Rapid Implementation of High End DSP System on FPGA-Centric Embedded Platforms

A. Meena, J. McAllister, R. Woods
Institute of Electronics, Communication and Information Technology (ECIT),
Queen's University of Belfast,
Belfast, BT3 9DT, UK.

Abstract

The evolution of Field Programmable Gate Array (FPGA) devices to the current state-of-the-art System-on-Chip (SoC) devices poses considerable problems in terms of ensuring designer productivity in deploying high end military signal processing applications to these devices. This paper introduces Owen, a system level design environment which aims to resolve this problem. Owen exploits a modular design approach to translate a system level expression of the algorithm (a dataflow graph) into an implementation on a sea of 'virtual processors'. This paper describes the design process used in Owen and an initial illustrative design study.

Keywords: Embedded systems, programmable hardware, DSP implementation

Introduction

The potential for use of Field Programmable Gate Array (FPGA) as an accelerator complement to the traditional platforms for high end military Digital Signal Processing (DSP) systems comprising multiple RISC and DSP software processors has long been recognised [1]. 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.

However, a significant problem exists is FPGA is to achieve it's potential as a single chip multiprocessing platform for the heart of modern DSP systems. No approach offers a holistic system level design environment for FPGA platforms. With the emergence of various software processing architectures (such as Molen [2] or Microblaze [3]) as a complement the to the wide range of accelerator hardware Intellectual Property (IP) available from third party vendors or in libraries such as Xilinx Core Generator, the resources to build complete System-on-Chip (SoC) solutions on a single FPGA device do exist, but the design tools and technology to support the required design process do not. This restricts designer productivity, which in turns acts as a barrier to more widespread adoption of FPGA as a single chip processing solution for DSP systems [4].

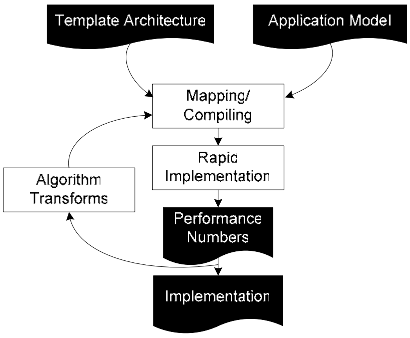
This paper introduces *Owen*, a system level design environment which aims to bridge this gap. Unlike other approaches, Owen takes an innovative approach to modelling both the operation and the architecture of the system it addresses, enabling rapid system implementation and algorithm level optimisation of the implementation. This paper outlines the Owen design approach. In the next section, peer approaches are evaluated and analysed, before the Owen design methodology, tool solution and application are outlined.

Background

Architectural exploration and optimization tools for rapid implementation of DSP applications on multiprocessor and FPGA-centric embedded systems is an increasingly popular research and development area. Tools such as Ptolemy [5], PeaCE [6], Compaan [7], and GEDAE [4] all offer some level of system level implementation. Ptolemy addresses system level modelling for heterogeneous systems, whilst PeACE is a cosimulation environment founded on Ptolemy application models. Compaan is a Matlab to FPGA compiler solution, whilst Gedae has consistently proven pedigree in the field of multiprocessor software synthesis.

However, none of these approaches is an holistic solution to the heterogeneous system design problem. None can achieve all of platform architecture synthesis, software synthesis, inter-processor communication (IPC) synthesis and techniques for optimisation of heterogeneous systems including FPGA. Furthermore, in addition to hard and soft-core microprocessor components (such as Microblaze) and IP cores in third-party libraries, whilst rapid core generation techniques such as Labview FPGA, CatapultC, Accelchip or IRIS [8] allow rapid synthesis of IP cores, there is no strategic approach to including these heterogeneous components in a system

The current best-in-class approaches for FPGA-based system design (PeACE and Compaan) use a hybrid Platform Based Design (PBD)/Function Architecture Codesign (FAC) approach [9] known here as Template-Based Design (TBD). The general operation of TBD is outlined in the diagram below. In all these approaches, an architecture template is exploited to provide a strong enough architectural foundation to enable the system level design features of PBD (i.e. rapid implementation and system level optimisation), whilst still leaving the final architecture open for implementation exploration, a key advantage of the use of programmable logic devices such as FPGA.

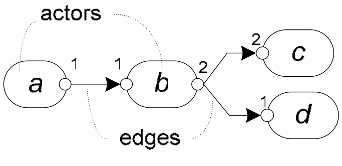


Template Based Design (TBD)

As outlined in the diagram above, a key aspect of the TBD design approach is the algorithm specification phase. In all the approaches outlined previously, the use of dataflow modelling is a key enabler for the system design process. The use of dataflow in the context of Owen is outlined next.

