QUKU: A Coarse Grained Paradigm for FPGA

Sunil Shukla[1,2], Neil W. Bergmann[1], Jürgen Becker[2]

[1] ITEE, University of Queensland, Brisbane,
QLD 4072, Australia
{sunil, n.bergmann}@itee.uq.edu.au
[2] ITIV, Universität Karlsruhe ,
76131 Karlsruhe, Germany
{shukla, becker}@itiv.uni-karlsruhe.de

Abstract. To fill the gap between increasing demand for reconfigurability and performance efficiency, coarse grain reconfigurable architectures are seen to be an emerging platform. The advantage lies in quick dynamic reconfiguration and power efficiency. Despite having these advantages they have failed to show their mark. This paper describes the QUKU architecture, which uses a coarse-grained dynamically reconfigurable PE array overlaid on an FPGA. The low-speed reconfigurability of the FPGA is used to optimize the CGRA for different applications, whilst the high-speed CGRA reconfiguration is used within an application for operator re-use.

1 Introduction

Generally, ASICs are seen as the only low-power solution which can meet the demand for fast computational efficiency and high I/O bandwidth. However, the rapidly spiraling NRE (Non-Recurrent Engineering) costs of ASICs are a strong motivation for some sort of general purpose computing chip with better power efficiency than microprocessors.

Recently, coarse grained reconfigurable architectures (CGRA) have been developed to bridge the gap between power-hungry microprocessors and single-purpose ASICs. Hartenstein [1] gives a good overview of various CGRAs that have been developed or are being developed in various universities and industry labs. Coarse-grained arrays have the advantages of power-efficiency for their intended application domain and also they are designed with efficient implementation of dynamic reconfiguration supported in hardware and in design software. Furthermore, this dynamic reconfiguration can be very fast, since configuration codes are very short. Despite having all these advantages, CGRAs have not been a commercial success. The main reasons being lack of flexibility, lack of a well defined design flow, lack of wide application domain, commercial availability, to name a few.

Traditionally, FPGAs have been thought of as a prototyping device for small scale digital systems. A decade back, they were generally used to implement glue logic. However current generation mega-gate FPGA chips have changed this perception. There has been an unprecedented growth in the field of FPGA technology. FPGA are

Dagstuhl Seminar Proceedings 06141
Dynamically Reconfigurable Architectures
http://drops.dagstuhl.de/opus/volltexte/2006/742

increasing in density and the cost has seen a continual decrease. FPGA has evolved from being just a fine grained architecture to more of mixed grain architecture. The geometric growth of logic density in FPGAs now makes them a viable alternative to ASIC implementations for many circuits. FPGAs can be reconfigured for an unlimited number of times to implement any circuit. FPGAs have proved their usability in control intensive as well as computational intensive tasks. With the latest Virtex-4 series of FPGAs, Xilinx has established the market in telecommunication, DSP as well as control intensive applications.

FPGAs are widely available as commercial chips, and their large consumer base ensures excellent tracking of the performance and price benefits of semiconductor technology improvements. An attractive feature of FPGAs is the ability to partially and dynamically reconfigure the FPGA function in order to provide application-specific co-processors for a general purpose computing system [2] [3] [4]. However, poor hardware and software support for dynamic reconfiguration, and large configuration file sizes make dynamic reconfiguration difficult and inefficient, so most FPGAs are used with a single configuration during normal operation. To overcome the above mentioned limitations of existing FPGAs and CGRAs, we have come up with this new architecture, QUKU [5][6]. The idea was flexibility and simplicity at design level.

The next section gives a short overview of FPGA based reconfigurable architectures. Section 3 describes the QUKU architecture. Section 4 describes the run time reconfigurability of QUKU. Section 5 describes the design approach followed by project status and conclusion.

2 Related Work

Some other systems consisting of MPU and FPGAs have been proposed in past [7][8][9][10][11][12]. Guccione [13] provides a detailed list of FPGA based reconfigurable architectures. YARDS [9] is a hybrid system of MPU connected with an array of FPGA and has been targeted for telecommunication applications. Spyder [11] consists of a processor with 3 reconfigurable execution units working in parallel. The configuration of the execution units is determined from the set of applications. DISC [10] is an FPGA based processor which loads application specific instruction from a library of image processing elements. It takes advantage of the partial reconfiguration feature of FPGA and implements a system analogous to memory paging in software. PRISM [12] is a reconfigurable architecture built using hardware-software co-design concepts. For each application, new instructions are compiled, synthesized and loaded. PRISM-I and PRISM-II have been prototyped using Xilinx XC3090 and XC4010 respectively. GARP [8] is yet another reconfigurable architecture which combines a MIPS processor with a reconfigurable array to achieve accelerated performance in certain applications. Each computing element performs a logical or arithmetic operation of up to 2 bit width. Larger computation involves aggregation.

