates the three L1A signals (1xL1Ap from the primary and 2xL1As from the secondary partitions) and the associated trigger type word. As depicted in Figure 2, 320 trigger input signals are received and synchronized from the PIT backplane. Additional 192 bits are received via the 96 LVDS DDR inputs for a total of 512 bits. After being combined in an array of Look-Up Tables (LUT), these 512 bits are used by a large ternary Content-Addressable Memory (CAM) to form 512 trigger items. The trigger items are then masked by the bunch group mask (BGRP) and individually pre-scaled by fractional scalers (PSC). The outputs of the pre-scalers (TAP) are gated with a programmable mask, dead-time (DT) and busy signal (VETO) before they are ORed to generate the L1A signals. The preventive dead-time (DT) is used to prevent the front-end buffers from overflowing. The three L1A trigger generation partitions share the LUT, the CAM and the pre-scaler resources, with independent veto logic, dead-time generation and logical OR. In

FIGURE 2. The L1A trigger generation (the so-called trigger path).



addition, the TRG must send about 2300 bits per BC containing information about the trigger decision to the RDT FPGA. The required bandwidth of roughly 92 Gbit/s is provided by 16 on-board high speed serial links which operate at a line speed of 6.4 GBaud with 66B66B encoding.

The RDT FPGA implements all the non-latency critical functionality such as readout and monitoring. Upon a L1A, Region of Interest information and trigger summary information are sent to the Level-2 trigger and DAQ systems through two serial optical readout links operating at 2 GBaud. The data contains the 512 trigger inputs as well as the 512 trigger items both before and after masking and pre-scaling (refer to Figure 2) for each bunch in a programmable window. It must be noted that only the primary partition can send data to the Level-2 and DAQ systems. A GPS timing reference is received from an external card (CTRP) via the General Machine Timing (GMT) system of the LHC. This latter is used to generate a precise time-stamp that is associated to each event. The RDT FPGA also implements advanced monitoring features, with 1800 integrating counters used for monitoring the trigger rates and 256 counters used for building histograms of selected trigger items as a function of the bunch number. To implement all these monitoring functionalities up to 50% of the available on-chip block RAM resources are required.

The CTPCORE+ module also includes various test and diagnostics features. The TRG and RDT FPGAs interface to DDR3 memories (1 module for the TRG, 2 modules for the RDT) which can be used for validating the system. The correctness of the optical inputs can be validated relying on a ribbon fiber transmitter module that can be loop-backed to the receiver module.

9.2. Design validation: the Demonstrator setup

The CTPCORE+ module is a state-of-the-art system that must operate reliably for a significant long period of time requiring limited maintenance.

The system comes with a complex design characterized by a convoluted power distribution network (large number of related voltage rails) and by components (the Virtex-7 chips used for the TRG and RDT FPGAs) that only recently entered into a full production stage. Hence, to validate various aspects of the design, feasibility studies were carried relying on a so-called Demonstrator.

The Demonstrator is composed of two Xilinx commercial evaluation boards (VC707, see [**www14p**]), connected via fast electrical links for resembling the final CTCPORE+ board structure, as depicted in Figure 3. More in detail each VC707 board contains:

FIGURE 3. The Demonstrator used for validating the CTPCORE+ design.



- A XC7VX485T Virtex-7 chip, equivalent to the two chips used for the TRG and RDT FPGAs.
- An external SODIMM DDR3 memory,
- Texas Instruments UCD9248 [9] DC/DC controllers, with control and status registers, that can be accessed via a Power Management Bus (PMBus) interface. Via PMBus, instantaneous measurements of current and voltage levels for the different power rails can be retrieved.
- An Ethernet PHY adapter connected to the FPGA via a Serial Gigabit Media Independent Interface (SGMII) interface.
- Two Full Mezzanine Card (FMC) connectors exporting a total of 16 MultiGigabit Transceivers (8 per each connector). Four FMC cards from Faster Technology (FMS-28, see [**www14k**]), together with high-speed electrical cables have been used for connecting the two boards, mimicking the PCB traces of the final CTPCORE+ board.

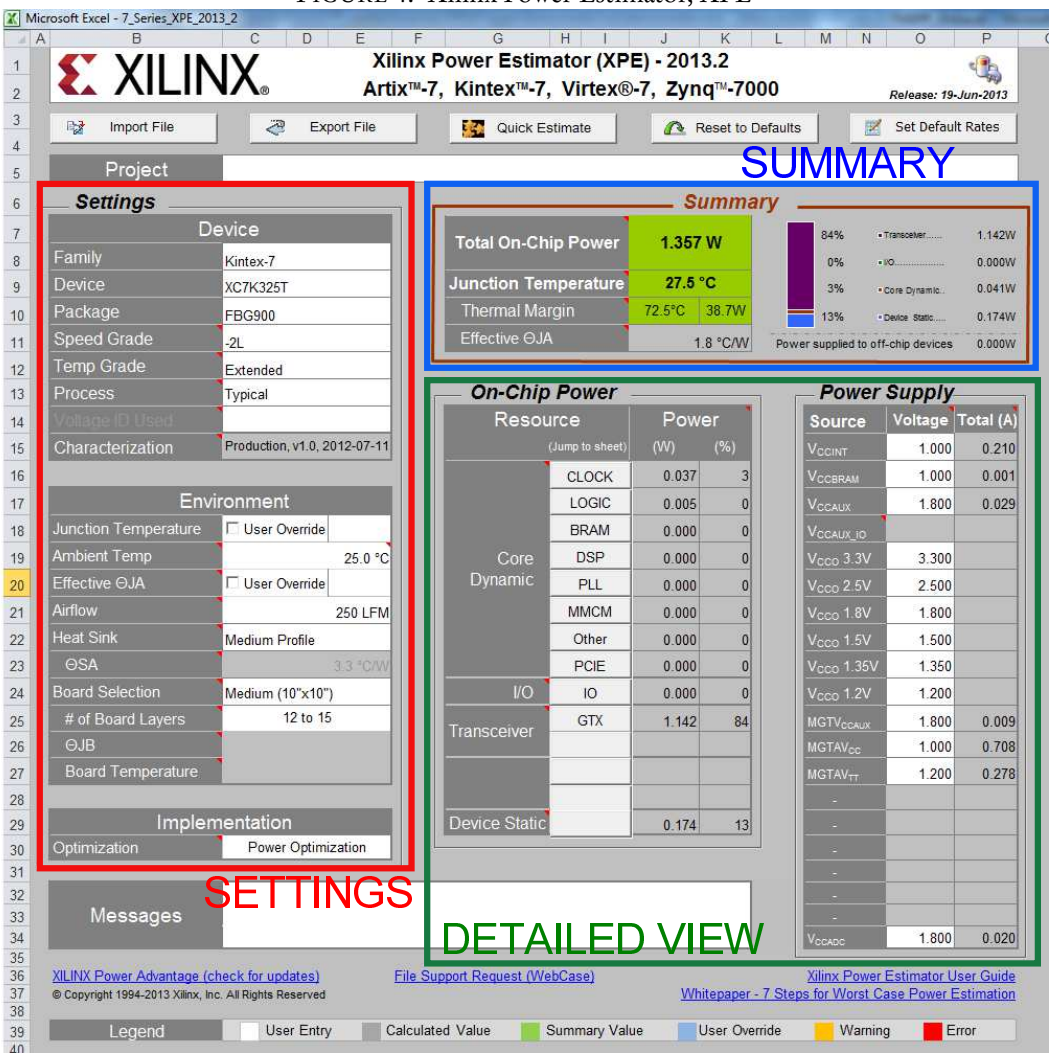
In the proceeding the activities conducted for validating the CTPCORE+ board will be presented. The design and validation of the firmware blocks will be described in the next Chapter.

9.3. The Power Distribution Network

To operate correctly, the CTPCORE+ module Power Distribution Network (PDN) must provide accurate and stable tensions to the different components. The Virtex-7 FPGA chips, in particular, require almost ten different low-ripple rails and a specific power-up sequence. Furthermore, the knowledge of the maximum current associated to each voltage rail is fundamental for the design of an efficient and reliable PDN.

Even though the Xilinx documentation provides specifications for the power-up sequencing and for the operating ranges of the various components, the design of the power distribution network imposed extreme care.

