2017 Euromicro Conference on Digital System Design

FPGA-Centric High Performance Embedded Computing: Challenges and Trends

Rabie Ben Atitallah and Karim M. A. Ali
University of Valenciennes and Hainaut-Cambrésis
LAMIH UMR CNRS 8201, Computer Science Department
Email: rabie.benatitallah@univ-valenciennes.fr, karim.ali@univ-valenciennes.fr

***Abstract*—Sophisticated embedded systems are increasingly used in defence, aerospace and avionic industries. They are responsible for control, collision avoidance, pilot assistance, target tracking, navigation and communications, amongst other functions. In this industrial field, High Performance Embedded Computing (HPEC) applications are becoming highly sophisticated and resource consuming for three reasons. First, they should capture and process real-time data from several I/O sources in parallel. Second, they should adapt their functionalities according to the application or environment variations within given Size Weight and Power (SWaP) constraints. Third, since they process several parallel I/O sources, applications are often distributed on multiple computing nodes making them highly parallel. The problem with current HPEC systems is that, they are usually built to meet the needs of a specific application, i.e., lacks flexibility to upgrade the system or reuse existing hardware resources. Due to the hardware parallelism and I/O bandwidth offered by Field Programmable Gate Arrays (FPGAs), application can be duplicated several times to process parallel I/Os, making Single Program Multiple Data (SPMD) the favorite execution model for designers implementing parallel architectures on FPGAs. Furthermore Dynamic Partial Reconfiguration (DPR) feature allows efficient reuse of limited hardware resources, making FPGA a highly attractive solution for such applications. This paper will address HPEC application design challenges and will discuss some solutions relying on FPGA technologies and referring to several industrial collaborations.**

I. Introduction

High Performance Computing (HPC) is entering the world of embedded systems and giving rise to a whole new field of High Performance Embedded Computing (HPEC). HPEC arises from the need for high throughput computations with Size Weight and Power (SWaP) constraints, acquire data from several parallel sensor sources with minimum computation and communication latencies, with requirements to operate under harsh and uncertain environments in embedded systems. HPEC systems are deployed to perform sophisticated tasks such as collision avoidance, driver assistance, target tracking, motion detection, path planning and navigation among others. In all these applications, simply stated, parallel data acquisition and processing in real-time drives the need for HPEC systems. Although the standard definition of HPEC still remains elusive, HPEC industries refer to it as: cutting edge technology with parallel processing techniques for running complex applications with large clusters of processing nodes with high-speed data acquisition and low-latency communication fabrics.

The reasons for such requirements lies in the operating principles of HPEC systems that are described as follows. The first paradigm of HPEC system design is to maximize the performance (in number of Operations per second) per cubic inch of hardware resource, while minimizing on SWaP requirements. The reason for this principle is that, HPEC systems are deployed in-field where there is a lot of computational demand, but under tight area, power and resource constraints. For instance, RADAR applications deployed for border surveillance requires extended operating time with typical computational throughput that ranges from 100 to 1000 billion operations per second with parallel hardware processing power.

Second paradigm in building HPEC systems is to minimize the communication and computational latencies of the system in parallel sensor data processing, that plays a vital role in the design of a good HPEC system. This is due to the fact that any unnecessary overhead may result in non-deterministic system behavior. An example of a highly time critical HPEC system would be anti-aircraft defence systems that intercepts the fired missiles. In such systems, even the slightest overheads can cause a lot of causalities. There are multiple sensors such as detecting, target tracking, impact analysis and anti-missile deployment, etc., involved at the same time.

The final paradigm of a HPEC system design is that the computing hardware should satisfy SWaP constraints of being smaller in size, lighter in weight with small batteries. In order to achieve that, the space of design alternatives is explored for the design which give the best trade-off between the design constraints. The process of design exploration is accelerated by using productivity tools like High-level synthesis tools to separate algorithm from implementations with the possibility to estimate the design constraints (utilization, execution time, power consumption, etc) at that level then full implementations are conducted to the candidate designs only.

