A Real-time Control Computer for the E-ELT

Document: GF-PDR-06

FPGA Solution for hard real-time

Version 1.0

18th January 2016

Change Record

Version	Date	Author(s)	Remarks
0.1	11 Jan 2016	Christian Patauner	Initial skeleton version
1.0	18 Jan 2016	Christian Patauner	Add chapter 2.4 - On board computational power considerations
1.0	18 Jan 2016	Christian Patauner	Add table of contents

Table of Contents

Applicable Documents (AD)

These are the Green Flash PDR documents

No.	Title	Reference	Issue	Date
AD01	Introduction to Green Flash	GF-PDR-01		
AD02	Management plan and WP definition	GF-PDR-02		
AD03	Requirements Specification	GF-PDR-03		
AD04	System Architecture	GF-PDR-04		
AD05	Distributed GPUs for real-time HPC	GF-PDR-05		
AD06	FPGA Solution for hard real-time	GF-PDR-06		
AD07	Smart Interconnects	GF-PDR-07		
AD08	Interface Control Document	GF-PDR-08		
AD09	Supervisor design	GF-PDR-09		

Reference Documents (RD)

These are documents external to the Green Flash project

No.	Title	Reference	Issue	Date

Acronyms and abbreviations

Table 1: Acronyms and Abbreviations

AO	Adaptive Optics
BW	BandWidth
CPLD	Complex Programmable Logic Device
DM	Deformable Mirror
DSP	Digital Signal Processor
FMC	FPGA Mezzanine Card
(s)FPDP	(serial) Front Panel Data Port
FPGA	Field Programmable Gate Array
HMC	Hybrid Memory Cube
HPC	High Performance Computing
MAC	Multiply-ACcumulate
MVM	Matrix-Vector-Multiply
OS	Operating System
PCIe	Peripheral Component Interconnect Express
(Q)SFP	(Quad) Small Form-factor Pluggable
RTAI	Real-Time Application Interface
SoC	System-on-Chip
WFS	Wavefront Sensor

1 Scope

This document presents the hard real-time architecture based on FPGAs for GreenFlash.

Microgate will develop a FPGA board that implements the hard real-time data pipeline. In AD04, the components of the hard real-time data pipeline are described. We will discuss different realization schemes for this board and give a tradeoff between these solutions in chapter 2. In chapter 3, the interfaces of this board are discussed. At the end a consideration about the specifications this board has to fulfill for the GreenFlash projects is given.

2 Hardware Architecture

In this chapter, three implementation schemes for the hard real-time FPGA board are

described and at the end a comparison of these schemes is given for a tradeoff.

All three schemes are based on ARRIA 10 FPGA devices form Altera that are well suited and already used to some extend by Microgate to perform MVM calculations as described in 2.4. The GreenFlash members Microgate and PLDA have a long experience with Altera FPGAs.

The ARRIA 10 Family is the most recent release of Altera and therefore optimal to guarantee long-term availability and support. ARRIA 10 FPGAs contain hard-wired DSP blocks that can perform a full Multiply-Accumulate (MAC) operation in single precision floating-point. By combining two of these DSP blocks, a dual precision floating-point MAC can be performed. One MAC operation can be executed in one clock cycle running at up to 200MHz. The larges devices of the ARRIA 10 family have more than 1500 such DSP blocks and therefore they are well suited to perform the real-time calculations in highly parallel execution.

2.1 Single FPGA system

The first scheme for the hard real-time FPGA board is based on one ARRIA 10 FPGA of the type GX. The ARRIA 10 GX contains the most transceivers. The block diagram for this scheme is shown in figure 1.