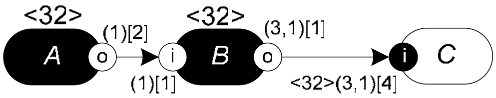
Dataflow in Owen

The use of Dataflow Graphs (DFGs) for rapid implementation of DSP system on multiprocessor platforms is a well established research area [10, 11]. A Dataflow Graph (DFG – see below for an example) $G=\{V, E\}$ describes a set of *actors* V connected by a set of *arcs* E. Arcs carry *streams* of *tokens* to/from actor input/output ports respectively. When a specified number of tokens (known as the *threshold*) are available at each input port of an actor, it *fires*, *consumes* tokens through input ports and *producing* results through output ports.



Example DFG

In Owen, this concept is extended to include arrays of actors and arcs in the Multidimensional Arrayed Dataflow (MADF) language [12]. An example MADF graph is shown below.

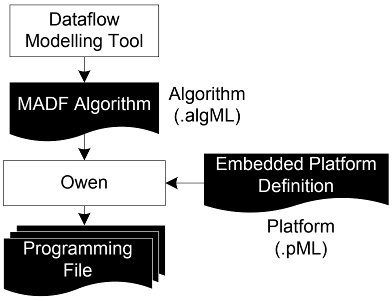


Example MADF Graph

This geometric algorithm description framework allows for more parameterised implementation

Owen Rapid Implementation

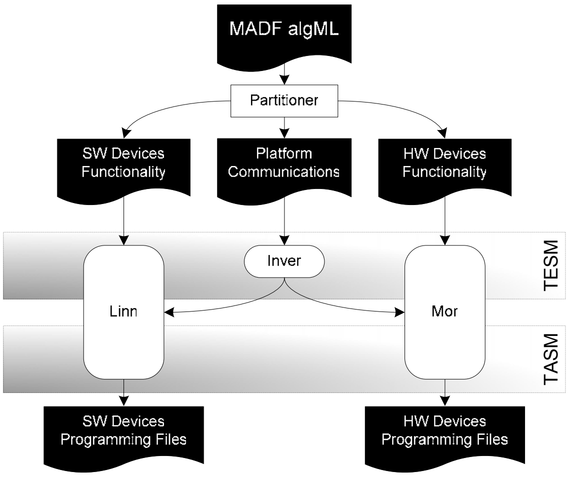
Owen is an implementation flow, and as such harnesses third party simulation technology to act as DFG development environments. The system design context in which Owen operates is shown below. As this shows, a dataflow modelling tool (e.g. Labview, Gedae or Ptolemy) is used to define the MADF algorithm. This tool-specific definition is then exported into a tool independent XML format (known as *algML*), which Owen accepts as its input. This input is then iteratively refined to an implementation for the chosen target platform. This conversion from tool specific formats to a tool independent markup allows the Owen implementation flow to be portable across any commercial modelling tool. However, development is focused on implementing Labview dataflow graphs.



Owen Operation Environment

Similarly, the target platform is defined in a platform description XML (known *pML*), meaning that the Owen design flow is portable across embedded platforms.

The internal operation of the Owen toolset to translate the MADF algorithm specification to an implementation on an embedded platform is outlined below.



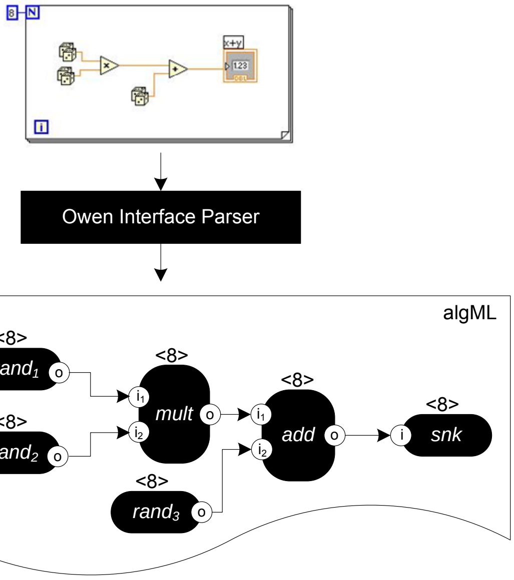
Owen Implementation Flow

The MADF algorithm representation is firstly partitioned between hardware and software. Three processes are involved in the *Owen* flow: software synthesis for software devices (*Linn*); hardware synthesis for the hardware devices (*Mor*); and Inter-Processor Communications (IPC) framework (*Inver*). They are divided between two distinct stages, Technology Specific Mapping (*TESM*) and Target Specific Mapping (*TASM*).