In this context, FPGAs (Field Programmable Gate Arrays) have emerged as a privileged target platform for HPEC applications [1]. FPGAs have the benefits of high speed and adaptability to the application constraints with an improved performance per watt comparing to the General Purpose Processors (GPP). In addition, FPGA devices can be used as a generic communication as well as computation-centric platform since they can be programmed with the necessary I/O interfaces [2], [3]. Furthermore, the FPGAs today such as Xilinx Ultrascale+ allow the implementation of massively parallel architectures due to the very nature of the hardware and technology. This makes Single Program Multiple Data (SPMD) as a favorite execution model for designers while implementing parallel architectures on FPGA. In addition, Dynamic Partial Reconfiguration (DPR) feature allows runtime

978-1-5386-2146-2/17 $31.00 © 2017 IEEE
DOI 10.1109/DSD.2017.88

customization of hardware resources to adapt to application constraints. In this particular context, where applications demand unlimited parallel computing power, high I/O bandwidth, low communication latencies, under tight SWaP constraints, we believe that the convergence of massive parallelism and dynamic reconfigurable hardware will help to build adaptable, reusable, resource efficient next generation computation platforms for HPEC applications. In this paper, we will identify challenges and promising solutions for building a scalable and flexible reconfigurable HPEC system dedicated to massively parallel applications with real-time and SWaP constraints. Several industrial examples will be given and discussed to argue the pertinence of using reconfigurable technology.

In this paper, we will present in Section II the challenge of providing on-demand parallel architectures dedicated to HPEC applications in order to offer high computation rates with respect of SWaP constraints. Thereafter, Section III details technological barriers and solutions to provide high-speed low-latency communication interfaces combined with hardware flexibility. Finally in Section IV, we propose adequate design methodology for HPEC applications in order to shorten the time-to-market and to increase the design productivity.

II. On-demand parallel architectures

Modern HPEC systems such as the Unmanned Ariel Vehicles (UAVs), use hardware chips to implement their real-time path planning algorithms [4]. The complexity of such algorithms requires the use of several parallel on-board data processing to achieve performance levels of tens of Giga FLOPS (GFLOPS), particularly because these systems must often process the data in real time. Furthermore, dealing with uncertain environments implies that the system has to be equipped with more on-board hardware and software resources to adapt during missions. Thus the ever increasing processing requirements and dealing with uncertainty, places more stress on resource requirements. In this context, hardware adaptability and re-usability will go a long way in minimizing hardware resources and to achieve sufficient computational throughput within a given set of SWaP constraints.

We address the previously described challenges of building a scalable and flexible HPEC systems at two levels. First, we address them at the package level. Indeed, we have designed and manufactured a new scalable and flexible multi-FPGA architecture with switched fabric communication and customizable FPGA modules as shown in Fig. 1. We have fully utilized the capabilities of a PCIe switch in our software support to provide SPMD execution model, to deal with parallel I/O sources. In our architecture, there can be a maximum of up to 4 FPGA modules. The number of modules have be limited in order to stay within a given form factor. Each FPGA is designed as a module and can be removed, upgraded or replaced in order to offer customizable computing power according to HPEC application requirements. Such modular Multi-FPGA board could be packaged in a small form factor satisfying SWaP constraints.

Second, we address challenges of building a scalable and flexible architecture HPEC systems at the FPGA circuit level. In this work, we have invested in research and development of parallel hardware-based architecture for video streaming-based embedded systems guided with a power-aware design