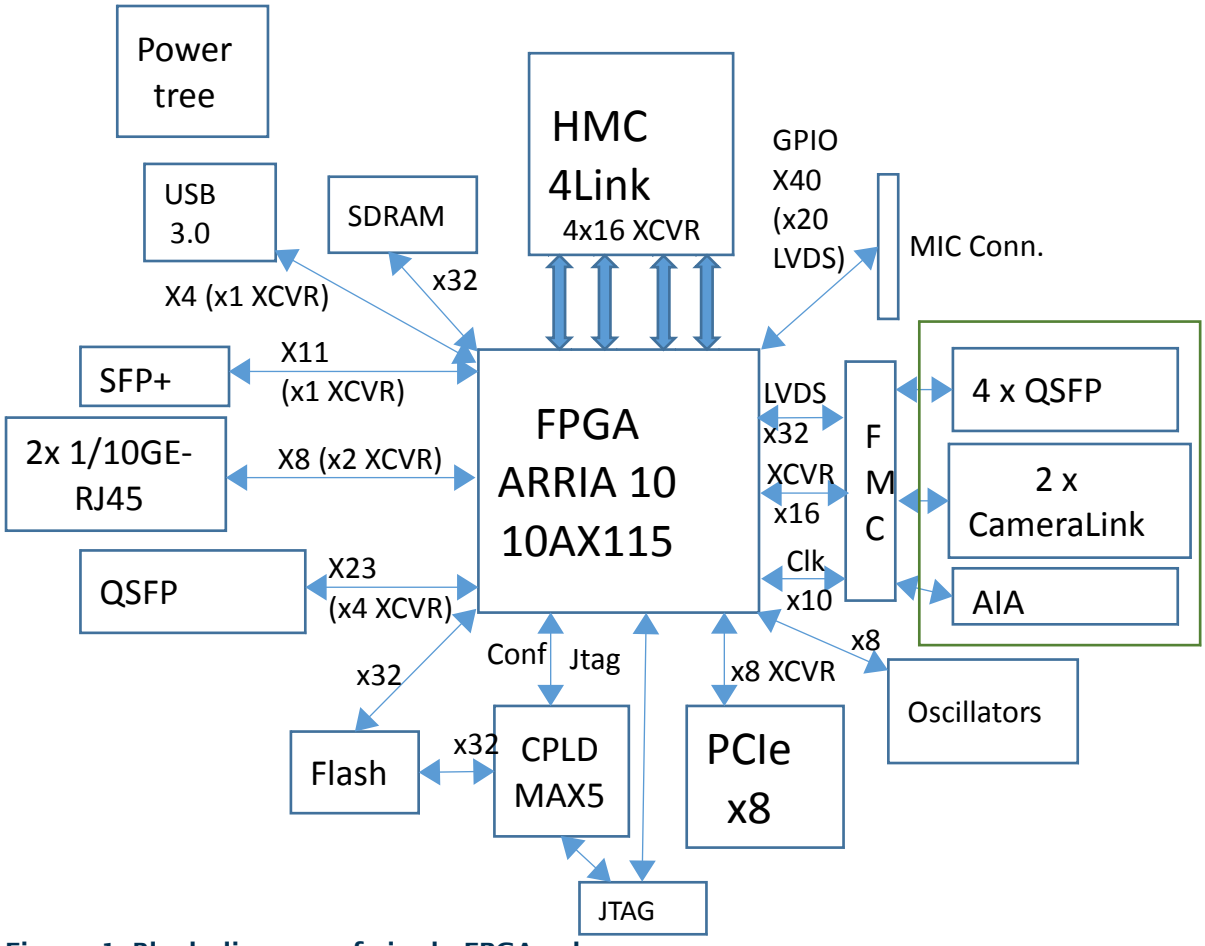

Figure 1: Block-diagram of single FPGA scheme

This large amount of transceivers (max. 96 for the 10AX115) allow an optimal interfacing with an external Hybrid Memory Cube (HMC). This HMCs are a new type of fast DRAM memories stacked vertically using true silicon via and are already available on the market e.g. from Micron. The HMC allows storing the matrices of the real-time reconstructor and to fast transfer these matrices to the FPGA (Microns HMC BW 120GB/s for each direction) in order to make the data available to the real-time calculation block inside the FPGA. The HMC chip currently available from Micron has a capacity of up to 4GB, with an interface containing up to 4 links and each link containing 16 transmit and receive lanes that allow data rates up to 15Gb/s each. In this scheme, the full bandwidth of the HMC can be explored by the selected ARRIA 10AX115.

In addition, several interfaces are connected to the FPGA in order to have a flexible board that can connect, to wavefront sensors on one side and deformable mirrors on the other side, in many different ways. The interfaces are described in more detail in chapter 3.

A Flash and CPLD MAX5 are used to store and configure the FPGA at power-on and the CPLD can as well take care of the housekeeping functionality of the board e.g. monitoring temperatures, power consumptions controlling fans etc. A SDRAM chip is added as well for using it with the soft-core NIOS II processor that can be instantiated in logic inside the FPGA.

