

Fast Design Metric Estimation for FPGA-Centric Embedded Systems

J. McAllister, A. Meena, R. Woods

Institute of Electronics, Communication and Information Technology (ECIT),
Queen's University of Belfast,
Belfast, BT3 9DT, UK.

Abstract

Field Programmable Gate Array (FPGA) devices offer the potential for significant improvements in real-time performance of embedded military systems, but at the cost of hugely increased design complexity, with the secondary effect of considerable reduction in design productivity for embedded military systems which harness these devices. In particular, when implementing a system on FPGA, the designer has no facility to estimate the cost of implementation of a specific application without actually creating the implementation, and faces a lengthy and time consuming process to explore the design space created by real-time performance/cost constraints.

This paper presents work which takes the first steps to providing this facility. It shows how OwenFPGA can produce implementations up to 20 times faster, and can provide estimation of implementation resource with 95% accuracy up to 170 times faster than conventional routes, boosting design productivity for FPGA systems. Furthermore, it shows how OwenFPGA effectively 'virtualises' the target hardware platform, enabling a rapid implementation estimation and design space exploration process for hardware which need not actually exist before application may be targeted towards it.

Keywords: FPGA, Embedded Systems, DSP

Introduction

The potential for use of Field Programmable Gate Array (FPGA) as an accelerator complement to the commercial off-the-shelf (COTS) embedded computing platforms for high-end military Digital Signal Processing (DSP) systems comprising multiple RISC and DSP software processors has long been recognised [1-5]. However, FPGAs have more recently evolved into single chip heterogeneous multiprocessing solutions, comprising potentially of numerous software processors as a complement to the standard dedicated hardware accelerators.

Despite their increasing popularity however, there are significant barriers to

easy integration of these devices into conventional embedded computing platforms for military applications. The standard Register Transfer Level (RTL) design approach, using hardware design languages such as VHDL or Verilog, demands architecture design and synthesis skills in short supply as compared with the more plentiful programming skills required for the higher abstraction level design process for multiprocessor software programmable embedded hardware. Furthermore, there are inherent problems with implementation of designs on FPGA, where the RTL programming files require time consuming propagation through low level design tools for implementation creation.

These are two critical factors in the vastly reduced productivity of FPGA system design cycles as compared with those for conventional microprocessor-based platforms. The RTL programming files are difficult to design and produce in the first place, and the lengthy implementation times before the design has any accurate feedback on design performance, means that the popular iterative embedded design styles of today, which rely on fast designer feedback, are severely productivity impaired.

The OwenFPGA design toolset addresses both these situations. It provides an abstract application design interface in commercial modelling environments, with the algorithm specified in this format automatically translated to an embedded FPGA architecture under coarse-grained designer direction. It raises the abstraction level of design and implementation of DSP applications on COTS embedded FPGA and multi-FPGA hardware in such a manner that application implementation productivity may be boosted by up to 20 times.

However, this is not enough to enable rapid system level design for FPGA DSP, due to the long implementation metric feedback process described earlier. OwenFPGA also enables high level estimation of implementation design metrics, in particular required resource. In a FFT application design example, OwenFPGA has shown its ability to provide 95% accurate resource estimates up to 170 times faster than conventional FPGA design routes. This enables fast iterative design space exploration.

This paper describes the OwenFPGA design flow and toolset and shows how it provides productivity increases in both the implementation of FPGA DSP applications, rapid resource estimation and iterative design space exploration. Section 2 outlines

the structure and behaviour of the OwenFPGA toolset and design methodology. Section 3 gauges productivity increases in the OwenFPGA-enabled FPGA design process as compared with conventional routes). Section 4 describes how the early resource estimation scheme operates, and measures the productivity increase of this process as compared with conventional approaches.

OwenFPGA Toolset and Design Methodology

The operating environment of the OwenFPGA toolset is shown in Fig. 1.