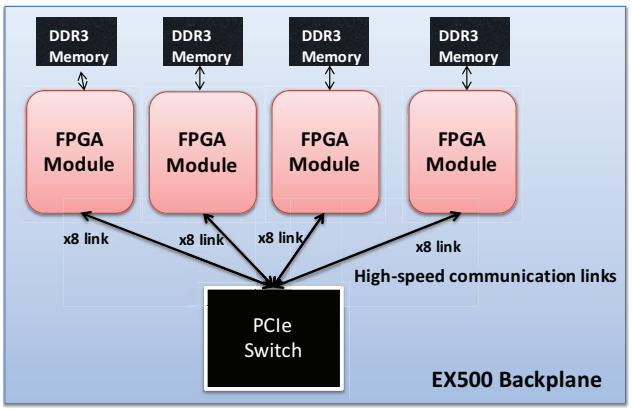


Fig. 1: Modular Multi-FPGA board

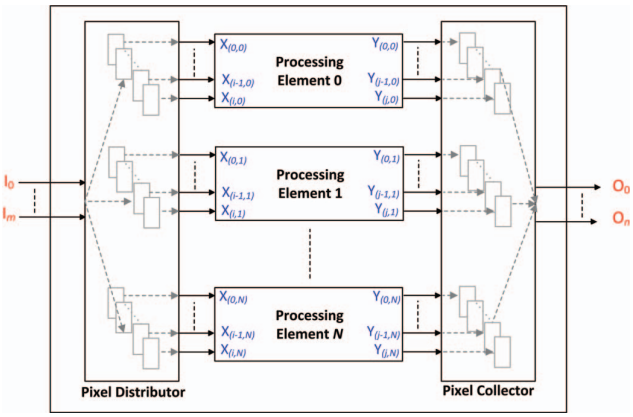


Fig. 2: Example of on-demand parallel architecture for video processing applications

criteria in addition to the satisfaction of the performance constraint. Mainly, we target reconfigurable technology to propose a flexible parallel system where the designers can adapt the parallelism level according to the available resources in order to control the overall system power consumption. Figure 2 represents a video processing architecture where N processing elements are running in parallel. Each processing element has i input ports (X_0, X_1, ..., X_i) and j output ports (Y_0, Y_1, ..., Y_j). The input pixel streams (I_0, I_1, ..., I_m) are copied and distributed to individual array structures through *Pixel Distributor*. While *Pixel Collector* stores the processed pixels then streams them out in order through system output ports (O_0, O_1, ..., O_n) [5].

The parallelism level can be flexible in conjunction with the frequency scaling as a technique for reducing power consumption. The chosen level of parallelism with a certain clock frequency scaling will offer several design choices leading to different trade-offs in terms of hardware cost and power consumption. As example of experimental results, we studied the implementation of a video downscaler (1:16) application, an HD frame of size 1920x1080 was scaled down to one sixteenth of its size to be 480x270 at a rate of 60 frames/s.

Table I listed a set of different design points. These points

Design point	Level of parallelism	CLK1 (MHz)	CLK2 (MHz)	FIFO depth
Video Downscaler (1:16) Application				
D1	1	148.5	148.5	0
D2	1	148.5	74.25	242
D3	1	148.5	37.125	362
D4	2	148.5	74.25	2
D5	2	148.5	37.125	242
D6	2	148.5	18.5625	362
D7	4	148.5	37.125	2
D8	4	148.5	18.5625	242
D9	4	148.5	9.28125	362

TABLE I: The design points for video downscler (1:16)

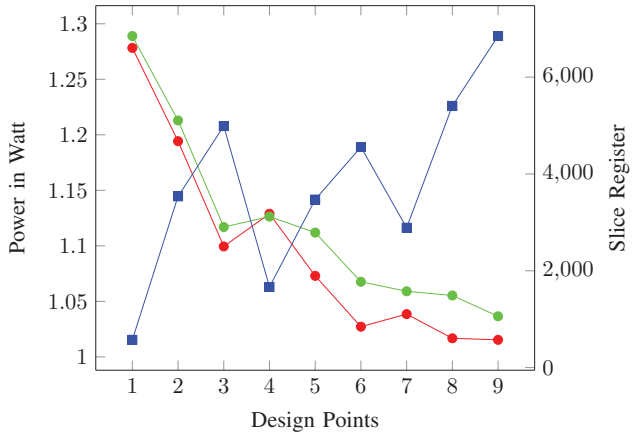


Fig. 3: The trade off between the estimated power —●—, the measured power —●— and the slice register cost —■— for each design point for video downscaler

are obtained by varying the level of parallelism or the operating frequency. The design point D1 is considered as the reference design point because it has the minimum required level of parallelism as well as it operates at the clock frequency (i.e. CLK=148.5 MHz) corresponding to the rate of 60 frames/s.