3 QUKU

QUKU is a merger of two technologies: CGRAs and FPGAs. The aim is to develop a system which is based on commercially available and affordable technologies, but at the same time provides active support for fast and efficient dynamic reconfiguration. This implementation will enable us to use the same platform for applications which are a mix of control flow and computationally intensive applications. Our system is unique in that it provides two levels of application-specific reconfigurability. Within a single execution of an application, the system works as a CGRA of processing elements. The operation of each PE, and the interconnections between PEs can be reconfigured on a cycle-by-cycle basis, giving maximum reuse of arithmetic and logical operators without long reconfiguration delays. Between different applications, the CGRA can be reconfigured at the FPGA level to provide a different CGRA which is optimized for that application. In other words, the architecture can be reconfigured at two levels, FPGA level and PE/function level. PE level reconfiguration takes place at the cost of few nano-seconds. This reconfiguration caters for any change in functionality or loading of different modules. FPGA level reconfiguration, although takes a few milli-seconds, serves the important function of physically reconfiguring the CG array itself. This feature is particularly useful when there is a need to handle different data widths, handling data in different format like floating point, fixed point, LNS format, etc or re-structuring the whole array to suit the target application in a better way.

The concept of partial reconfiguration on FPGA is to use it as a virtual hardware, implementing applications which are larger than the available hardware by utilizing parallelism. Owing to the problems associated with partial dynamic reconfiguration, as mentioned previously, it can not be used in most of the real life applications. The QUKU approach allows applications which have larger hardware requirements than are available on a single FPGA to efficiently reuse operators through fast CGRA reconfiguration on a cycle-by-cycle basis. However, this architecture does not suffer from the normal CGRA problem of trying to identify the optimal mix of operators for all present and future applications to be used on that array. Rather, the structure of the CGRA can be periodically re-optimized for each new application at a cost of several milliseconds of FPGA reconfiguration time.

It is our conjecture that such a PE array has the potential to provide area and power efficiencies not normally associated with FPGAs.

3.1 Architecture Description

QUKU is a complete SoC solution consisting of a coarse-grained PE matrix overlaid on a FPGA. The aim is to develop a system which is based on commercially available and affordable technologies, but at the same time provides active support for fast and efficient dynamic reconfiguration. Our system is unique in that it provides two levels of application-specific reconfigurability.

Fig. 1 shows a detailed block diagram of QUKU. The coarse grain PE array is coupled with Microblaze based soft processor core using FSL (Fast Simplex Link)

[14]. OPB (On-chip Peripheral Bus) is used to connect to peripherals like external memory controllers, UART, interrupt controller, timer device and other user defined IP cores.