For the PCIe interface this board can be inserted as an Endpoint accelerator card into a server or PC or other PCIe trees that contain a root complex. For a stand-alone solution, this board is limited because it contains no powerful on-chip processor as it has the next proposed scheme (only a small soft-core NIOS II processor can be instantiated inside the FPGA).

2.2 FPGA with ARM coprocessor system

This scheme described next is based on an ARRIA 10 AS FPGA that contains an ARM dual-core coprocessor in the same die. This system-on-chip (SoC) FPGA has less logic memory and transceiver resources but incorporates a dual-core ARM Cortex 9 processor. The block-diagram of this scheme is shown in figure 2.

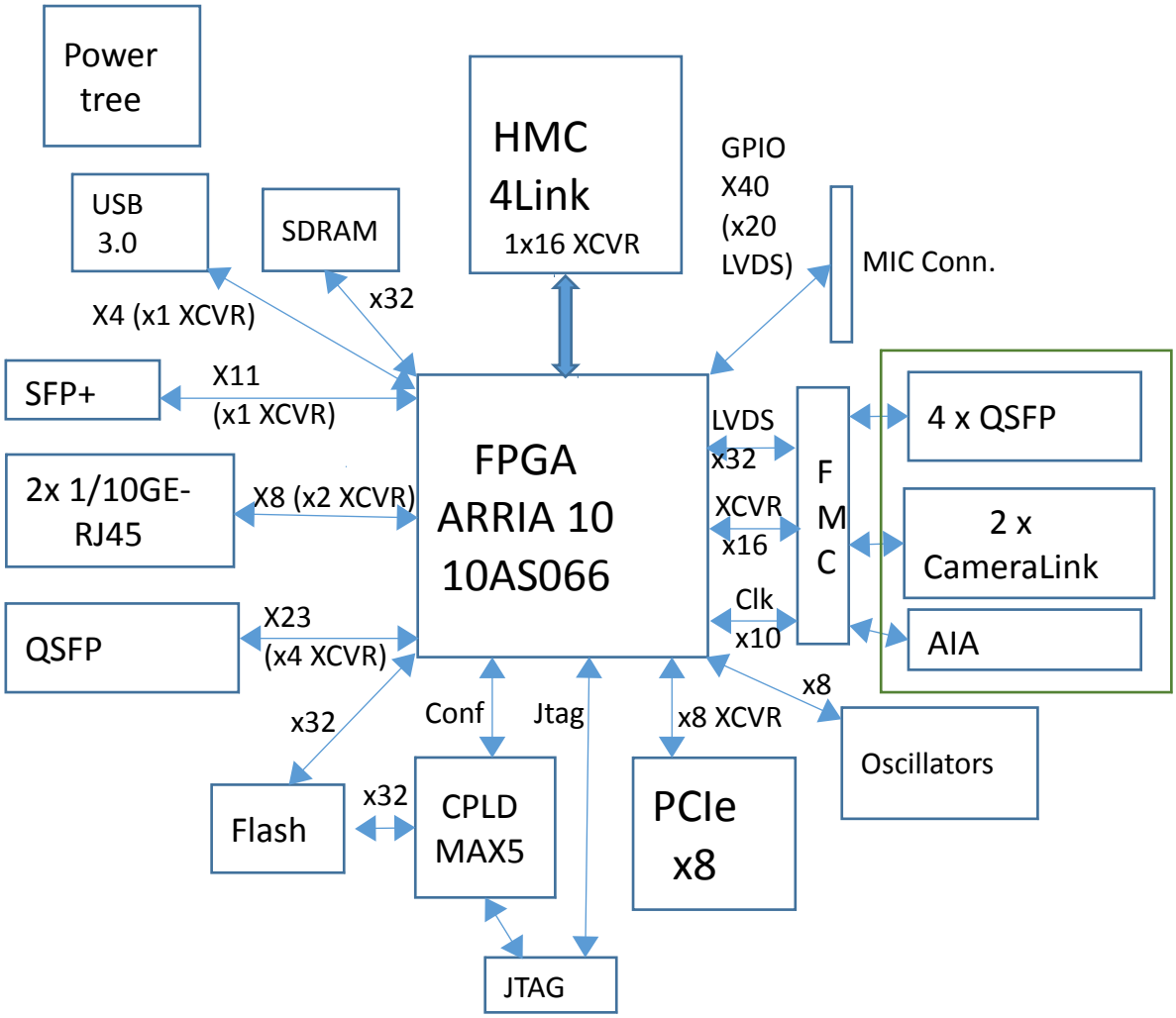

Figure 2: Block-diagram of FPGA with ARM coprocessor scheme

This scheme is similar to the previously presented scheme in chapter 2.1. The main difference is the interconnection between FPGA and HMC. Because of the smaller amount of available transceivers (max. 48 for the 10AS066) only one link with 16 lanes can be implemented, which reduced the bandwidth between FPGA and the external HMC memory by a factor 4. This will reduce the real-time performance of the reconstructor because of the bottleneck of transferring the matrices from the HMC to the FPGA.