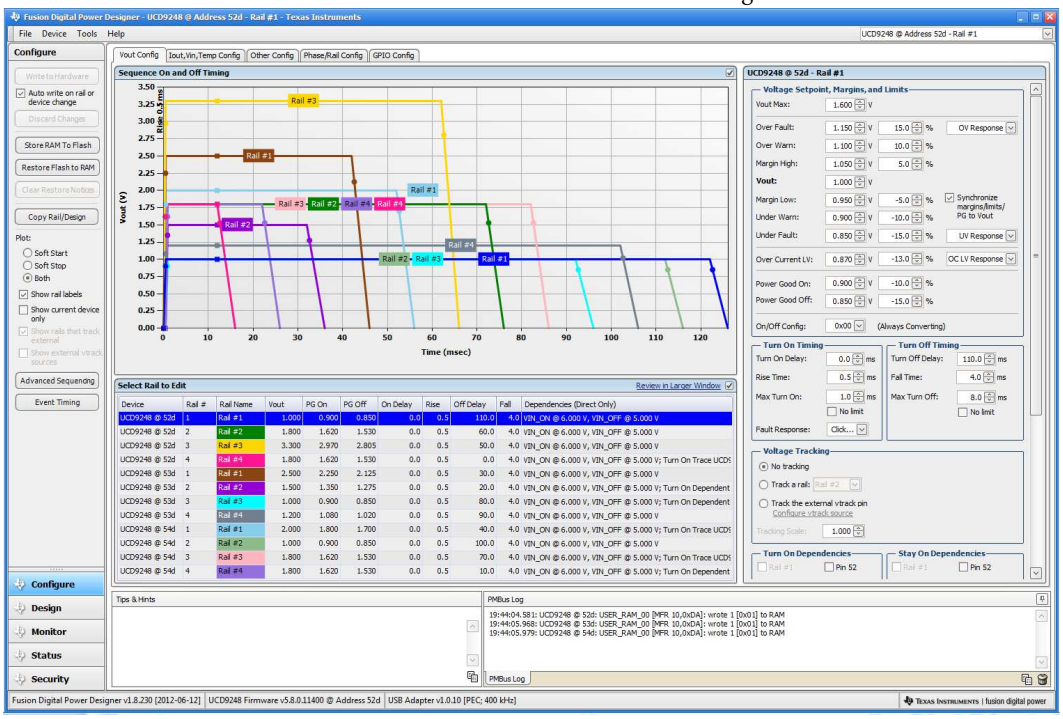
FIGURE 4. Xilinx Power Estimator, XPE



In fact, the current required by the different FPGA blocks is extremely variable and related to their specific configuration and toggling activity. Preliminary estimations can be made via proprietary analysis tools such as the Xilinx Power Estimation (XPE) one. The XPE tool forecasts the current consumption of the various components taking into account the specific chip, the operating temperature, the status of the block (i.e. enabled or in power down state) and finally the frequency of the clock used by the different blocks. The tool lists aggregated values for the dynamic and static power consumptions, providing as well a detailed consumption per each voltage rail as shown in Figure 4.

To minimize the probability of making wrong design assumptions, the consumption figures generated by XPE have been validated and verified relying on real measurements conducted on the Demonstator setup.

FIGURE 5. Texas Instruments Power Designer



The Texas Instruments Power Designer tool provides an easy to use interface for monitoring the status of the controllers as well as for reconfiguring them. Figure 5 shows the output measured for one of the controllers.

To fairly compare the results, the internal registers of the UCD9248 DC/DC controllers have been queried after having configured the demonstrator FPGAs with different firmwares. The XPE tool has been set-up to mimic the configuration of the FPGA used in the various firmwares (i.e. setting up the logic clocks frequencies, the number of MGTs used and their baud-rates and the eventual DDR3 memories clocks).

Considering the tests conducted, the values calculated by XPE appear to be generally conservative, with the estimated values 5-20% higher than the measured ones. However, for one of the MGT supply rails XPE underestimates the power consumption by about 200%, potentially leading to PDN design errors.

9.4. High speed links

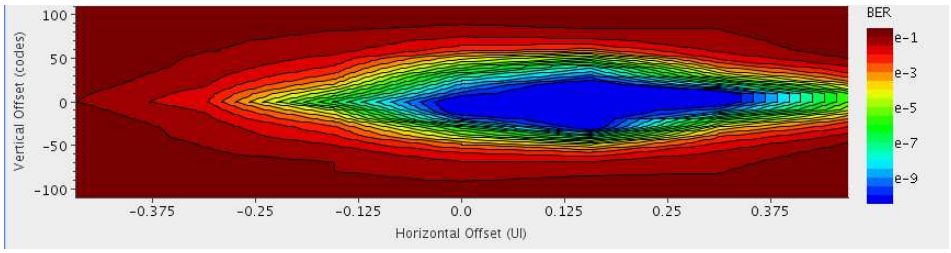
As previously stated, every 25 ns the TRG FPGA must transmit to the RDT FPGA roughly 2300 bits of trigger summary information, corresponding to a bandwidth of 92 Gbps. In the CTPCORE+ module 32 MGTs (16 per each side) operating at 6.4 Gbps will be used for this purpose.

In order to verify the feasibility of this approach, the two evaluation boards have been connected through four FMC mezzanine cards (FMS-28 from Faster Technology [**www14k**]) and four high speed electrical cables as shown in Figure 3. The FMS-28 FMC card provides two QSFP sockets, each of them

FIGURE 6. Chip-to-chip links quality assessment.



(a) Link Bathtube Analysis



(b) Link Eye Diagram Analysis

connected to 4 MGTs of the FPGA. In the Demonstrator setup, high speed electrical cables designed for 10 Gbps are used to mimic the PCB traces connecting the FPGAs on the CTPCORE+ module.

The Xilinx Integrated Bit Error Rate Tool (IBERT) was used for measuring the channel Bit Error Rate (BER) and to generate the bathtub curve and the eye diagram. A BER of better than 10^{-15} was measured transmitting a 31 bits Pseudo Random Bit Sequence (PRBS-31) at 10 Gbps. Concerning the bathtub curve, the IBERT tool leveraged the internal measurement capability of the Virtex-7 chips to generate Figure 6.

The measured BER and the generated plots were considered a valid proof of the CTPCORE+ chip-to-chip high speed link feasibility. In fact, the tests conducted on the Demonstrator exercised more the links, running at an higher baud-rate (10 Gbps instead of 6.4 Gbps) and over a much longer distance (1m instead of 10 cm).

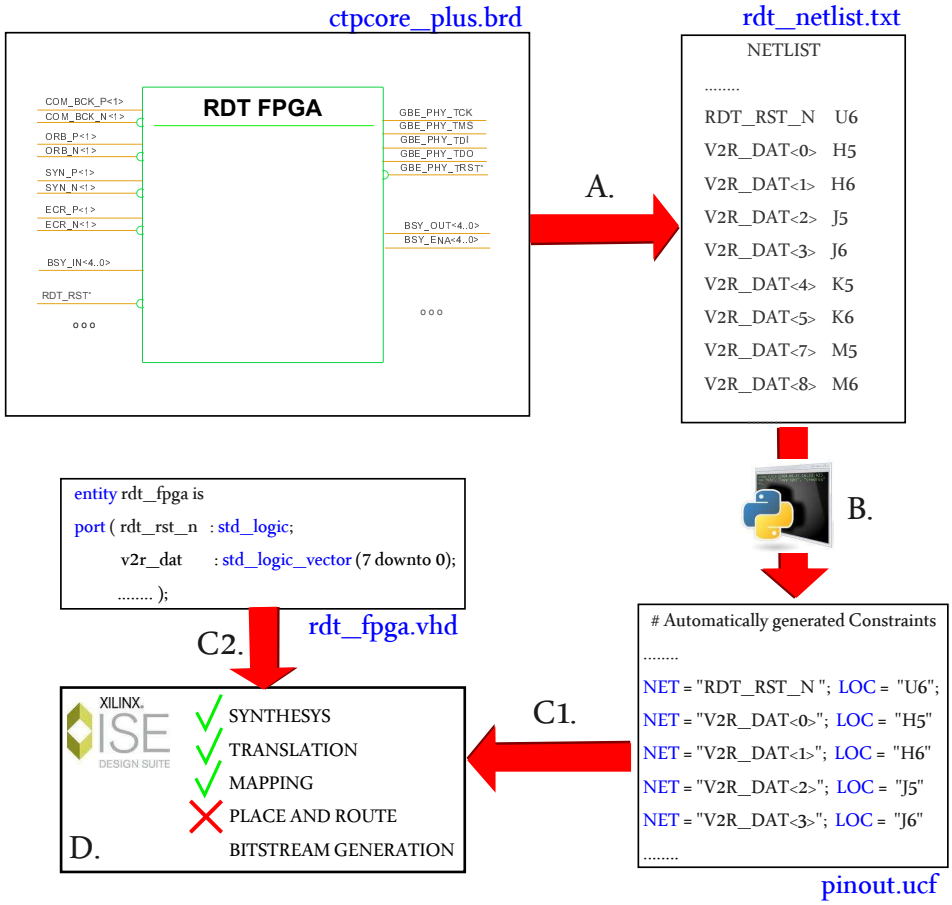
9.5. Automatic pinout verification tool

The Virtex-7 FPGA chip selected for the CTPCORE+ board, the XC7VX485T-1157 comes with 600 user assignable I/O pins. For each link, designers should select the appropriate I/O standard and pin location in order to maximize signal integrity and minimize signal switching noise ([**Bog10**]). Additional complexity comes from fast interfaces/buses, such as Multi Gigabit Transceivers and DDR3 that impose additional rules on the pinout.

For the CTPCORE+ board, the time required for manually validating the pinout was considered prohibitive and thus a semi-automatized procedure was deployed to verify the pinouts of the three FPGA chips. The procedure relies on various steps:

- *Netlist extraction and pinout generation.*
  A text-only netlist is generated by the Schematic Entry software (Cadence Allegro [**www14g**]).

FIGURE 7. Pinout validation process. The schematic entry software extracts the TRG, RDT and VME netlists (**B**). A python script converts the netlist into Xilinx location constraints (**B**). Finally, the pinout constraints (**C1**) and a minimal functionality design (**C2**) are passed to the Xilinx compilation tool-chain (**D**). In this way, the Synthesis and Place and Route tools will identify design errors, such as wrong pin location or forbidden I/O standards.