In Fig. 3, the estimated and measured power for video downscaler application was plotted against the number of slice register required for each design point. Experimentally, the power consumption decreased from 1.29 W for D1 to be 1.04 W at D9 with a percentage power reduction equal to 19.6%. According to the available register resources, the designer can select which design alternative to use and what percentage decrease in power to gain. For example, the percentage power reduction for D7 was 17.8% at register cost = 2889 and for D6 was 17.1% at register cost = 4557 so D7 is always better than D6 since it achieved more power reduction at lower register cost. Also, we can consider D7 as a design choice better than other points like D8 or D9 because the percentage decrease in power between these points and D7 is not so significant (0.3% for D8 and 1.7% for D9) if compared to the percentage increase in the register cost (87% for D8 and 137% for D9) [5].

III. High throughput reconfigurable I/O

In this section, we will highlight how FPGAs can provide high-speed low-latency communication interfaces combined with hardware flexibility in order to satisfy the requirement of HPEC applications derived from defence, aerospace or avionic industries.

Indeed, depending on the sensors deployed, the communication fabrics supported on an HPEC system are often a mix of analog and digital interfaces such as ADC, MIL-STD-1553, ARINC, CAN, sFPDP, Camera-link, Ethernet, RapidIO, AFDX and ATM [6]. A typical system may employ any combination of different number of sensors and different types of protocols. Since the system can have any number of sensor channels and any number of protocols, software overheads such as context switch and interrupts must be minimized or avoided altogether while sampling several sources, to ensure deterministic system behavior [7]. Unfortunately, systems based on microprocessors are restricted only to interfaces that are provided on-chip, such as PCIe, Ethernet, etc. However, FPGAs provide high-speed SERDES [**?**] links that can be configured as a wide range of high-speed low-latency I/O interfaces on a single chip based on the requirements. It has been shown that a single FPGA chip can be used both as a computation and communication centric platform in avionic applications [8], implementing multiple communication protocols on a single chip. When this I/O flexibility is combined with directly coupled high-speed DDR memories and reconfigurable SRAM based computational power, FPGAs can be a powerful tool for front-end signal processing providing high levels of determinism i.e., fixed cycle delays. As example, a 64-channel Beamforming application is implemented on 3 Virtex7 FPGAs using high-speed GTX transceivers providing high-bandwidth (8 x 16 Gb/s) and low communication latencies [9].

Industrial HPEC systems such as phased-array radars, sensory arrays and video processing, are becoming highly sophisticated and distributed as they capture and process real-time data from several sources at the same time [10]. FPGAs are good candidates to perform such sophisticated and intensive processing. Capturing and processing data from several sources at the same time requires different I/O interfaces or several I/O channels. Designing the I/O interfaces along with the rest of the PCB causes obsolescence of I/O interfaces as requirements change. Thus FPGA Mezzanine Card (FMC) standard has evolved to be used along with FPGAs, to provide I/O modularity in reconfigurable data acquisition systems. The above is a direct consequence of two things:

- first, different applications use different I/O interfaces and different number of channels for data acquisition from several sources. Thus when applications evolve, the data sources change along with the protocols and the corresponding I/O interface. Since FMCs are designed as a modular I/O interfaces, separately from the rest of the PCB system, the I/O interface can be swapped independently of the rest of the system;
- second, FMCs are I/O interface standards created especially to be used with FPGAs. This is because, the interest in reconfigurable embedded high performance computing in the defence industry and aerospace market has grown due to the various above mentioned reasons. New generations of FPGAs are able to provide a level of processing performance with high I/O bandwidth and thus closing the gap in the I/O bottleneck.

In order to address the changing I/O requirements, FMC (ANSI/VITA 57.1-2008) [**?**] has evolved over the years for solutions that require the benefits of an FPGA unlike the other standards [**?**]. The purpose of the FMC specification is to allow (usually) one FPGA on a host card to connect directly with the I/O devices on the mezzanine module just as if the device were on the host board. This simple and elegant modular solution in combination with a programmable hardware (FPGA) not only provides high bandwidth low latency communication, but also eliminates I/O obsolescence. The FMC specification provides for a large number of differential connections up to 80 pairs or 160 single-ended signals to support high speed parallel interfaces between the FPGA and I/O devices. There are also a number of serial connections (up to ten pairs) suitable for Multi-Gigabit Transceivers (MGTs) operating up to 10Gb/s. FMC modules and hosts support two connector options; a Low Pin Count (LPC) 160 pin connector or High Pin Count (HPC) 400 pin connector. The connector pins are generically defined for power, data, control and status with specific implementation depending on the design. The majority of FMC solutions are likely to use the HPC variant. Although aimed at I/O, FMC can be used for any function that might connect to an FPGA including DSPs, memory or even another FPGA. A FMC module is mounted on a carrier board shown in Fig. 4 (left). The FMC standard forms a parallel I/O acquisition system in our architecture.