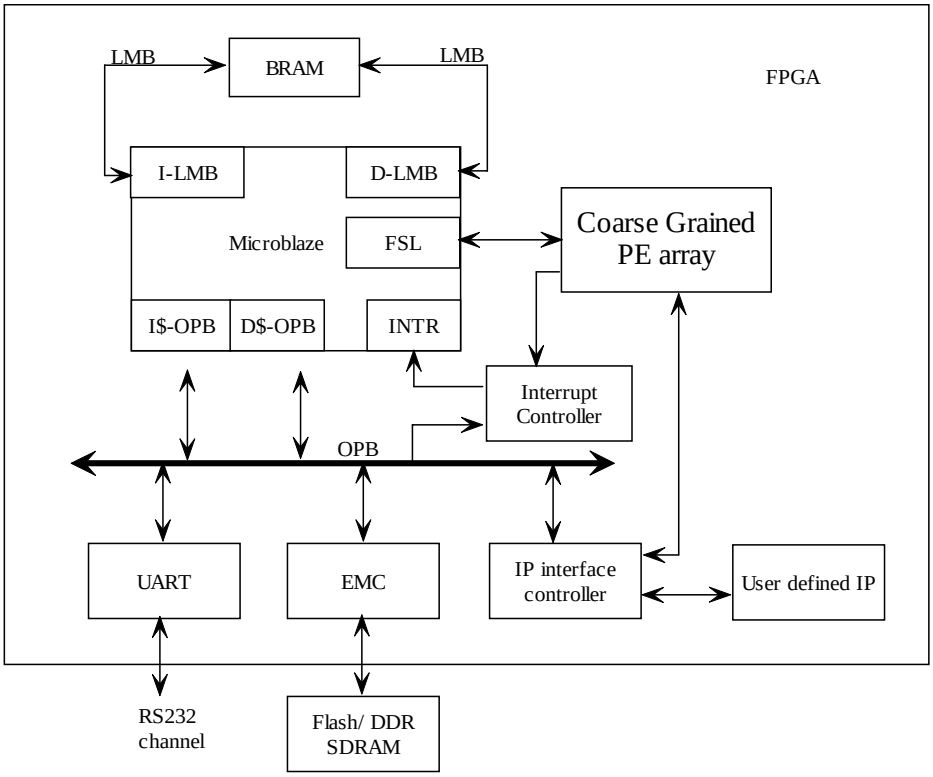

Figure 1 System level diagram of QUKU

3.1.1 PE Structure

The coarse gained programmable matrix consists of a dynamically reconfigurable PE array, CMM and AMM. Refer to fig. 2 for a block level description and acronyms. Each PE consists of a LCC and a LAC besides containing a functional unit and a memory sub-system. CMM is responsible for loading the configuration data onto the LCC of the PEs. AMM loads address parameters onto the LAC of the PEs. LAC controls read and write access to data and result memory of the respective PE. It also generates address for accessing each memory.

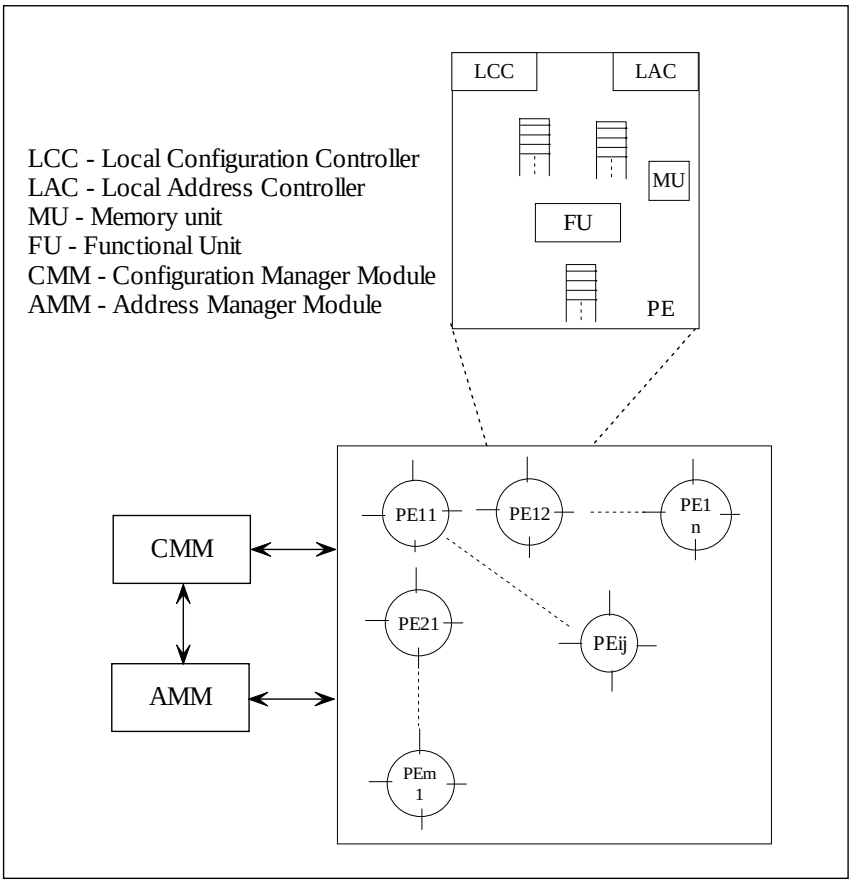

Figure 2 PE array structure

4 Reconfiguration

Conventional CGRAs supports very fast dynamic reconfiguration using micro-codes as opposed to Megabytes of configuration data in FPGAs. But CGRAs have fixed physical layout which ties their usability to a particular application domain. The uniqueness of QUKU lies in the fact that it supports dual layered reconfiguration which overcomes the problem of finding an optimal architecture which suits a wide variety of application domains. The two reconfiguration planes are:

4.1 FPGA level reconfiguration

The FPGA level reconfiguration occurs in partially static manner. This is a very infrequent reconfiguration technique in our design approach. FPGA level reconfiguration is only required when there is a need to physically change the PE array structure. This includes a change in data width, change in numeric representation of data like a switch from fixed point format to floating point format. As obvious, this reconfiguration takes a few milli-seconds.

4.2 PE Level Reconfiguration at Run Time

The very frequently happening reconfiguration is at PE level which configures the PE array to adapt the new functionality. In this section, the configuration memory organization to support run time reconfiguration of whole or partial PE array, is described. As shown in fig. 2, the configuration controller (CMM) is responsible for the loading of different modules on the PE array. The configuration BRAM contains configuration codes for up to four modules.