A script parses the netlist and associates the physical pin name to each symbolic name of the TRG, RDT and VME FPGAs. The generated pinout map is defined using the Xilinx directives and thus it is fully recognized by the synthesis and Place & Route tools.

- *Minimal functionality design.*
  A minimal functionality design is used for verifying the pinout correctness. The design contains the same I/O interfaces as the final CTPCORE+, providing a reliable way for verifying the correctness of the connections in terms of locations and I/O standards.
- *Place & Route and bitstream generation.*
  Potential pinout errors are identified running the Xilinx tools on the testing design. Misplaced

clock pins, sub-optimal connections, wrong I/O standards and not assigned pins can be easily identified by the tools and traced back to the schematic.

Figure 7 summarizes this process. As an additional benefit, the minimal functionality design can be adopted as initial firmware skeleton that can be gradually expanded integrating the various functionalities.

A more in-depth analysis of the firmware design process will be presented in the next Chapter.

CHAPTER 10

Firmware design

The Demonstrator setup was successfully used to validate the design of the CTPCORE+ board. Furthermore, it was largely exploited to design, debug and validate the firmware components required by the upgraded unit. Conducting in parallel the firmware and the hardware design saved a significant amount of time and helped identifying various architectural limitations that were correctly addressed in the early design stages.

This Chapter will present the main CTPCORE+ firmware blocks designed leveraging the Demonstrator setup.

10.1. Monitoring and Control Interface

Since the demonstrator setup does not come with a VME interface, a different control interface had to be used to emulate the VME bus interface available on the CTPCORE+ module. IPBus [**www11e**], a UDP based control and monitoring protocol, was identified as a valid alternative to the VME bus. IPBUs can be used for accessing the internal registers of the FPGA providing, from the software side, simple register read and write operations as the VME bus. Table 10.1 lists the main IPBus operations.

TABLE 1. Main operations supported by the IPBus protocol.

Operation	Explanation
SingleWrite	Writes a single 32 bits value in a location pointed by a 32 bits address
SingleRead	Reads a single 32 bits value from a location pointed by a 32 bits address
BlockWrite	Writes **N** 32 bits values in **N** memory contiguous locations
BlockRead	Reads **N** 32 bits values from **N** memory contiguous locations
FifoWrite	Writes **N** 32 bits values in a single memory location
FifoRead	Reads **N** times a 32 bits value from the same memory location

Furthermore, the IPBus protocol has been already adopted by the CMS experiment for current upgrade projects and it is being considered for future ATLAS upgrades. The IPBus FPGA firmware has been ported to various architectures, with a version for Virtex-7 FPGAs only recently released (IPbus 2.0). For the Demonstrator, the IPBus 1.4 release was extended to support the Virtex-7 architecture, providing two PHY flavours, SGMII and 1000-BASE-T.

10.2. SDRAM Memory Controller

The demonstrator and the CTPCORE+ use DDR3 memories as playback and snapshots memories. To implement these functionalities, the memories must be accessible from two sides: internally from the FPGA logic and externally from the Control and Monitoring Interface. The internal logic accesses the memory synchronously to the BC, demanding a constant and guaranteed bandwidth, while the Control and Monitoring Interface accesses the memory sporadically and has relaxed requirements in terms of bandwidth and latency.

Being these two accesses concurrent, additional control logic had to be include to implement a virtual dual port memory controller.

Such a memory controller embeds the standard external memory controller IP Core provided by Xilinx and leverages a custom made access scheduler that implements a fixed priority scheduler with additional anti-starvation logic. The main aspects of the implemented logic can be identified by the following pseudo-code block that describes the scheduler.

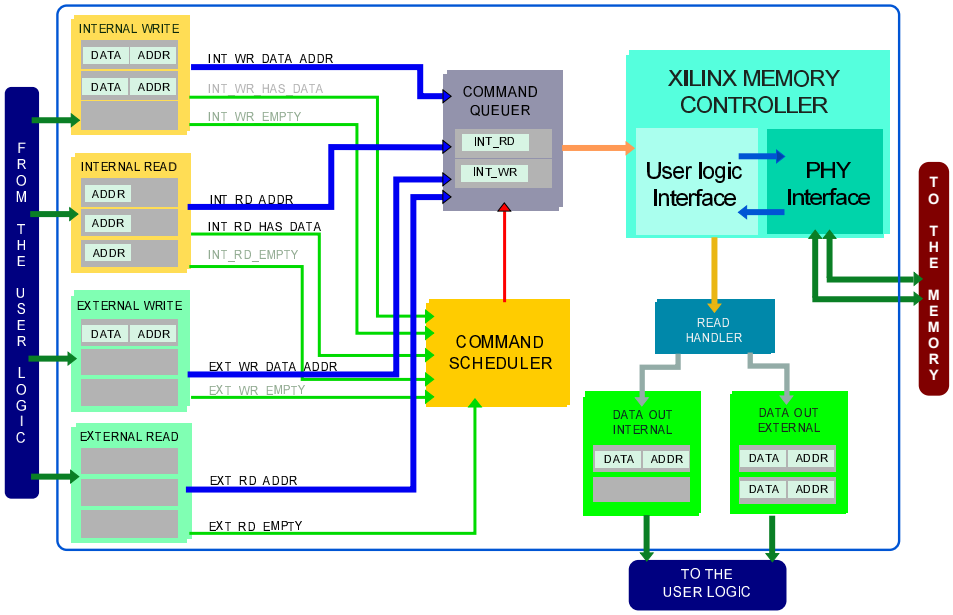
```
6         ...
7         // Starvation avoidance
8         if (not EXT_WR_EMPTY) {
9                 EXT_WR_STARV_CNT++;
10        }
11        if (not EXT_RD_EMPTY) {
12                EXT_RD_STARV_CNT++;
13        }
14        // Next command generation
15        if (EXT_WR_STARV_CNT > STARVATION_THR) {
16                CMD                 = EXT_WR;
17                EXT_WR_STARV_CNT    = 0;
18        } elsif (EXT_RD_STARV_CNT > STARVATION_THR) {
19                CMD                 = EXT_RD;
20                EXT_RD_STARV_CNT    = 0;
21        } elsif (INT_WR_HAS_DATA) {
22                CMD = INT_WR;
23        } elsif (INT_RD_HAS_DATA) {
24                CMD = INT_RD;
25        } elsif (not INT_WR_EMPTY) {
26                CMD = INT_WR;
27        } elsif (not EXT_WR_EMPTY) {
28                CMD = EXT_WR;
29        } elsif (not EXT_RD_EMPTY) {
30                CMD = EXT_RD;
31        } else {
32                CMD = NO_OP;
33        }
34        ...
```

While figure 1, instead, provides a graphical overview of the full SRAM firmware block.

It is worth noticing, that due to the data paths existing between the memory, the BC and the control clock domains, dual clock FIFOs have been extensively used on the clock domain boundaries to avoid metastability. Roughly less than 10 percent of the available Block Rams resources have been used to implement this feature.

The firmware correctness and performance have been investigated on the demonstrator system, operating the interface at the maximum supported line rate of 1.6 Gbps.

FIGURE 1. Read and write requests from the internal and external logic are initially accumulated into domain crossing FIFOS. Successively, the scheduler selects the next operation to be queued. The Xilinx Memory Interface IP receives the command requests and executes them. Data read back from the memory are sent to the correct requesting interface (INT or EXT).



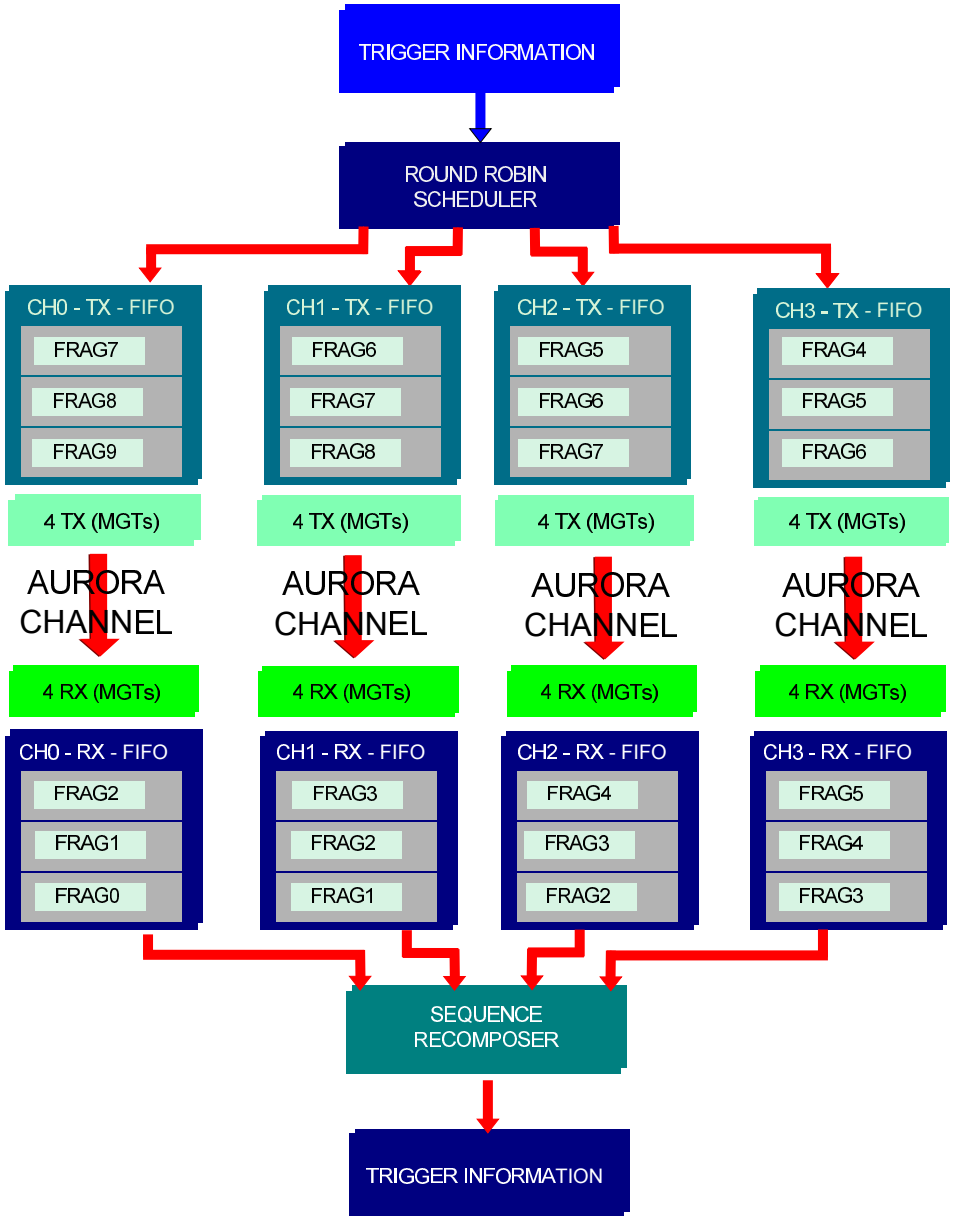
10.3. Chip to Chip communication protocol