On the other hand, the dual-core ARM Cortex-9 coprocessor allows building a stand-alone system with a powerful operating system and the possibility to act as a Host with PCIe root port. This opens the possibility to offload part of the real-time computation or all of it to additional accelerator cards e.g. GPUs or other FPGA board and therefore overcome the drawback of the bottleneck between FPGA and HMC. In

this constellation, the FPGA board acts mainly as interface board to connect to the WFS and to the DMs, while the computational part will be outsourced.

The interface are the same as in the previous scheme and are described in more detail in chapter 3.

A Flash and CPLD MAX5 are used to store and configure the FPGA at power-on and the CPLD can as well take care of the housekeeping functionality of the board e.g. monitoring temperatures, power consumptions controlling fans etc. A SDRAM chip is added for using it with the soft-core NIOS II processor that can be instantiated in logic inside the FPGA.

2.3 Two FPGA System

The third scheme is a combination of the two previous scheme to profit of the advantages of both. This scheme is illustrated in figure 3.

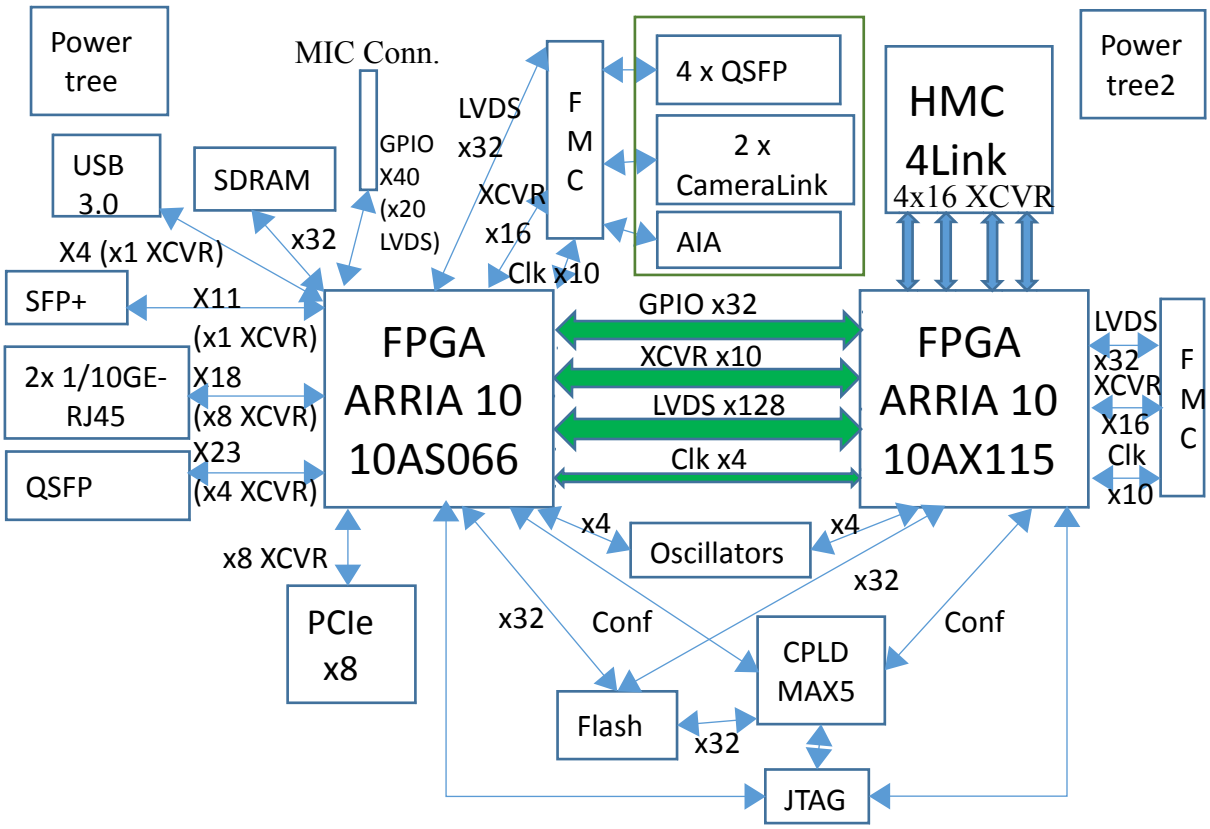

Figure 3: Block-diagram of the two FPGA scheme

This system contains two FPGA devices, one SoC with an ARM coprocessor and one GX type. This allows exploiting the full bandwidth between an external memory (HMC) and the

GX FPGA, obtaining a powerful real-time computational unit, while implementing all relevant interfaces with the SoC AS FPGA and using its powerful ARM coprocessor. This coprocessor can be used e.g. for a Host system in a stand-alone implementation. Between the two FPGAs a fast interconnect will be realized to send the input vectors to the computational unit and get back the result vector.

The combination of the two previously described schemes in one board clearly increases the complexity of routing, power supply and cooling but this is not a showstopper as confirmed by several PCIe boards with two FPGAs of other families already available from different vendors. This scheme requires increasing the board length to Full Length PCIe standard. It will still fit in server racks or PCs that provide full spacing PCIe slots and it will be comparable to the size of modern general purpose GPUs for HPC.

2.4 On board computational power considerations

In order to evaluate the achievable performance we have performed a real test case based on the E-ELT M4 DM computational requirements. The underlying calculation is targeting a MVM with a matrix size of 5316x5316.