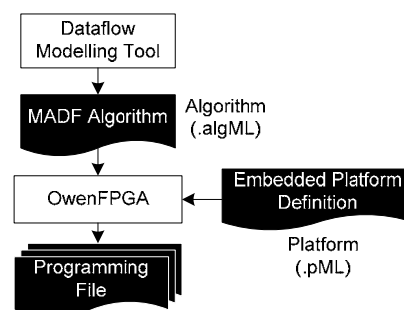


Fig. 1 Owen Operating Environment

The algorithm is described using a dataflow graph (DFG) representation [6] in any appropriate dataflow modelling environment (e.g. Gedae, Ptolemy, DIF or OepnDF) although primarily in this project, Labview. This graph is imported into the OwenFPGA toolset into an open algorithm interface definition XML dialect known as *.algML*. As Fig.1 shows, the COTS FPGA platform is described using a so-called *Embedded Platform Definition* and encapsulated in an XML dialect known as Platform Markup (*.pML*). This definition is part of the platform Board Support Package (BSP), which defines the platform structure and firmware/software for communication with the outside world, as commonly provided by COTS platform vendors. This infrastructure effectively virtualises the hardware, i.e. so long as the platform is not actually required to be programmed, it need

not actually exist to enable the design of the FPGA implementation.

The internal operation of the OwenFPGA toolset to translate the DSP algorithm to an implementation on the COTS FPGA is outlined in Fig. 2. The DFG algorithm representation is firstly partitioned between hardware and software. Three synthesis processes are involved in the *Owen* flow: software synthesis via *Linn*; hardware synthesis via *Mor* and Inter-Processor Communications (IPC) synthesis via *Inver*.

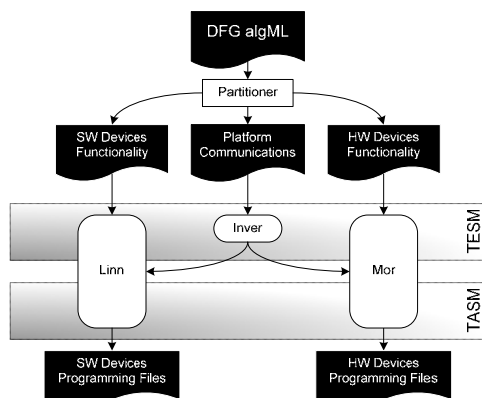


Fig. 2 Owen Toolset Structure

Owen’s VPC design approach enables implementation via different architecture styles. Currently two styles are supported: a point-to-point communication scheme ($P2P_0$) and a bus-based (bus_0) architecture, with each hinting at the communications protocol used for communication between nodes. In the $P2P_0$ template, components implementing the functionality of the algorithm are connected via point-to-point FIFO links directly related to the FIFO channels used in dataflow models, and can be optimised using DFG transformations [10, 11]. The bus_0 template constructs a core network built around a centrally shared bus communications resource.

In all cases, no functional design of the components required to implement the functionality of the algorithm is conducted: all functionality satisfaction is presumed fulfilled by pre-existing functional *cores*.

This framework enables rapid implementation and optimisation of complex DSP system applications on FPGA. To illustrate the productivity increases that this process can enable, Section 3 studies the design time required for implementation of an FFT application on an Alpha DataADM-XRC 5T1 platform hosting a Xilinx Virtex 5 SX95T-1 FPGA.

FFT Demonstrator Analysis

The Labview representation of the FFT algorithm to be implemented and its partitioning across the target platform consisting of the host PC and the ADM-XRC PMC module is shown in Fig. 3. As this shows, the algorithm is partitioned to have only the FFT on the FPGA, with the remainder of the source and sink functionality on the host PC. This partitioning is described using the *.algMAP* XML description shown in Fig. 4.

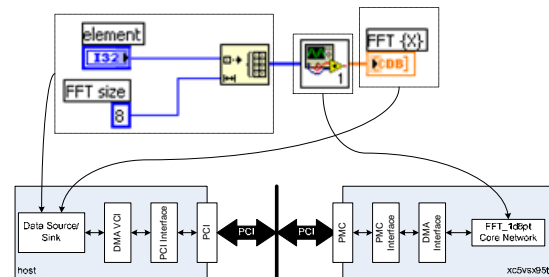


Fig. 3 FFT Demonstrator Implementation

```

<partition>
  <pair>
    <alg_n_6/>
    <device>ADMXRC5T1TestBed-a:ADMXRC5T1:d0</device>
  </pair>
  <pair>
    <alg_n_7/>
    <device>ADMXRC5T1TestBed-a:host</device>
  </pair>
  <pair>
    <alg_n_8/>
    <device>ADMXRC5T1TestBed-a:host</device>
  </pair>
  <pair>
    <alg_n_9/>
    <device>ADMXRC5T1TestBed-a:host</device>
  </pair>
  <pair>
    <alg_n_10/>
    <device>ADMXRC5T1TestBed-a:host</device>
  </pair>
</partition>