In collaboration with Nolam Embedded Systems [**?**], we have implemented several communication protocol (ARINC429/CAN Bus/MIL-STD-1553) via a FMC interface. The communication protocol is selected and configured during runtime according to the request. Ideally, the data have to be transmitted to external sub-systems. However, for testing purposes, we have done an external FMC loopback to verify if the transmission is correct as shown in Fig. 4 (left). After that, we analyze the FPGA resource utilization, transmission characteristics, I/O pin requirement and scalability for each core. The results are elaborated in [3]. In this paper we give an overview of the MIL-STD-1553 I/O IP core description and a comparison with other protocols while transmitting terrain pictures taken from UAV avionic system.

MIL-STD-1553: The MIL-STD-1553 is originally serial military standard protocol that defines the mechanical, electrical, and functional characteristics of a serial data bus. It is now also being used in spacecraft On-Board Data Handling (OBDH) subsystems, both military and civil. The architecture of the bus system consists of a Bus Controller (BC) controlling multiple Remote Terminals (RT) all connected together by a data bus providing a single data path between the bus controller and all the associated remote terminals. The RT is used to interface with other user defined subsystems. There can also be one or more Bus Monitors (BM); however, they are not allowed to do any data transfers, and are only used for recording the data for analysis. The protocol also supports several data buses to provide multiple redundant data paths upto a maximum of 4. The protocol follows very strict timing constraints and requirements and provides a maximum bandwidth of 1 Mbps. We have developed our own IP core according to the MIL-STD-1553 specification [**?**] and used it to evaluate our system. The architecture of the MIL-STD-1553 is given in Fig. 4 (right).

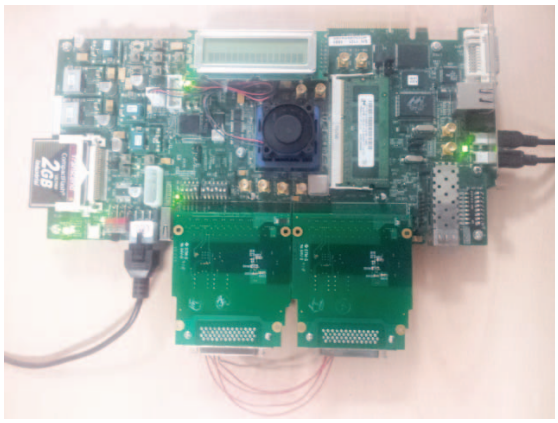
As example of results, Fig. 5 shows the performance of our IP cores in terms of time taken for transmitting different number of frames. The system performance has been evaluated up to 1000 frames. The transmission and reception is done synchronized according to the protocols' internal clocks in pre-programmed frequencies. However, the time measured, also takes into consideration the overhead of configuring the registers, the overhead caused due to the transfer of status words and the idle time between each transaction. From the graph shown in Fig. 2, it is seen that MIL-STD-1553 is the fastest in transmitting the frames. Although MIL-STD-1553 and CAN Bus have the same maximum bandwidth, the fact that MIL-STD-1553 is able to pack more data into a single message and to operate with minimal status feedback, gives it an extra edge in transmitting efficiently.

IV. SWaP CONSTRAINTS

During the design phase, it is not feasible to satisfy all the design constraints to the same extent but there is a trade-off between them. For example, it is not practical to increase the system performance without increasing the number of utilized resources. Consequently, the priority of constraints could vary from one design case to another. For example, hardware resources matter if several functionalities would be embedded on the same chip. Quality-of-Service is important for interactive or multimedia applications while timing is crucial for safety critical applications. SWaP (Size, Weight and Power) constraints have the priority in military and aerospace applications in order to build systems characterized by smaller footprint, lighter weight with smaller battery-size. In addition to the former constraints, business constraints like time-to-market, non-recurring engineering costs, project budget, etc. could influence the design phase.