In a first step, an implementation of a MVM in an ARRIA 10 development board was carried out by Microgate showing excellent performances focusing on a downsized M4 prototype. A matrix of 222 rows and 5316 columns of single precision floating-point values was multiplied with a 5316 vector of single precision floating-point values in less than 30µs. This leads to a computational throughput of 40 GMAC/s by using 222 DSP blocks inside the FPGA. In theory, considering that the ARRIA 10 have more than 1500 DSP blocks this throughput could be scaled up to 280 GMAC/s. This is only true when the matrix data can be provided fast enough to the DSPs inside the ARRIA 10 because the ARRIA 10 has not enough internal memory to store the 5316x5316 matrix. The matrix data have to be saved in an external memory that has a fast interconnection to the FPGA such as the HMC. With a full bandwidth of the HMC of 120GB/s, and in addition using as well internal FPGA memory, about 120µs will be required to load the matrix of one M4 segment (5316x886) into the FPGA during calculation. This takes longer than processing the MACs inside the FPGA and therefore this time is dominant whereas the calculation will be performed contemporaneous to the reading of the matrix. The reduction of the calculation throughput caused by the memory bottleneck is significant and results in about 40GMAC/s per board. This is comparable to GPU performances for MVMs.

2.5 Tradeoff

A tradeoff between these three schemes will be presented in this chapter. Based on this tradeoff, a decision will be taken in the early next phase of the GreenFlash project to implement the FPGA based hard real-time data pipeline. In table 2, the characteristics of the three solutions are summarized and evaluated. The results should be considered absolutely

preliminary.

Table 2: Tradeoff table for the three implementation schemes

		FGPA only	SoC - FPGA	Dual FPGA
Argument	**Weight**	**Mark**	**Mark**	**Mark**
Interfaces	5	5	5	5
On board Computation power	4	4	2	5
PCIe root port and ARM for stand-alone	4	2	5	5
Power efficiency	3	2	4	5
Routing complexity	2	2	3	1
Cost	3	3	4	2
Size	2	3	3	1
Weighted sum		**74**	**89**	**90**

3 Interfaces

In this chapter a closer look to the planed interfaces on the FPGA based hard real-time board is given. An important requirement of the GreenFlash project is that it is highly flexible in interconnecting with different camera instruments and DMs. Therefore, a variety of different high-speed interfaces is foreseen on this board.

As described in the interface document AD08, for the WFS camera interfaces GreenFlash will point mainly on 1Gb and 10Gb Ethernet links. Other camera interface as AIA or CameraLink can be as well supported by this board using additional expansion (daughter) cards as described in the following. For the interface versus the DMs GreenFlash will take a decision as soon as additional information from E-ELT arrives. Microgate in their past AO projects has implemented sFPDP interfaces to the DMs and this board can realizing a large number of such sFPDP connections.