The current toolset manifestation of Owen uses Labview as a graphical modelling environment for the MADF modelling paradigm, exploiting Labview's capability for loop expression to mimic the array semantics of MADF. This principle extends to the use of multiple instance actors in Ptolemy and families in Gedae, and as such the MADF paradigm is widely portability across current best-in-class dataflow modelling tools. A simple example of how this relates to an MADF expression is outlined overleaf.

Software synthesis from DFG algorithm specifications is a well established research area. As such, this project focuses on generation of heterogeneous architectures incorporating hard and softcore

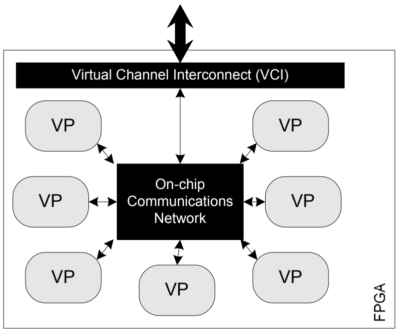
Microprocessors and dedicated hardware components. This project focuses on synthesis of IPC architectures in Inver and heterogeneous system architectures in Mor, and in particular in the first instance, system architecture synthesis for FPGA in Mor. How this is achieved is outlined now.



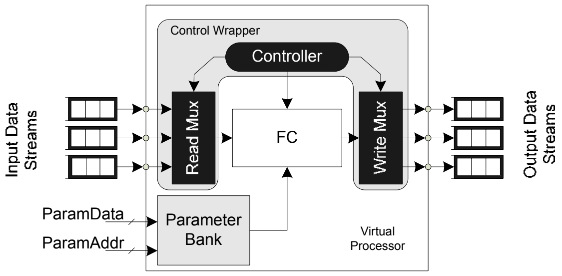
Labview MADF Correspondence

System Architecture Synthesis in Mor

As outlined earlier, Owen is a TBD design approach, exploiting architecture templates to enable rapid implementation. These templates dictate two main themes: how specific computing nodes in the system architecture - known as *Virtual Processors* VPs communicate and are controlled. To enable this abstraction, the target FPGA architecture is a sea of conceptually mesh interconnected VPs, as outlined opposite. As the diagram shows, the FPGA architecture includes a *Virtual Channel Interface* (VCI) to enable the device to connect to the other devices in the platform, and an internal structure composed of the VPs connected to an on-chip communications network. The main purpose of Owen then is to determine the architecture of the VPs and refine the communications network between them. The current template architecture, the Owen $P2P_0$ template, exploits a VP architecture as shown below.



FPGA Architecture



Virtual Processor Architecture

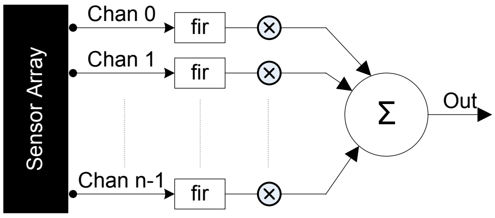
The VP architecture is a locally controlled transputer-like architecture. It consists of a *Functional Core* (FC) part, a *Control and Communications Wrapper* (CCW) and a *Parameter Bank* (PB). The CCW implements any switching necessary to switch data into and out of the FC. This FC is an IP core in the form of either dedicated hardware or a software processing components. Finally, the PB is a memory architecture which stored local constant parameters e.g. FIR filter weights.

This architecture hints at the communications protocol used for communication between nodes. In the $P2P_0$ template, two types of data traffic are supported: streams and parameters. Streams are run-time bandwidth demanding streams of data resulting from computations on the input data samples. These are directly related to the FIFO channels used in dataflow modelling, and can be optimised using DFG transformations [10, 11].

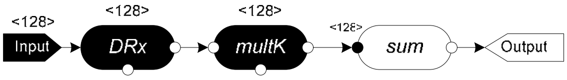
This framework enables rapid implementation and optimisation of complex DSP system applications on FPGA. To illustrate how this may be achieved, the design of a dedicated hardware DSP system, a Fixed Beamformer [12] system is illustrated next.

Mor FBF System Design

Beamforming is an effective method of spatial filtering for radar, sonar, biomedical and communications applications. A beamformer is typically used with an array of sensors, where the FBF determines which direction a particular signal originated in a noisy environment by attenuating the noise and interference coming from other directions by FIR filtering and summing the outputs of an array of sensor elements (below). The the MADF graph of a 128 channel Fixed Beamformer (FBF) system is shown below.



FBF Schematic



FBF MADF Graph

As outlined in the illustrations, a single MADF graph can be used for any size of FBF algorithm, as the entire structure is parameterised in terms of the number of input data channels. In addition, the MADF model exploits cyclic actors, which may be shared amongst multiple channels of data [13]. Hence an *n*-channel FBF system can be implemented using *m* actors, where each actor processes *n*/*m* channels in an interleaved or block processed manner. This type of capability enables the designer to perform system level optimisation of the implementation.