Taking these constraints into consideration involves a multi-objective design problem with a space of design alternatives. During the design phase, it is the role of the designer to define the priorities of system constraints then to explore the design space for the implementation that efficiently satisfies them. Fast converging to the set of the desired designs is recommended. This could be achieved by performing quick estimations for area, power, performance, etc. at higher design levels then conducting full implementations for the candidate designs only. For that purpose productivity tools in FPGA design play a key role in reducing the design efforts.

High-level synthesis (HLS) tools emerged in Electronic Design Automation as an step in the design flow aiming at moving the design efforts to higher abstraction levels [11]. Regardless of the first failure of the former generations of HLS tools; different reasons have motivated researchers to continue improving these tools. We can mention among these reasons [12] : (i) The huge growth in the silicon capacity pushes towards using HLS tools. (ii) Design productivity is enhanced by reusing behavioural IPs instead of RTL IPs which have fixed architecture and interface. (iii) Recent designs tend to widely use accelerators and heterogeneous SoCs. (iv) Time-to-market constraint usually presses to reduce the design time by avoiding long chip design process. Today several existing HLS tools have shown their efficiency for producing acceptable design performances and shortening time-to-market [11] [13]. In fact, designing using HLS tools is not a straight forward



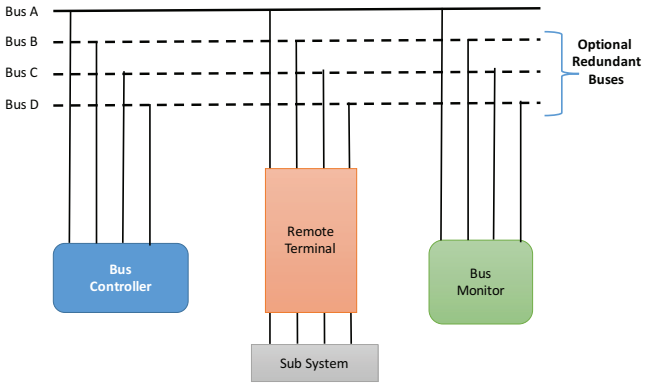


Fig. 4: A MIL-STD-1553-based FMC module mounted on a carrier board (left) and the MIL-STD-1553 bus architecture (right).

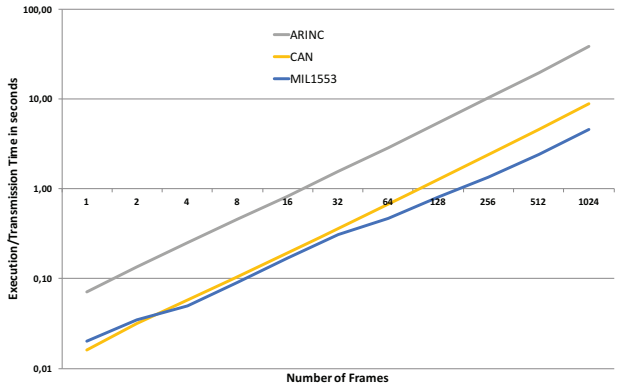


Fig. 5: Transmission time vs Number of frames

transformation of behavioural descriptions into RTL designs but a set of optimization steps are involved at that abstraction level in order to obtain an efficient hardware implementation [14].

Fig. 6 shows the steps for how the design space is explored for parallel video processing architectures. Two input files which are: (1) *Processing element file* which implements the functionality of the processing element described in behavioural language. (2) System specification file which describes the parallel architecture depicted in Fig.2 . For example, it defines the size of input/output images, what level of parallelism to build and how the input pixel stream is distributed over the parallel processing elements. In the next step, high level description code for parallel architectures are automatically generated at different parallelism levels. After that, estimations for HW resource utilization, execution time and power consumption are calculated to predict the performance of each design alternative. Based on the design constraints, the design space is trimmed to keep only the set of candidate designs for full implementation. later, candidate designs are experimentally tested and verified to select the required design.