In the CTPCORE+ board, being the TRG and RDT FPGAs the most consuming devices, their correct configuration represents the most effective way to minimize the overall total power consumption and thermal dissipation. From the tests conducted on the Demonstrator setup, the on-board MultiGigabit Transceivers (MGT) represent one of the most consuming part of the chip. As a results, significant savings in terms of total power consumption can be achieved operating the MGTs at the lowest speed possible, thus minimizing the drained current. Link speed can be reduced by maximizing the link efficiency, or in other words, by minimizing the transmission overhead.

Due to the minimal transmission overhead penalty that it introduces, the Xilinx proprietary Aurora64b66b protocol was selected for the chip to chip communication link. The Aurora64b66b IP core takes also care of setting-up, synchronizing and verifying the status of multiple MGTs, considerably reducing the amount of firmware to be designed. Furthermore, the so-called Aurora channels can be configured to run at different baud rates, aggregating a different number of MGTs (up to 16 MGTs per Aurora channel).

Additional control logic has been developed on top of the Aurora protocol in order to maximize channel efficiency independently by the size of the transmitted data. Groups of 4 MGTs are connected to a round-robin scheduler that sends and reconstructs data in the correct order. More in details, the trigger information (currently 2308 bits) is split into 9 blocks of 256 bits (fragments) and transmitted

FIGURE 2. Chip to chip communication protocol. A Round Robin Scheduler distributes the Trigger information to the different Aurora channels. Each channel splits the received data into fragments of 256 bits that are transmitted via the 4 grouped MGTs. On the receiver side the opposite process takes place.



from the TRG to the RDT, where it is reconstructed. Figure 2 shows a simplified diagram of this process.

The implemented firmware can be customized to run in two different configurations: a 4 channels one for the CTPCORE+ board(16 MGTs at 6.4 Gbps, for a total bandwidth of 99.3 Gbps) and a 3 channels one for the Demonstrator setup (12 MGTs running at 10 Gbps for a total of 116 Gbps). The latter configuration has been dictated by MGTs clocking limitation identified in the Demonstrator setup. Nevertheless, in both the scenarios the bandwidth requirements of the system have been fully satisfied.

An additional benefit associated to this approach is represented by the possibility of varying the number of bits to be transferred per BC, easing possible modifications of the firmware.

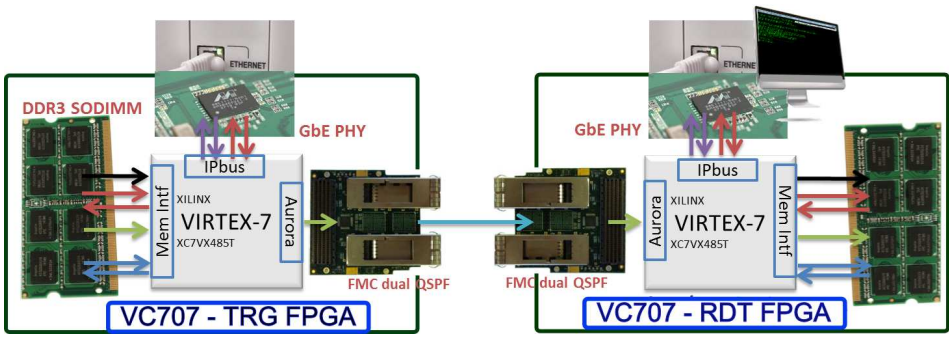
10.4. Firmware validation

Anticipating the final CTPCORE+ firmware, the deployed firmware blocks were integrated into a system that was tested on the Demonstrator. As depicted in Figure 3, of the two VC707 boards, the first one was configured as the TRG FPGA, while the second one implements the RDT FPGA. The Control and Monitoring Interface (IPBus) was used to configure the system and preload/readback the content of the DDR3 SDRAM modules.

More in details, during the initialization phase, the TRG DDR3 memory is loaded with trigger inputs and the chip to chip links are configured and enabled. Successively, the pre-loaded trigger inputs are read and transmitted via the chip to chip links to the RDT board where are stored in the on-board DDR3 memory. Finally, via the control and monitoring interface, the data can be extracted from the RDT DDR3 memory and sent to the monitoring computer, where their correctness can be verified.

The validation system has proven the correctness of the design at block and system level, providing an excellent starting point for the design of the final CTPCORE+ module firmware. This latter will be presented in the next Chapter, together with the CTPCORE+ validation process.

FIGURE 3. Firmware validation. The demonstrator setup has been used to validate a subset of the CTPCORE+ board functionalities.



CHAPTER 11

CTPCORE+ module: validation and testing

PREVIOUS Chapters have presented the CTPCORE+ module, its main functionalities and finalities, describing as well as the activities conducted to design and validate its components. This Chapter addresses the final validation activities conducted on the CTPCORE+ module, presenting as well some preliminary characterizations of the the final system.

As it will be better detailed through all the Chapter, the deployed CTPCORE+ board satisfies all the design specifications and will be installed in the ATLAS experiment environment before the end of the technical shutdown.

11.1. CTPCORE+ board validation

As it has already been discussed, various proof-of-concept tests had been conducted on a system similar to the CTPCORE+ (the so-called Demonstrator setup) in order to validate various aspects of the CTPCORE+ module design. The same tests has been executed on the first prototype of the CTPCORE+ board, in order to confirm the validity of the design choices made.

Nevertheless, some features of the final system, due to dissimilarity between the Demonstrator and the CTPCORE+ module required new tests, specifically designed for the CTPCORE+ module.

In this initial part of the Chapter, these tests and their outcomes will be presented.

11.1.1. Clock generator.

The CTPCORE+ board relies on an external jitter cleaner (the TI CDCE62005 [**www14n**]) for cleansing the Bunch Crossing clock and for generating the MGTs reference clocks (160 MHz).

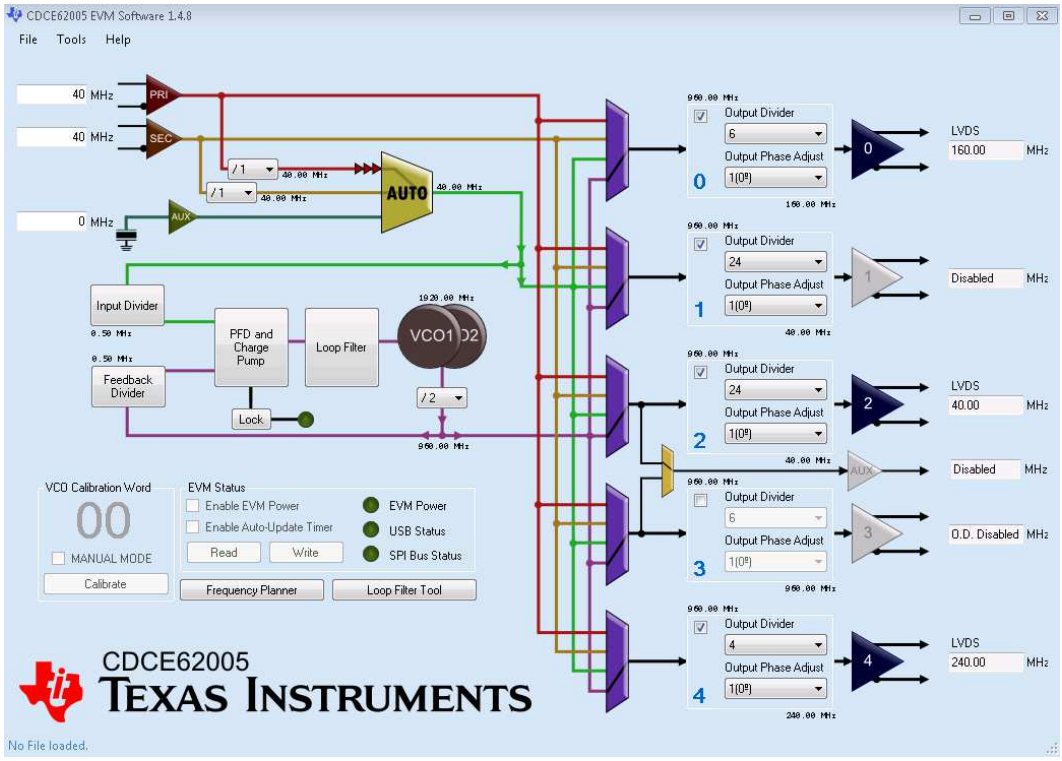
The module can be configured to output multiples of the BC clock (e.g. 40.078, 160.312 and 240.468 MHz) via an SPI interface directly connected to the VME FPGA.