```

Fig. 4 FFT Demonstrator *.partML*

This is the only effort the designer must expend to realise the implementation on the

target platform: the structure of the FPGA device, software for the host, and communications between the host and FPGA are automatically realised in OwenFPGA by Mor, Linn and Inver respectively under the direction of the chosen architecture template (P2P₀ in this case) and platform BSP. The functionality of the FFT is realised by the FFT core available in Xilinx Core Generator.

From a manual implementation viewpoint, there are three key tasks to be undertaken for implementation of this demonstrator:

1. COTS FPGA BSP Establishment
 2. FFT Core Integration
 3. COTS FPGA Architecture Generation
- Integration of the IP must be repeated for every new platform and interface signalling standard; 3 must be repeated for every specific architecture configuration analysed during system design space exploration.

OwenFPGA, on the other hand, builds on once-only integration of components. The BSP must only be established once, likewise IP must only be integrated once before it can be reused and reconfigured an infinite number of times. Considering the starting point of completely unintegrated resources for the FFT design (i.e. no BSP established for the ADM-XRC card and unintegrated FFT core), and comparing this with the scenarios where one or more of the board/core are already integrated, the estimated experienced design times for this application, and associated speedups, are outlined in Table 1.

As this shows, depending on the level of integration of BSP and IP cores, the productivity gain of using OwenFPGA can vary from between 2 and 20 times approximately. OwenFPGA builds on previous design experiences; i.e. platforms and cores must be integrated once and only once; the more designs it is used for (and the more platforms and IP that are integrated), the more productive the design process is.

Furthermore, extension to multi-FPGA is trivial for OwenFPGA. The inherently parallel application DFG means that no algorithm rework is required to express parallelism. Furthermore, once the BSP is established for a single COTS platform, extension to multiple platforms is trivial. Consider the extension to the FFT application of Fig. 3 to implement a dual-channel FFT on two separate ADM-XRC-5T1 devices. The only design efforts required are:

1. Development of the new algorithm DFG
2. Definition of the partitioning across the dual-FPGA platform via a new *.almap*.

No manual hardware (or software) development effort is required to produce such an implementation. The ability to naturally extend to multi-FPGA COTS platforms with no hardware development cost is a unique feature of OwenFPGA. In Section 4 we study how new early resource estimation technology can further increase the productivity of the OwenFPGA system design process.

Scenario	Target BSP	IP	OwenFPGA	Productivity Gain
1	Unintegrated	Integrated	6 weeks	2
2	Integrated	Unintegrated	3 weeks	4
3	Integrated	Integrated	3 days	20

Table 1. OwenFPGA Productivity Gains

Early Resource Estimation in Owen

Embedded system design approaches, such as Platform Based Design (PBD) or Function-Architecture Codesign (FAC) [7] base their productivity on the ability to rapidly transform an abstract specification of the behaviour to effect some optimisation of resource, power, throughput or some other physical factor on the implementation, and rapidly gain feedback on the outcome of this process. The speed and reliability of feedback is of paramount importance for such design styles.

In contrast, when a design is automatically generated by an FPGA behavioural synthesis tool (either OwenFPGA or alternatives), the result must propagate through the low-level design tools supplied by FPGA vendors (e.g. Xilinx ISE) before implementation metrics are available, and this is quite a time consuming process. Consider the time taken to process to the first reliable results for the FFT demonstrator of Section 3 using Xilinx ISE, for various point-sizes of FFT, as outlined in Table 2.

Point Size	Processing Time (s)
8	185
16	181
32	182
64	206
128	216
256	214
512	342
1024	511
2048	728
4096	1508
8192	1552
16384	1200
32678	2035

Table 2. Virtex 5 FFT Performance Metrics Processing Time.

As this shows, the processing time increases rapidly with the complexity of

circuit architecture, taking almost 40 minutes to produce estimates for the largest point size design. Furthermore, due to peculiarities regarding the circuit architecture, for the largest to circuits the process actually fails – i.e. this type of design style takes tens of minutes before it can even tell you that it cannot host the design. Clearly, for a high productivity, iterative approach this is not feasible.