In this design, dedicated hardware IP cores are used to implement the *fir*, *mult* and *sum* actors in the MADFG. As such, these fulfil the role of the FC of the VP architecture outlined earlier. To enable full operational compatability with the VP behaviour the design procedure in [13] must be maintained. In the Owen P2P$_0$ template, each actor on the DFG relates directly to a VP in the implementation. Hence manipulating the number of actors of each type (i.e. the fir and mult actors, as there is only one sum actor) enables control of the number of VPs in the implementation, and hence amount of resource used.

The table below outlines the resource requirements for a number of implementations of the FBF system on Xilinx Virtex-II Pro XC2VP125. Each configuration relates to the use of M *fir*/*multK* cores in an interleaved (*i*) or block processed (*b*) shared manner.

M (i/b)	LUTs			mult18x18 (%)	Slices (%)	Speed (MHz)	Throughput (MSamples/s)
	Logic	DisRam	SRL				
1(i)	3493	8448	16128	99 (22)	15309 (35)	185.5	1.45
2(i)	4813	8448	16128	198 (45)	25444 (58)	203.5	3.18
4(i)	8554	8448	16128	396 (89)	29983 (68)	198.2	6.19
1(b)	3490	24576	0	99 (22)	15105 (34)	185.5	1.45
2(b	4812	24576	0	198 (45)	16033 (36)	224.7	3.51
4(b)	8554	24576	0	396 (89)	19240 (44)	233.4	7.29
8(b)	56386	24576	0	444 (100)	44094 (100)	71.8	4.49

As the table overleaf demonstrates, this shows, a wide range of implementations with various resource constraints and various real-time performance parameters are achieveable. Hence algorithm level optimisation of the implementation performance and resource requirements is achieveable. This work has been extended to explore system level optimisation of the power consumption of the implementation. The results from this explorative effort are outlined in the table below.

	Processing Type	**Power (mW)**
1	Block	597
	Interleaved	508
2	Block	683
	Interleaved	559
3	Block	793
	Interleaved	688

The implementation in Table 2 is a single mode of operation (1 FIR core, 1 multK core) and proves that by switching between block and interleaved shared processing (a DFG level threshold optimisation) can produce a 15-22% implementation power reduction.

Conclusions

The paper has described *Owen*, a high level design flow based on dataflow representation which allows core optimization to be carried out from a dataflow level. In this instance, we have made the core more flexible by making the internal delays more programmable. The paper shows how the concept is applied to the design of a fixed beamformer, where it has proven effective at optimising the implementation in terms of resource, throughput and power consumption.

References

1 Walke, RL, Smith, RWM, Lightbody, G, Advanced Signal Processing Algorithms, Architectures and Implementations X, San Diego, 2000.

2 Vassiliadis, S, Wong, S, Gaydadjiev, GN, Bertels, KLM, Kuzmanov, G, Panainte, EM, IEEE Computer.,**53**, 1363-1375, 2004.

3 Xilinx, Inc, '*Microblaze Processor Reference Guide*', 2007.

4 Madahar, BK, Alston, ID, Aulagnier, AD, Schurer, H, Thomas, M, Saget, B, Eurasip JASP, **2003**, 580-593, 2003.

5 Pino, JL, Ha, S, Lee, EA, Buck, JT, Journal of VLSI Signal Processing, **9**, 7-21, 1995.

6 Oh, H, Ha, S, Journal of VLSI Signal Processing, **21**, 388-366, 2004.

7 Stefanov, T, Zissulescu, C, Turjan, A, Kienhuis, B, Deprettere, E, DATE, France, 2004.

8 Ying, Y, Woods, R, IEEE Trans. CAD, **25**, 806-820, 2006.

9 Pop, P, Eles, P, Peng, Z, IEE Proc. Computers and Digital Techniques, 152, 130-142, 2005.

10 Murthy, PK, Bhattacharyya, SS, '*Memory Management for Synthesis of DSP Software*', CRC Press, 2006.

11 Sriram, S, Bhattacharyya, S, '*Embedded Multiprocessors: Scheduling and Synchronization*', Marcel Dekker, Inc, 2000.

12 Haykin, S, 'Adaptive Filter Theory', Prentice Hall, 2001.

13 McAllister, J, Woods, R, Walke, R, Reilly, D, Journal of VLSI Signal Processing,, **43**, 207-221, 2006.

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, the Engineering and Physical Science Research Council and Selex SAS. The DTC is run by a consortium of Selex SAS, Thales Defence, Roke Manor Research and Filtronic.