Two main sets of interfaces of the board can be distinguished, the ones that are accessible on the PCIe front-panel and the ones that can be implemented later on via expansion boards connected to connectors on the board. In addition, at the bottom of the board a PCIe 8 lane connector is placed that allows inserting the board as a PCIe endpoint card up to Gen3 x8 in an existing Host computer. With an additional adapter (e.g. PCIe crossover cable) that

converts the male to female connector, this PCIe board can be used as well in a stand-alone environment, where it acts for example as a PCIe root with a PCIe riser card attached that has several PCIe slots to insert e.g. GPUs.

Next, the two main sets of interfaces are described in more detail.

3.1 Front-panel interfaces

The interfaces that are accessible from the PCIe front-panel can be seen in figure 4.

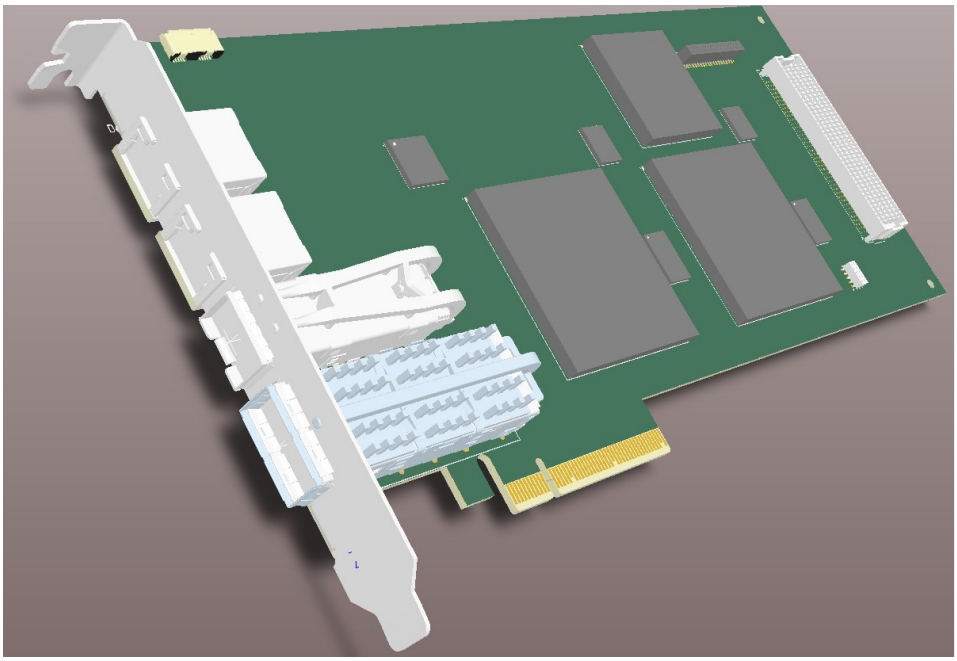

Figure 4: PCIe Front-Panel connectors

On top of the front-panel, two RJ45 connectors are placed. These can be used to attach standard Ethernet cables in order to realize 1/10Gb Ethernet connections.

Next, a SFP+ receptacle is placed that allows attaching fiber channels for 10Gb Ethernet. In addition the SFP+ can be used as well to connect an additional 1/10Gb Ethernet over copper (direct attach) or one sFPDP using SFP+ to FPDP adapters.

At the bottom of the front-panel, a QSFP receptacle is implemented to realize Infiniband interfaces, or a 40Gb Ethernet using the 4 channels of QSFP, or four times sFPDP.

3.2 Additional interfaces via FMC

Additional interfaces can be implemented by connecting daughter cards to the FPGA board using the FMC connector. It should be remarked that the development of such daughter cards is out of the scope of the GreenFlash project. This connector can be seen (in white) at the

right end of the board in figure 4. The FMC allows to connect a large variety of interconnects to the FPGA by using appropriate daughter cards that contain the desired interfaces. This connector contains 16 transceiver links, 32 LVDS lines that can be used as well as 64 Programmable I/Os and some clock signals. More concrete, this would allow implementing additional 4 QSFP receptacles for example connecting the board via sFPDP to large DMs. For the connection to WFS, CameraLinks, or AIA interfaces can be attached and many more interfaces are possible. For the third scheme, a second FMC connected to the second FPGA can be foreseen to obtain even more interconnection possibilities.