A proprietary GUI, (shown in Figure 1), can be used to configure the properties of the CDCE62005 clock generator. The GUI generates an initialization file that contains the values of all the internal registers. These values can be easily read and used to configure the module via the SPI interface.

The final release of the CTPCORE+ module firmware provides VME access (presented in the proceeding) to these configuration registers easing the module setup. The correctness of the generated clocks have been verified using a custom firmware block. This latter compares the generated clocks against a known-value reference clock, providing a measure of the relative clock frequency that can be used to compute the absolute values of the clock frequencies.

The CDCE62005 module and its integration with VME interface has been fully validated. The module operates correctly and it generates the expected MGTs reference clocks.

FIGURE 1. TI CDCE62005 Jitter Cleaner configuration GUI. Clocks frequencies, output signal properties and internal PLL settings can be fully customized via the user interface.



11.1.2. Connectivity tests.

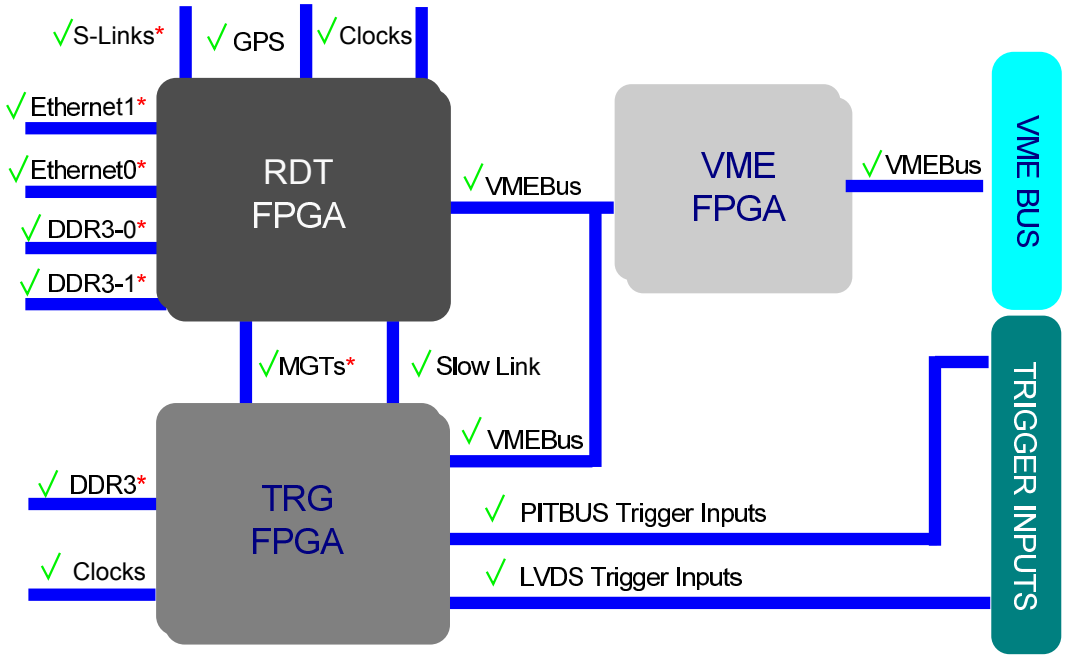
The TRG FPGA receives trigger inputs from the backplane and from the front panel, while the RDT FPGA receives trigger summary information from the TRG FPGA, a timing reference from the GPS card and data from two Ethernet interfaces. Furthermore, bidirectional links exist between TRG and RDT FPGAs as well as between the two Virtex-7 chips and the VME FPGA.

Due to this extensive number of connections, a manual connectivity test was considered excessively time consuming and extremely difficult to implement. Therefore internal probing firmware blocks were implemented, mainly relying on the Internal Logic Analyser (ILA) and Virtual IO (VIO) blocks provided by Xilinx. Nevertheless, for some of the I/O interfaces, special firmware blocks had to be used in order to verify the I/O connectivity.

Figure 2 lists the verified connections, indicating with a * the custom made firmware blocks. It should be clarified that the target of this step was only the validation of all connections and links, whether the full functionality checks has been conducted in a sequent phase.

The results of the the analyses are extremely satisfactory: on the CTPCORE+ board under analysis all the connections are valid and correct.

FIGURE 2. CTPCORE+ connectivity tests. The Ethernet, the DDR3 interfaces and the MGT links have been verified with the validation firmware blocks previously presented. For testing the other links, instead, edge detectors, counters or internal logic analyser blocks have been used.



11.1.3. Chip to Chip links testing.

The TRG to RDT link has been preliminary validated on the Demonstrator using the Xilinx IBERT tool. Due to the high data-rates at which these links must be operated (6.4 Gbps) and the differences existing between the Demonstrator setup and the CTPCORE+ board, the previously presented IBERT based tests had to be repeated also on the CTPCORE+ module.

The outputs of the IBERT tool are depicted in Figure 3. The eye-diagram and bathtube curves have been generated by the IBERT tool operating the links at 6.4 Gbps and targeting a BER of 10^{-12}.

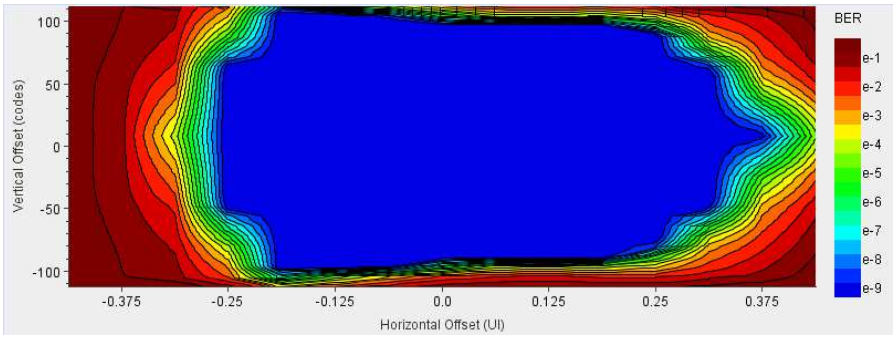
The tests conducted on the CTPCORE+ module fully validate the design choices, in fact, the links appear to be extremely well behaving with wide open eye-diagrams.

It is worth noticing that the good quality of the links can indirectly help in reducing the overall power consumption of the MGT modules without impairing the link quality. In particular, the amount of current drained by the MGTs can be significantly reduced using a limited voltage swing output on the transmitter side and adopting a low-power equalization mode on the receiver side.

FIGURE 3. CTPCORE+ board chip-to-chip link quality assessment.



(a) Link Bathtube Analysis



(b) Link Eye Diagram Analysis

11.1.4. Power consumption and thermal dissipation investigations.

The CTPCORE+ module must operate continuously and without disruptions for a long period of time. The reliability of the system depends primarily on the correct design of both the Power Distribution Network and the cooling system.

TABLE 1. Set of functionality supported by the VME, TRG and RDT FPGA.

Module	Functionality
VME FPGA	External jitter cleaner configuration Routing of VME requests to/from TRG and RDT FPGAs Interface to the VME bus (backplane)
TRG FPGA	Synchronization and alignment of backplane signals DDR3 based snapshot and playback functionality Transmission of the trigger information to the RDT FPGA
RDT FPGA	Reception of trigger information from the TRG FPGA GPS timing reference generation DDR3 based snapshot functionality

While the correctness of the former has been preliminary investigated relying on the Demonstrator, the latter could only be validated on the CTPCORE+ module. To properly characterize these aspects, the CTPCORE+ board had to be installed in the experiment crate, all the FPGAs must be loaded with a complete and functional firmware and all the peripheral had to be configured with the final firmware. Concerning the latter aspect, Table 1 summarizes the set of functionality supported by the different modules.

Power consumption and thermal analysis had been conducted leveraging two different tools: the ProGUI [**www11d**] software had been used to access the DC/DC controller and to retrieve the current consumption of each different rails, while the Xilinx Vivado environment has been used to read temperature sensors embedded in the Virtex-7 chips. Table 2 contains the measured temperatures, while Table 3 presents the current consumption of the main voltage rails.

TABLE 2. Temperature measurements. The currently mounted heat dissipation units may be inadequate. Heat sinks with higher thermal conductivity can improve the scenario, preventing an accelerated ageing of the chip.