In collaboration with NAVYA company, we used the previous described design flow to implement Multi-window Sum of Absolute Difference (Multi-window SAD) stereo matching algorithm. A stereo vision system of two cameras is used to determine the depth of objects in front of the camera. Figure 7 depicts some of the candidate designs (#1, #2, #3, #4 and #5) along with the system constraints to guide the designer towards the efficient solution. The orange shaded area represents the system constraints defined by the designer which are: power consumption $\leq$ 2 W, execution time $\leq$ 15 ms, LUT $\leq$ 180000, FF $\leq$ 140000, BRAM $\leq$ 700 and frequency $\leq$ 150 MHz. From Fig. 7, we could deduce that design #3 succeeded to satisfy all the system constraints. Design #1 had relatively less hardware utilization and acceptable execution time in compare with design #3; however, it failed to meet two design constraints (power consumption and frequency).

V. Conclusion

In this paper, we have addressed the design challenges of High Performance Embedded Computing (HPEC) applications derived from defence, aerospace and avionic industries. First, we have elaborated the challenges of such domain related to the requirements in terms of hardware, communication and design methodology taking into account the SWaP constraints. After that, we have exposed the scientific and technological trends to tackle these challenges relying on reconfigurable technology and referring to several industrial collaborations. At the hardware level, we have presented our approach on how to have a parallel, scalable, and flexible system to customize the hardware based on SWaP constraints. At the communication level, we have detailed the design of high throughput reconfigurable IP protocols in order to address I/O obsolescence and we have analysed its benefits on the modularity of the system. Finally, we have presented the design space exploration flow to select the design which better satisfies our design constraints.

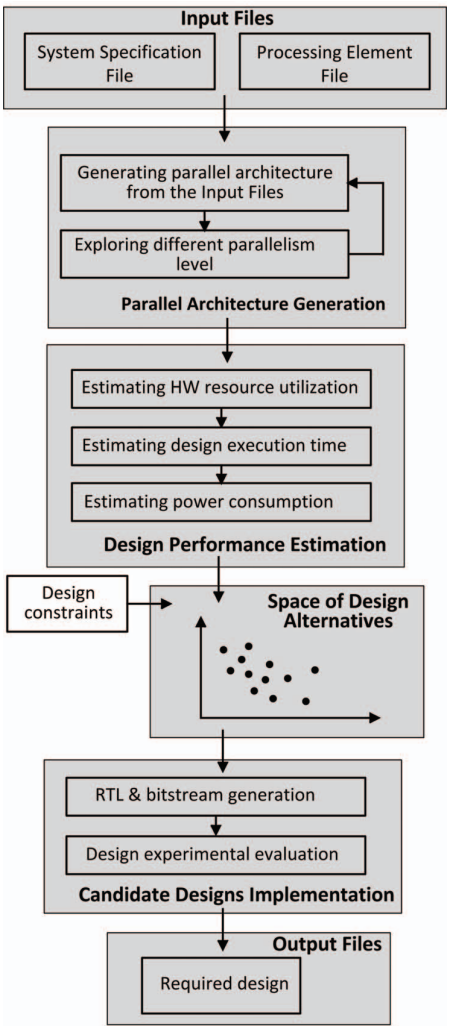


Fig. 6: Design Space Exploration flow

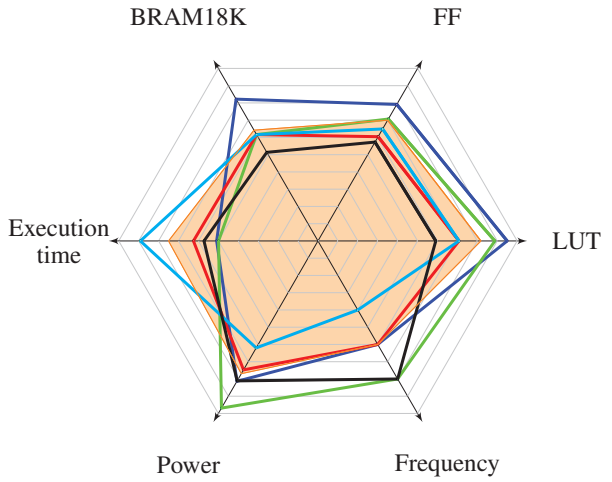


Fig. 7: Radar chart for designs #1 ▬, #2 ▬, #3 ▬, #4 ▬, #5 ▬ and system constraints ▬