OwenFPGA enables a solution to this problem. OwenFPGA targets networks of interconnected cores for functional satisfaction of the application. Given that these architectures are generated fairly rapidly (around 10 seconds from Labview DFG to device architecture translation), exploiting appropriate models of the cores when generating the architecture may provide rapid feedback on the implementation metrics of the application.

Table 3 outlines the results of an experiment which measured the speed of generation and accuracy of resource estimates for the FFT applications used in Table 2 by employing such high level models of the FFT cores in the designs. It describes the time taken to produce core network level estimates, and the accuracy of these estimates as compared with the final absolute values produced post-processing by the Xilinx ISE toolset.

As Table 3 shows, the results are startling. This kind of approach enables 95% resource estimation accuracy after a fraction of the time. With estimated results consistently produced after 12 seconds, this represents a speedup of up to 170 times in the feedback of implementation resource requirements. This virtualisation of the abstraction level at which these metrics are obtained away from the device level and towards the architecture level has enabled a significant breakthrough which can enhance the productivity of iterative FPGA design.

Conclusion

This paper has described the productivity increases in FPGA system design facilitated by OwenFPGA. Depending on the level of previously integrated facilities (COTS platforms and functional cores) the system integration process can be accelerated by a factor of up to 20 approximately. Further, the hardware virtualisation enabled by Owen can produce 95% accurate resource estimates up to 170 times faster than conventional approaches.

References

1. R.L.Walke, R.W.M. Smith and G. Lightbody, "20-GFLOPS QR Processor on a Xilinx Virtex-E FPGA", *Proc. SPIE vol. 116, Advanced Signal Processing Algorithms, Architectures and Implementations X*, pp 300-310, San Diego, 2000
2. B.K. Madahar et. al "How Rapid is Rapid Prototyping? Analysis of the ESPADON Programme Results", *Eurasip JASP*, vol. 2003, pp. 580 -593, January 2003.
3. T. Stefanov et al, "System Design using Kahn Process Networks: The Compaan/Laura Approach", *Proc.*

Design, Automation and Test in Europe Conference (DATE), 2004.

4. Y. Yi, R. Woods, "Hierarchical Synthesis of Complex DSP Functions Using IRIS", *IEEE Trans. CAD*, vol. 25, no. 5, pp. 806-820, 2006.
5. J. McAllister et al, "Multidimensional Core Network Synthesis for FPGA", *Journal of VLSI Signal Processing*, vol. 43, no. 2/3, pp. 207-221, 2006.
6. E.A. Lee and T.M. Parks, "Dataflow Process Networks", *Proc. IEEE*, vol. 83, no. 5, pp. 773-801, May 1995.
7. P. Pop, "*Analysis and Synthesis of Distributed Real-Time Embedded Systems*", Springer 2004.

Acknowledgements

The work reported in this paper was funded by the Electro-magnetic Remote Sensing (EMRS) Defence Technology Centre, established by the UK Ministry of Defence and run by a consortium of SELEX Galileo, Thales UK and Roke Manor Research.

The authors are grateful to the role played by technical lead of the Transducer Embedded Processing (TEP) theme, Bryan Rickett.

Point Size	Feedback Time (s)			Resource Error (%)				Notes
	ISE	Owen	Speedup	LUT	Flip Flop	BRam	DSP48E	
8	185	12	15.4	1.5	4.5	0	0	
16	181	12	15.1	1.5	4.3	0	0	
32	182	12	15.2	-2.1	0.1	0	0	
64	206	12	17.2	1.2	4.0	0	0	
128	216	12	18	0.9	4.1	0	0	
256	214	12	17.8	0.6	3.2	0	0	
512	342	12	28.5	0.4	3.0	0	0	
1024	511	12	42.6	2.5	0.2	0	0	
2048	728	12	60.7	0.3	-0.2	0	0	
4096	1508	12	125.7	-0.1	-0.4	0	0	
8192	1552	12	129.3	0.1	3.0	0	0	
16384	1200	12	100	0.2	2.9	0	0	FAIL
32768	2035	12	169.6	0.0	2.8	0	0	FAIL

Table 3. Virtex 5 FFT Resource Estimation Time.