Module	Min. Temperature[C°]	Max. Temperature[C°]
TRG FPGA	38	45
RDT FPGA	44	59+

TABLE 3. Voltage rails consumption measurements. It must be noted that some DC/DC outputs feed multiple chips, thus the individual chip consumption can not be easily extracted. Concerning the idle current values, they have been measured with the device brought to a partial reset status (VME fully operative, TRG and RDT hold on reset).

Voltage [V]	Idle current [A]	Peak current [A]
2.5	3	3
1.8	2.4	2.8
1.5	1.875	2
1	1	1.5
1	1.5	2.8
1.5	2.5	3.5
1.2	4.7	6

Considering the power consumption of the module, all the DC/DC converters appear to be operating within their specifications. Future investigations could potentially address DDR3 memories and MGTs controllers in search of ways to reduce their consumption (i.e. more power-aware transceivers configuration and slower memory operating frequency/better throughput).

Concerning the heat dissipation subsystem of the CTPCORE+ module, instead, minor modifications are already foreseen. In particular, the RDT FPGA, that tends to overheat more than the TRG counterpart, could benefit a more effective heat-sink, eventually a custom made one.

11.2. CTPCORE+ Firmware

As presented before, the Demonstrator has been used as development platform to anticipate the design of the main CTPCORE+ module firmware modules. The previously described IP-based control interface, Chip-to-chip protocol and DDR3 controller have been designed, tested and validated on the Demonstrator. Starting with these tested blocks, to obtain the final firmware, few missing components had to be designed and the legacy CTPCORE firmware had to be ported to the new architecture. In particular, various portions of the L1A signal generation logic (TRG FPGA) and of the monitoring functionality (RDT FPGA) had to be redesigning considering the specific characteristics of the Virtex-7 chips.

The status and results of these activities are presented in the proceeding.

11.2.1. VME Interface.

While the Demonstrator relies on an Ethernet based control and monitoring protocol, the CTPCORE+ board must communicate with the other interconnected modules via the VME interface.

In the ATLAS experiment, a single board computer (SBC) can monitor and configure the CTPCORE+ module through the VME bus. The SBC maps internal resources of the TRG, RDT and VME FPGAs into the Operative System address space.

More in detail, the VME FPGA — that is directly connected to the VME bus — directs the requests received from the SBC to the TRG/RDT FPGAs and vice-versa. The VME FPGA receives the VME bus signals directly from the backplane and using an internal address decoder routes the requests to the targeted module. If the request is directed to the VME FPGA address space the register access is resolved internally and the operation result is sent back to the SBC. Whether, instead, the address belongs to the TRG or RDT FPGA address spaces, the request has to be routed to the appropriate destination. Figure 4 provides a graphical description of the internal implementation of this interface.

It is worth noticing that in order to reduce the bus pin-count the internal connections between TRG and VME FPGAs as well as between RDT and VME FPGAs have been implemented in a partially serialized fashion, enabling two full duplex links (TRG-VME and RDT-VME). A serialization factor of 4 is used to decrease the number of required pins per FPGA from more than 70 down to 20 pins per duplex link. The serialization and de-serialization blocks have been designed to accommodate this specific need, implementing for the reception logic an additional BC synchronization block.

The implemented interface has been validated leveraging the automatic code generation feature of the so-called hardware compiler [**SBE**$^{+}$09]. The compiler relies on an XML based representation of the different modules connected to the VME bus. Specific XML tags can be used to define VME-addressable elements such as registers and memories. The hardware compiler parses the XML file and generates C++/Python functions to access the registers and memories defined by the XML description file.

The use of the hardware compiler provides significant benefits. Firstly, the code readability largely increases while the software design and system debug become less error-prone. Secondly, the compiler

FIGURE 4. VME Controller functional blocks. The VME FPGA is directly interface to the VME bus, while the TRG and RDT FPGAs are connected to the bus indirectly, in a daisy-chain fashion.

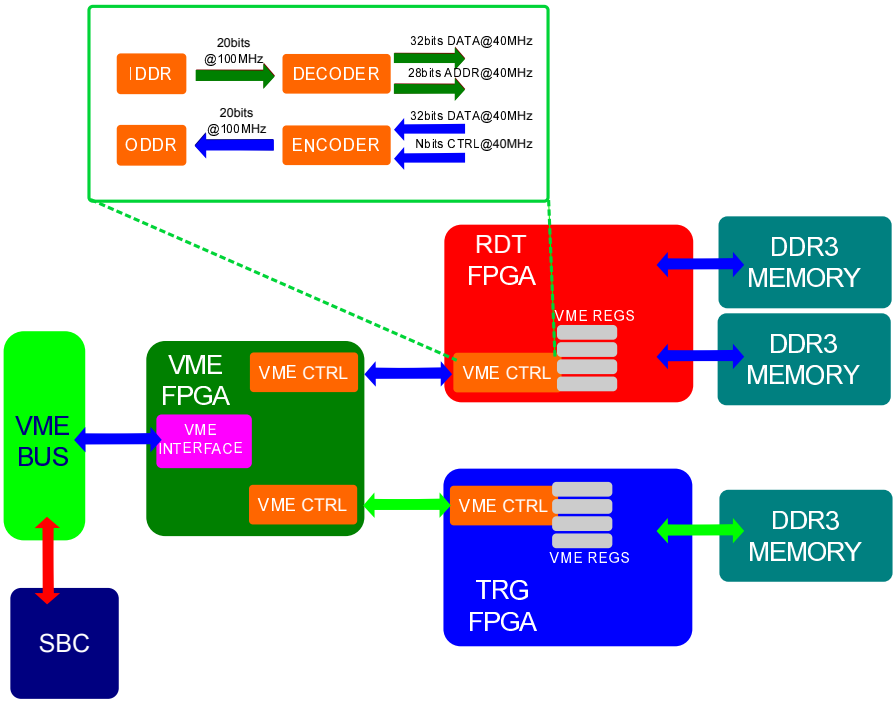
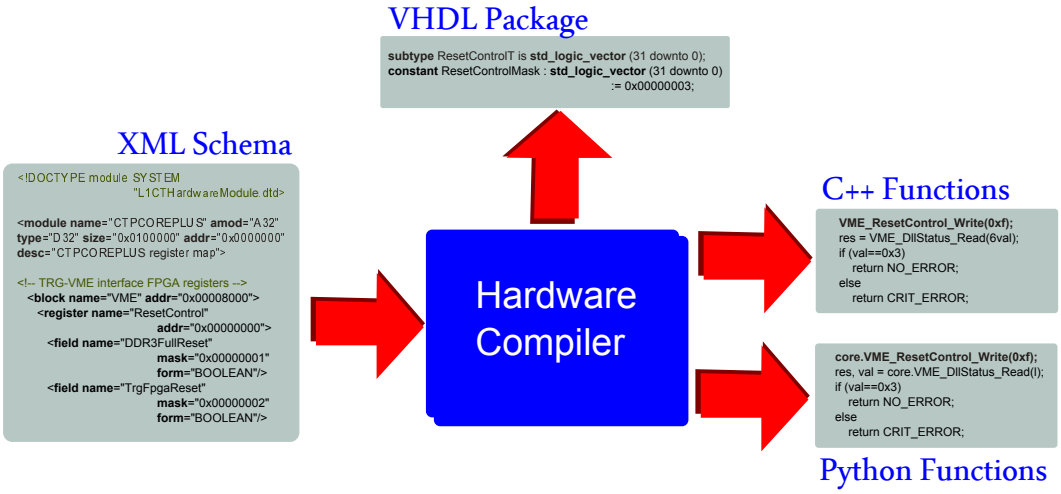


FIGURE 5. Level-1 Hardware Compiler functionality.



verifies the address map correctness enforcing additional rules, such as bit-by-bit access modifiers and operation return type.

A simplified graphical description of the Hardware Compiler functionality is presented in Figure 5.

11.2.2. Firmware finalization.

The CTPCORE+ firmware has been developed in an incremental fashion in order to validate the various sub-blocks and ease the debug process.

Concerning the VME FPGA, the interface to the VME bus and the SBC as well as the two interfaces to the TRG and RDT FPGAs represents up to now, the main firmware components programmed in this device. The remaining activities are expected to be completed in the near future, with a finalized version foreseen for Q3 2014. Current activities are addressing a VME-based remote reprogramming feature. This latter will provide a way to reprogram the TRG and RDT FPGAs in a remote fashion, thus simplifying functionality upgrade, bug fixes and module maintenance.

The TRG FPGA has readable and writeable VME registers, a validated version of the DDR3 controller and a fully operative Chip-to-chip fast links interface. Inputs from the backplane are aligned and synchronized to the BC domain before being routed to the L1A generation logic. Optionally, the system user can store the trigger inputs inside the DDR3 memory by configuring specific VME registers. The DDR3 memory can be also configured directly via the VME interface, again by setting some internal VME registers. Finally, playback feature is supported: pre-written data can be read from the memory and passed to the L1A generation logic. In all the modalities, trigger summary information are transmitted to the RDT FPGA via the Chip-to-chip fast link. This link can be reset and configured via VME interface. Monitoring registers are foreseen (i.e. number of packet dropped, link reliability). Expect for the L1A generation logic, all the other blocks have been fully integrated and validated. The missing part of firmware has been finalized and tested and it is under integration. The block strict timing constraints appear to be fully respected.