REFERENCES

[1] N. Tredennick and B. Shimamoto, "The inevitability of reconfigurable systems," *Queue*, vol. 1, no. 7, pp. 34–43, Oct. 2003. [Online]. Available: http://doi.acm.org/10.1145/957717.957767

[2] M. Bouain, V. Viswanathan, R. Ben Atitallah, and J.-L. Dekeyser, "Communication-centric design for fmc based i/o system," in *Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC), 2014 9th International Symposium on*, May 2014, pp. 1–8.

[3] V. Viswanathan, R. Ben Atitallah, J. Dekeyser, B. Nakache, and M. Nakache, "Dynamic reconfiguration of modular I/O IP cores for avionic applications," in *Reconfigurable Computing and FPGAs (ReConFig), 2012 International Conference on*, Dec 2012, pp. 1–6.

[4] F. Allaire, M. Tarbouchi, G. Labont, and G. Fusina, "Fpga implementation of genetic algorithm for uav real-time path planning," *Journal of Intelligent and Robotic Systems*, vol. 54, no. 1-3, pp. 495–510, 2009. [Online]. Available: http://dx.doi.org/10.1007/s10846-008-9276-8

[5] K. M. A. Ali, R. B. Atitallah, N. Fakhfakh, and J. L. Dekeyser, "Using hardware parallelism for reducing power consumption in video streaming applications," in *2015 10th International Symposium on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC)*, June 2015, pp. 1–7.

[6] C. W. Company, "Cw defense products." [Online]. Available: http://www.cwcdefense.com/products/subsystems/program-specific-subsystems/smu.html

[7] J. NEWPORT, "Avionic Systems Design," in *Avionic Systems Design*, ser. CRC Press, 1994, pp. 219–220.

[8] V. Viswanathan, R. Ben Atitallah, J.-L. Dekeyser, B. Nakache, and M. Nakache, "Redefining the Role of FPGAs in the Next Generation Avionic Systems (Abstract Only)," in *Proceedings of the 2014 ACM/SIGDA International Symposium on Field-programmable Gate Arrays*, ser. FPGA '14, 2014, pp. 248–248.

[9] X. Company, "Xilinx virtex7 product guide." [Online]. Available: http://www.xilinx.com/publications/prod_mktg/Virtex7-Product-Brief.pdf

[10] D. Martinez, "Future challenges in the development of real-time high performance embedded systems," in *Real-Time Systems Symposium, 1998. Proceedings., The 19th IEEE*, Dec 1998, pp. 1–2.

[11] W. Meeus, K. Van Beeck, T. Goedemé, J. Meel, and D. Stroobandt, "An Overview of Today's High-Level Synthesis Tools," *Design Automation for Embedded Systems*, vol. 16, no. 3, pp. 31–51, 2012. [Online]. Available: http://dx.doi.org/10.1007/s10617-012-9096-8

[12] J. Cong, B. Liu, S. Neuendorffer, J. Noguera, K. Vissers, and Z. Zhang, "High-Level Synthesis for FPGAs: From Prototyping to Deployment," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 4, pp. 473–491, April 2011.

[13] R. Nane, V. M. Sima, C. Pilato, J. Choi, B. Fort, A. Canis, Y. T. Chen, H. Hsiao, S. Brown, F. Ferrandi, J. Anderson, and K. Bertels, "A Survey and Evaluation of FPGA High-Level Synthesis Tools," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. PP, no. 99, pp. 1–1, 2016.

[14] K. M. Ali, R. B. Atitallah, N. Fakhfakh, and J.-L. Dekeyser, "Exploring hls optimizations for efficient stereo matching hardware implementation," in *International Symposium on Applied Reconfigurable Computing*. Springer, 2017, pp. 168–176.