An additional custom light connector is as well foreseen on the board, to attach eventually small interface boards using for example flat cables to expand to additional PCIe front-panels. Another way to use this small connector is to attach LEDs or user buttons or expose internal signals from the FPGA for test purposes or debugging tasks.

4 Software considerations

Each of the presented hardware configurations have at least one processor. For the software running on those processors we can see 3 different options:

- No operating system at all, just implement a main loop and various interrupt service routines. This solution has been adopted in the past by Microgate on its own boards. With this solution, a simple and basic interface to the board components has been achieved in a straightforward way. This approach would make sense for an embedded NIOS II processor inside the FPGA. For a more powerful ARM processor this is probably not the right approach.
- MicroC OS. This is a portable, ROMable, scalable, preemptive, real-time deterministic multitasking kernel for microprocessors, microcontrollers and DSPs. MicroC is delivered with complete 100% ANSI C source code and in-depth documentation and runs on the largest number of processor architectures, including NIOS II and ARM processors. It manages up to 250 application tasks including: semaphores, event flags, mutual-exclusion semaphores, message mailboxes and queues, time and timer management, and fixed sized memory block management. This is an interesting solution if some additional features like a TCP/IP stack are required. It is simple to use and can accomplish a variety of real time tasks.
- Finally, the third option is to adopt a Linux OS. While it would be possible to run Linux on an embedded NIOS II processor, this option becomes really interesting when adopting a more powerful ARM processor. With respect to MicroC OS, Linux comes with a variety of drivers and services for networking, file system, web server and so on. If our board shall become a stand-alone system and act as the host, than Linux OS will probably be the better choice. In this scenario, it would initialize other boards on the system, for instance other GPU boards, it would act as the PCIe root complex. For what concerns the specific distribution to be used if we like to remain on a widespread distribution, Debian could be a viable option, giving big flexibility and providing a large selection of development tools and libraries. While Debian supports also the ARM processors, we doubt that it will run out of the box on our

system without any customization. For this reason, to be more independent, and for the long term, using the Yocto Project Build system would probably be the best choice.
The Yocto Project is an open source collaboration project that provides templates, tools and methods to help you create custom Linux-based systems for embedded products regardless of the hardware architecture. Using these tools, we could build a custom distribution. This is the typical way to proceed for embedded products.
For what concerns real time demands on the OS, a RT Linux kernel or RTAI kernel (for ultimate performance) can be used.

As conclusion, we can say that if we use a SoC a Linux OS is probably the right choice while for an embedded NIOS II processor MicroC OS would also be a good option. If we want to turn our system into a stand-alone system SoC and Linux is probably the only way to go.

5 Summary

The proposed FPGA board will allow implementing the hard real-time data pipeline for the GreenFlash project in a highly efficient and flexible way. Many interfaces are foreseen to connect the board to cameras, DMs, Host systems and other accelerator cards. This board will fulfill the interface requirements R7.24 described in AD03.

Especially the PCIe interface can be used to connect the board with other PCIe cards. In this context, the Microgate board can be used for example as interface board and connect to other FPGA based accelerator cards or GPU cards that perform the real-time calculations. In a stand-alone environment, the board can interface to the cameras and DMs and by implementing a PCIe root port it connects via PCIe switches to several accelerator cards, allowing the realization of a hard real-time box.

On the other hand, the board itself can be used as an accelerator card to perform real-time calculations in combination with the novel memory type HMC that allows a large data throughput compared to its small area requirement. In particular, in the dual FPGA version, the board can be a very efficient and valuable solution to implement in a single card both interfaces and computational task required by relatively simple problems, like some SCAO implementations.

Considering an on board computational power of 40GMAC/s we could fulfill the 1.5TMAC/s requirement R9.30 described in AD03 by using about 38 boards. This is comparable to the considerations of using 45 GPUs as described in AD01. The advantage of the FPGA boards is that they provide more interfaces, are more compact in size and achieve by architecture a very deterministic time behavior. This would help e.g. to fulfill the tight jitter requirement of less than 100µs as given in R9.50 of AD03. A latency requirement of 2ms as stated in R9.40 in AD03 seams as well to be feasible considering a computation time of a few hundreds of microseconds and high-speed interconnection possibilities. On contrary the advantage of using GPUs in combination with one of our boards as interface card, is the reduced costs in terms of hardware and development effort.

In general, the Microgate board can be used either as an interface board, as FPGA based accelerator card, or both at the same time.