Similarly, the current RDT FPGA firmware supports the virtual dual port DDR3 memory and the chip-to-chip fast interface. Trigger summary information received from the TRG FPGA are stored in the two DDR3 memories (configured as a single twofold size memory) and read back via the VME interface. The interface to the GPS module has been ported from the legacy CTPCORE+ firmware and partially integrated in the system. The monitoring functionality, while fully ported from the CTPCORE architecture to the CPTCORE+ one are still to be integrated and fully verified.

A simulation model contains the firmware of the three FPGA modules, functional models for the DDR3 memories and a VME interface emulator. This model provides a valuable tool to validate design modifications and to identify bugs and design sub-optimality.

Bibliography

[ACO12] B. Alecsa, M.N. Cirstea, and A. Onea. Simulink modeling and design of an efficient hardware-constrained fpga-based pmsm speed controller. *Industrial Informatics, IEEE Transactions on*, 8(3):554–562, Aug 2012.

[Agr10] G. P. Agrawal. *Fiber-Optic Communication Systems*. Wiley, 2010.

[BB99] S. Benedetto and E. Biglieri. *Principles of Digital Transmission: With Wireless Applications*. Springer, 1999.

[BCA12] H. Boughrara, M. Chtourou, and C. B Amar. Mlp neural network based face recognition system using constructive training algorithm. *Multimedia Computing and Systems (ICMCS)*, 2012.

[BETVG08] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding (CVIU)*, 2008.

[BGS+11] R. Bhargavi, K. Ganesh, M. Sekar, P. Singh, and V. Vaidehi. An integrated system of complex event processing and kalman filter for multiple people tracking in wsn. *Recent Trends in Information Technology (ICRTIT)*, 2011.

[BK08] G. Bradski and A. Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly, 2008.

[Bog10] E. Bogatin. *Signal and Power Integrity - Simplified*. Prentice Hall Modern Semiconductor Design Series, 2010.

[BRC60] R. C. Bose and D. K. Ray-Chaudhuri. On a class of error correcting binary group codes. In *Information and Control*, pages 68–79, 1960.

[Can86] J. Canny. A computational approach to edge detection. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 1986.

[CFFP09] G. Colavolpe, T. Foggi, E. Forestieri, and G. Prati. Robust multilevel coherent optical systems with linear processing at the receiver. *Lightwave Technology, Journal of*, 27(13):2357–2369, 2009.

[Col94] CMS Collaboration. *The Compact Muon Solenoid: Technical Proposal, CERN-LHCC-94-38*. CERN, 1994.

[Col95] ALICE Collaboration. *Technical Proposal for a Large Ion Collider Experiment at the CERN LHC, CERN-LHCC-95-71*. CERN, 1995.

[Col96a] ATLAS Collaboration. *Calorimeter Performance Technical Design Report*. CERN/LHCC/96-40, 1996.

[Col96b] ATLAS Collaboration. *Liquid Argon Calorimeter Technical Design Report*. CERN/LHCC/96-41, 1996.

[Col96c] ATLAS Collaboration. *Tile Calorimeter Technical Design Report*. CERN/LHCC/96-42, 1996.

[Col97] ATLAS Collaboration. *Muon Spectrometer Technical Design Report*. CERN/LHCC/97-22, 1997.

[Col98] LHCb Collaboration. *LHCb Technical Proposal, CERN-LHCC-98-04*. CERN, 1998.

[Col99] ATLAS Collaboration. *Detector and Physics Performance Technical Design Report, CERN-LHCC-99-14*. CERN, 1999.

[col11] The ATLAS collaboration. Letter of intent for the phase-i upgrade of the atlas experiment. *CERN-LHCC-2011-012*, 2011.

[CPC+12] R. Corsini, R. Pelliccia, G. Cossu, A. M. Khalid, M. Ghibaudi, M. Petracca, P. Pagano, and E. Ciaramella. Free space optical communication in the visible bandwidth for v2v safety critical protocols. *Wireless Communications and Mobile Computing Conference (IWCMC)*, 2012.

[CR99] G. Colavolpe and R. Raheli. Noncoherent sequence detection. *Communications, IEEE Transactions on*, 47(9):1376–1385, 1999.

[CZJ11] B. Cao, Q.Z. Zhang, and L. Jin. Polarization division multiple access with polarization modulation for los wireless communications. *EURASIP Journal on Wireless Communications and Networking*, 35(9):877–887, 2011.

[Der88] R. Deriche. Fast algorithm for low-level vision. *International Conference on Pattern Recognition*, 1988.

[DPFG+09] A. De Paola, A. Farruggia, S. Gaglio, G. Lo Re, and M. Ortolani. Exploiting the human factor in a wsn-based system for ambient intelligence. *Complex, Intelligent and Software Intensive System(CISIS)*, 2009.

[ea08] S. Ask et al. The atlas central level-1 trigger logic and ttc system. *JINST 3 P08002*, 2008.

[EU10] EU. On the framework for the deployment of intelligent transport systems in the field of road transport and for interfaces with other modes of transport. `http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2010:207:0001:0013:EN:PDF`, 2010.

[GM05] A. Galtarossa and C. R. Menyuk. *Polarization Mode Dispersion*. Springer, 2005.

[GMW81] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization*. Prentice HallEmerald Group Publishing Limited, 1981.

[GPK11] L. Gogolak, S. Pletl, and D. Kukolj. Indoor fingerprint localization in wsn environment based on neural network. *Intelligent Systems and Informatics (SISY)*, 2011.

[GSR+09] M. P. Gahemmaghami, H. Sameti, F. Razzazzi, B. BabaAli, and S. Dabbaghchian. Robust speech recognition using mlp neural network in log-spectral domain. *Signal Processing and Information Technology (ISSPIT)*, 2009.

[Hay08] S. Haykin. *Neural Networks and Learning Machines*. Prentice Hall, 2008.

[HC99] Y. L. Huang and R. F. Chang. Mlp interpolation for digital image processing using wavelet transform. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1999.

[HCS07] L. L. Hanzo, P. Cherriman, and J. Streit. *Video Compression and Communications: From Basics to H.261, H.263, H.264, MPEG4 for DVB and HSDPA-Style Adaptive Turbo-Transceivers*. Prentice Hall, 2007.

[HFB94] K.B. Hardin, J.T. Fessler, and D.R. Bush. Spread spectrum clock generation for the reduction of radiated emissions. In *Electromagnetic Compatibility, 1994. Symposium Record. Compatibility in the Loop., IEEE International Symposium on*, pages 227–231, Aug 1994.

[HP98] J. Hennessey and D. A. Patterson. *Computer Organization and Design (2nd ed.)*. Morgan Kaufmann Publishers, 1998.

[Kaz89] Leonid G. Kazovsky. Phase- and polarization-diversity coherent optical techniques. *Lightwave Technology, Journal of*, 7(2):279–292, 1989.

[KMH10] Z.G. Kovacs, G.E. Marosy, and G. HorvaÌĄth. Case study of a simple, low power wsn implementation for forest monitoring. *Biennial Baltic Electronics Conference (BEC)*, 2010.

[Lam12] V. Lambersky. Model based design and automated code generation from simulink targeted for tms570 mcu. In *Education and Research Conference (EDERC), 2012 5th European DSP*, pages 225–228, Sept 2012.

[Low99] D. Lowe. Object recognition from local scale-invariant features. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 1999.

[LXSL09] Liqun Li, Guoliang Xin, Limin Sun, and Yan Liu. Qvs: Quality-aware voice streaming for wireless sensor networks. In *Distributed Computing Systems, 2009. ICDCS '09. 29th IEEE International Conference on*, pages 450–457, 2009.

[MD97] U. Mengali and A. N. D'Andrea. *Synchronization Techniques for Digital Receivers (Applications of Communications Theory)*. Springer, 1997.

[MMN+11] M. Magrini, D. Moroni, C. Nastasi, P. Pagano, M. Petracca, G. Pieri, C. Salvadori, and O. Salvetti. *Pattern Recognition and Image Analysis*, chapter Visual sensor networks for infomobility, pages 20–29. Elsevier, 2011.

[Moo05] T. D. Moon. *Error Correction Coding: Mathematical Methods and Algorithms*. Wiley, 2005.

[MP43] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 1943.

[MRP+11] R. Mambrini, A. Rossi, P. Pagano, P. Ancilotti, O. Salvetti, A. Bertolino, P. Gai, and L. Costalli. Ipermob: Towards an information system to handle urban mobility data. *Models and Technologies for ITS*, 2011.

[MRRS06] R. Mangharam, A. Rowe, R. Rajkumar, and R. Suzuki. Voice over sensor networks. In *Real-Time Systems Symposium, 2006. RTSS '06. 27th IEEE International*, pages 291–302, 2006.

[MRX08] S. Misra, M. Reisslein, and G. Xue. A survey of multimedia streaming in wireless sensor networks. *IEEE Communications Surveys and Tutorials*, 2008.

[Nak09] M. Nakazawa. Optical quadrature amplitude modulation (qam) with coherent detection up to 128 states. In *Optical Fiber Communication - incudes post deadline papers, 2009. OFC 2009. Conference on*, pages 1–3, March 2009.

[Nas12] C. Nastasi. *Distributed Video Signal Processing for Wireless Multimedia Sensor Networks*. PhD thesis, SSSA, October 2012.

[PBR+11] S. Paniga, L. Borsani, A. Redondi, M. Tagliasacchi, and M. Cesana. Experimental evaluation of a video streaming system for wireless multimedia sensor networks. In *Ad Hoc Networking Workshop (Med-Hoc-Net), 2011 The 10th IFIP Annual Mediterranean*, pages 165–170, 2011.