5 Design Approach

The conventional hardware design approach involves designing, coding, verification and then synthesis and other backend activities. Most of the time is spent in the iterative process of design and verification. A commonly perceived obstacle to the use of FPGAs to provide algorithm speedup is the difficulty of designing custom hardware for each new algorithm. QUKU architecture shifts the focus from complex hardware design problem to writing configuration code for programming the PE array. We have found that the product design time has reduced when using QUKU design flow as compared to conventional HDL based design. The design process can be divided into three phases: design phase, compile phase and execution phase. Each of the phases are described in following sections.

5.1 Design Phase

During design time, the user decides the modules that are to be executed. The first step is to make a schema of processes. For each process, a data flow network diagram is prepared. At this phase, the designer is not bothered about the physical aspects of the PE array. The user perceives the PE array as a homogeneous function rich array with no interconnection bottleneck.

5.2 Compile Phase

After Design phase, QUKU compiles the process network to generate a heterogeneous array of PEs, with each PE optimized for just the range of operations it is required to perform for a given set of applications. In design phase, the aim is to find an optimum PE layout and do hardware software partitioning of tasks.

5.3 Execution Phase

During the execution phase, the central software controller runs the software processes and controls the loading of different modules on the PE array and provides inter-module synchronization. If not active, PEs remain in reset state to save power.

6 Status & Conclusion

We have described the architectural details and advantages of QUKU. At the SoC level, the complete system has been divided into two parts, the coarse grained PE array and the soft processor core. The complete SoC solution is ready and has been downloaded on Xilinx ML401 board with XC4VLX25 device. On the tool flow side, we are yet to develop a tool suite for code profiling, task graph generation and configuration code generation.

We have tested our hardware with a set of experiments like FIR filter and image edge detection algorithms [5] [6]. These set of experiments are mostly to confirm the QUKU design flow, and to establish performance and reconfigurability features. QUKU performance has been found to be intermediate between that of custom logic and FPGA based soft processor core.

6 Acknowledgement

This project is proudly supported by the International Science Linkages programme established under the Australian Government's innovation statement, *Backing Australia's Ability*.

References

1. Reiner Hartenstein, "Coarse Grain Reconfigurable Architectures" Proceedings of the 2001 conference on Asia South Pacific design automation
2. F. Ferrandi, M. D. Santambrogio, and D. Sciuto, "A design methodology for dynamic reconfiguration: The caronte architecture", 19th International Parallel and Distributed Processing Symposium 2005

3. S. Trimberger, D. Carberry, A. Johnson and J. Wong, "A Time-Multiplexed FPGA", Proceedings of FCCM'97 Conference, Napa (1997), pp 22-28
4. J. Becker, M. Hübner, M. Ullmann, “Real-Time Dynamically Run-Time Reconfiguration for Power-/Cost-optimized Virtex FPGA Realizations”, 12th International Conference on Very Large Scale Integration VLSI-SoC 2003 , Darmstadt, 1.-3. December 2003
5. Sunil Shukla, Neil Bergmann, Jürgen Becker, “QUKU: A Fast Run Time Reconfigurable Platform for Image Edge Detection", proceedings of International Workshop on Applied Reconfigurable Computing (ARC 2006)
6. Sunil Shukla, Neil Bergmann, Jürgen Becker, “QUKU: A Two-Level Reconfigurable Architecture”, proceeding of ISVLSI 2006, pp. 109-116
7. Alexandro M. S. Adário, E. L. Roehe and S. Bampi, “Dynamically Reconfigurable Architecture for Image Processor Applications”, DAC 99, New Orleans, Louisiana
8. Hauser, J. R.; Wawrzyneck J., “GARP: A MIPS Processor with a Reconfigurable Coprocessor” Proceedings of the 1997 IEEE symposium on FPGAs for Custom Computing Machines, pp 24-33
9. A. Tsutsui, T. Miyazaki, “YARDS: FPGA/MPU Hybrid Architecture for Telecommunication Data Processing”. Proceedings of FPGA 1997, pp. 93-99
10. M. J. Wirthlin, B. L. Hutchings, “DISC: The dynamic instruction set compiler”, in FPGAs for Fast Board Development and Reconfigurable Computing, Proc. SPIE 2607, pp. 92-103 (1995)
11. C. Iseli, E. Sanchez, "Spyder: A Reconfigurable VLIW processor using FPGAs”, in proceedings of IEEE workshop on FPGAs for Custom Computing Machines, 1993, pp. 17 – 24
12. P. Athanas, H. F. Silverman, “Processor Reconfiguration through Instruction Set Metamorphosis”, IEEE Computer, March 1993, pp. 11-18
13. S. Guccione, “List of FPGA based Computing Machine”, online at http://www.io.com/~guccione/HW_list.html
14. Xilinx Inc. online at "http://www.xilinx.com"