[PGPP12] M. Petracca, M. Ghibaudi, R. Pelliccia, and P. Pagano. Perceptual voice communications in ieee802.15.4 networks for the emergency management support. In *Future Internet Communications (BCFIC), 2012 2nd Baltic Congress on*, pages 121–126, 2012.

[PGS+11a] M. Petracca, M. Ghibaudi, C. Salvadori, P. Pagano, and D. Munaretto. Performance evaluation of fec techniques based on bch codes in video streaming over wireless sensor networks. In *Proceedings of the 2011 IEEE Symposium on Computers and Communications*, ISCC '11, pages 43–48, Washington, DC, USA, 2011. IEEE Computer Society.

[PGS+11b] M. Petracca, M. Ghibaudi, C. Salvadori, P. Pagano, and D. Munaretto. Performance evaluation of fec techniques based on bch codes in video streaming over wireless sensor networks. In *Computers and Communications (ISCC), 2011 IEEE Symposium on*, pages 43–48, 2011.

[PLR+09] M. Petracca, G. Litovsky, A. Rinotti, M. Tacca, J.C. De Martin, and A. Fumagalli. Perceptual based voice multi-hop transmission over wireless sensor networks. In *Computers and Communications, 2009. ISCC 2009. IEEE Symposium on*, pages 19–24, 2009.

[PM08] J. Proakis and Salehi M. *Digital Communications*. McGraw-Hill Higher Education, 2008.

[PP87] G. Picchi and G. Prati. Blind equalization and carrier recovery using a "stop-and-go" decision-directed algorithm. *Communications, IEEE Transactions on*, 35(9):877–887, 1987.

[PPAS13] P. Pagano, M. Petracca, D. Alessandrelli, and C. Salvadori. Is ict mature for an eu-wide intelligent transport system? *Intelligent Transport Systems (IET)*, 2013.

[PPP+12] M. Petracca, P. Pagano, R. Pelliccia, M. Ghibaudi, C. Salvadori, and C. Nastasi. *Roadside Networks for Vehicular Communications: Architectures, Applications and Test Fields*, chapter On Board Unit hardware and software design for Vehicular Ad-hoc NETworks, pages 38–56. IGIGlobal, 2012.

[RB93] M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: The rprop algorithm. *IEEE International Conference on Neural Networks*, 1993.

[RHW86] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 1986.

[Ros58] F. Rosenblat. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 1958.

[SBE+09] R Spiwoks, D Berge, N Ellis, P Farthouat, S Haas, J. Lundberg, S. Maettig, A. Messina, T. Pauly, and D. Sherman. Framework for testing and operation of the atlas level-1 muctpi and ctp. In *Topical Workshop on Electronics for Particle Physics, Paris, France*, September 2009.

[SBP+12] C. Salvatore, S. Bocchino, M. Petracca, R. Pelliccia, M. Ghibaudi, and P. Pagano. Wsn and rfid integrated solution for advanced safety systems in industrial plants. *Telecommunications and Computer Networks (SoftCOM)*, 2012.

[SG99] R. Schober and W.H. Gerstacker. Metric for noncoherent sequence estimation. *Electronics Letters*, 35(25):2178–2179, 1999.

[SGG+13] L. Sanchez, V. Gutierrez, J. Galache, P. Sotres, J. Santana, J. Casanueva, and L. Munoz. Smartsantander: Experimentation and service provision in the smart city. *Wireless Personal Multimedia Communications (WPMC)*, 2013.

[Sim12] E. et al. Simioni. Topological and central trigger processor for 2014 lhc luminosities. In *18th IEEE Real-Time Conference, ATL-DAQ-PROC-2012-041*, June 2012.

[SLXX10] Z. Sun, W. Li, H. Xiao, and L. Xu. The research on solar power system of wireless sensor network node for forest monitoring. *Web Information Systems and Mining (WISM)*, 2010.

[SPP+12] C. Salvadori, M. Petracca, R. Pelliccia, M. Ghibaudi, and P. Pagano. Video streaming in wireless sensor networks with low-complexity change detection enforcement. In *Future Internet Communications (BCFIC), 2012 2nd Baltic Congress on*, pages 8–13, 2012.

[st600] 64b/66b low-overhead coding proposal for serial links, 2000.

[Tay04] M.G. Taylor. Coherent detection method using dsp for demodulation of signal and subsequent equalization of propagation impairments. *Photonics Technology Letters, IEEE*, 16(2):674–676, 2004.

[TC11] M. Taj and A. Cavallaro. Distributed and decentralized multicamera tracking. *Signal Processing Magazine, IEEE*, 28(3):46 –58, May 2011.

[WTK11] N. Watthanawisuth, A. Tuantranont, and T. Kerdcharoen. Design for the next generation of wireless sensor networks in battlefield based on zigbee. *Defense Science Research Conference and Expo (DSR)*, 2011.

[www96] ITU-T P.800. `http://www.itu.int/rec/T-REC-P.800-199608-I/en`, 1996.

[www98] VME64x standard specifications. `https://ph-dep-ese.web.cern.ch/ph-dep-ese/crates/standards/Av1dot1.pdf`, 1998.

[www01] Principles of QPSK and QAM modulations. `http://www.cascaderange.org/presentations/QPSK_and_16-QAM_Digital_Modulation.pdf`, 2001.

[www09] Codec and Media Specification. `http://www.cablelabs.com/wp-content/uploads/specdocs/PKT-SP-CODEC-MEDIA-I07-090702.pdf`, 2009.

[www10] The Road to 100G Networking. http://www.nggovernmentsummit.com/media/whitepapers/Ciena_GOVCIO.pdf, 2010.

[www11a] Ml605 Evaluation board. http://www.xilinx.com/products/boards-and-kits/EK-V6-ML605-G.htm, 2011.

[www11b] Virtex-6 FPGA DSP48E1 Slice. http://www.xilinx.com/support/documentation/user_guides/ug369.pdf, 2011.

[www11c] Xilinx FFT IPcore. http://www.xilinx.com/support/documentation/ip_documentation/xfft_ds260.pdf, 2011.

[www11d] General Electric ProGui monitoring tool. http://ptelectronics.ru/wp-content/uploads/otladochniy_nabor_digital_power_insight_ge_critical_power.pdf, 2011.

[www11e] The IPbus Protocol & The IPbus Suite. https://cactus.hepforge.org/attachments/rob_frazier_ipbus_tipp_2011.pdf., 2011.

[www12a] Modulation Schemes, the Mach-Zender Modulator. http://www.shf.de/communication/support/tutorial_notes/getfile/72/288/tutnote5.pdf, 2012.

[www12b] States of Optical Polarization, white paper. http://m.semrock.com/Data/Sites/1/semrockpdfs/whitepaper_understandingpolarization.pdf, 2012.

[www12c] Coherent detection for fiber optics networks. http://ewh.ieee.org/r2/baltimore/leos/Sampled_coherent_detection_APL.pdf, 2012.

[www14a] Altera Stratix 10 FPGA. http://www.altera.com/devices/fpga/stratix-fpgas/stratix10/stx10-index.jsp, 2014.

[www14b] Dispersion and Ultrashort Pulses. http://ticc.mines.edu/csm/wiki/images/f/f0/UFO05-Dispersion.pdf, 2014.

[www14c] Matlab HDL Coder. http://www.mathworks.co.uk/products/hdl-coder/, 2014.

[www14d] Metastability explained. Altera White Paper. http://www.altera.com/literature/wp/wp-01082-quartus-ii-metastability.pdf, 2014.

[www14e] Virtex-6 FPGA Clocking Resources. http://www.xilinx.com/support/documentation/user_guides/ug362.pdf, 2014.

[www14f] Xilinx Virtex Ultrascale. http://www.xilinx.com/products/technology/ultrascale.html, 2014.

[www14g] Cadence Allegro, PCB editor. \urlhttp://www.cadence.com/products/pcb/pcb_design/pages/default.aspx, 2014.

[www14h] Cisco 100G solutions. http://www3.alcatel-lucent.com/features/100g_era/, 2014.

[www14i] Cisco whitepaper, the Zettabyte Era. http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/VNI_Hyperconnectivity_WP.html, 2014.

[www14j] Codec comparison. http://www.broadcom.com/support/broadvoice/codec_comparison.php, 2014.

[www14k] FM-28, dual QSFP Faster Technologies FMC card. http://www.fastertechnology.com/products/fmc/fm-s28.html, 2014.

[www14l] Mathworks Matlab Neural Network Toolbox. http://www.mathworks.co.uk/products/neural-network/, 2014.

[www14m] NTT. http://www.ntt-at.com/product/speech2002/, 2014.

[www14n] Texas Instrument CDCE62005 Low-Jitter Clock Generator. www.ti.com/lit/ds/symlink/cdce62005.pdf, 2014.

[www14o] The LightWeight IP stack. http://savannah.nongnu.org/projects/lwip/, 2014.

[www14p] VC707, Xilinx Virtex-7 Evaluation Board. http://www.xilinx.com/products/boards-and-kits/EK-V7-VC707-G.htm, 2014.

[Zha11] X. Zhang. Adaptive control and reconfiguration of mobile wireless sensor networks for dynamic multi-target tracking. *IEEE Transactions on Automatic Control*, 2011.