Recent Trends in FPGA Architectures and Applications

Philip H.W. Leong
Dept. of Computer Science and Engineering
The Chinese University of Hong Kong, Hong Kong

Abstract

Since their introduction in the 1985, field programmable gate arrays (FPGAs) have become increasingly important to the electronics industry. They have the potential for higher performance and lower power consumption than microprocessors and compared with application specific integrated circuits (ASICs), offer lower non-recurrent engineering (NRE) costs, reduced development time, easier debugging and reduced risk. Since modern FPGAs can meet many of the performance requirements of ASICs, they are being increasingly used in their place. In this paper, some recent developments in FPGA devices, platforms and applications are reviewed, with a focus on high performance applications of this technology.

1. Introduction

Today's FPGAs are entire programmable systems on a chip (SoC) which are able to cover an extremely wide range of applications. The main features of modern FPGAs are first described and compared with those of the past. In particular, the Altera Stratix III and Xilinx Virtex-5 families of devices, both using a 65 nm copper interconnect process, will be used as examples of contemporary FPGAs.

FPGAs are composed from clusters of logic cells (LCs), interconnected via programmable routing resources [1]. The configuration of the FPGA is stored on embedded static RAM within the chip, this controlling the contents of the LCs and multiplexers that perform routing. This basic architecture has not changed dramatically since their introduction in the 1980s. Early FPGAs used a logic cell consisting of a 4-input lookup table (LUT) and register. Present devices employ larger numbers of inputs (6-input for Virtex-5 and 7-input for Stratix III) and have other associated circuitry. Since area increases with the number of inputs but logic depth decreases, the trend for larger LUTs reflect the increased interconnect to logic delay in modern integrated circuit (IC) technology. Clusters interconnect multiple LCs and serve to provide local routing within the cluster. The larger number of metal layers available in IC processes (12 layers in the case of Virtex-5) is reflected in better programmable interconnect density. Whereas early devices had issues with routability, this is rarely the case in contemporary devices.

Embedded blocks are extensively used in FPGAs, serving to improve delay, power and area if utilised by the application, but waste area and power if unused. Early embedded blocks included fast carry chains, memories, phase locked loops, delay locked loops, boundary scan testing and multipliers. More recently, multipliers have been replaced by digital signal processing (DSP) blocks which add support for logical operations, shifting, addition, multiply-add, complex multiplication etc. Their generality has been improved through features such as multiple wordlength support and cascadability. As a result, they can implement primitives such as filters, transforms and floating point more efficiently. Direct support for most standard modern high-speed external memory standards, networking, communications buses, and IO are also included. Both embedded hard (e.g. PowerPC) and soft (e.g. NIOS II) processors for FPGAs are available; hard processors appear to be on the decline, possibly due to their increased cost when not needed. Processors, hard or soft, support user-configurable peripheral devices implemented on the FPGA and run common operating systems such as Linux.

Another recent trend is the offering of device families with a different mix of features. For example, in Stratix III, the L and E families offer different balances of logic, memory and multiplier ratios, E having higher ratios of memory and multipliers for data-centric applications.

Finally, FPGAs have cost reduction paths for volume production. The Altera HardCopy and Xilinx EasyPath solutions are able to achieve equivalent functionality with significant cost reduction and guaranteed success. In the HardCopy case, a structured application specific integrated circuit (ASIC) can be generated which achieves an average 50% performance improvement and 40% power reduction over the FPGA. In EasyPath, FPGAs which are guaranteed to function for a specified design are provided by the vendor at reduced cost allowing short time to market with no

0-7695-3110-5/08 $25.00 © 2008 IEEE
DOI 10.1109/DELTA.2008.14

requalification requirements.

2. Platforms and Tools

Early research using FPGAs for large-scale computing applications used boards containing large arrays of FPGA devices [2]. A recent development is to use the multi-gigabit transceivers (MGT) available on FPGAs which allow boards to be interconnected via commodity network switches.

An example is the Berkeley Emulation Engine 2 (BEE2) [3] illustrated in Figure 1. It is built around compute module boards which contain 5 Xilinx XC2VP70 FPGAs connected to 4 double data rate 2 (DDR2) dual inline memory modules (DIMMs), providing a maximum capacity of 4GB per FPGA. Each FPGA has two PowerPC 405 processor cores and a local mesh connects four of the five FPGAs in a 2 dimensional compute grid via low-voltage CMOS (LVCMOS) parallel signaling. The remaining FPGA is used for control. Off-module communications are via 18 (2 from the control FPGA and 4 from each of the compute FPGAs) Infiniband $4X$ channel bonded full duplex 2.5 Gbps links providing 180 Gbps off-module bandwidth. Modules can be interconnected in different topologies including tree, 3D mesh or crossbar and standard interfaces are used so standard Infiniband and 10-Gigabit Ethernet switches can be used. Finally, a 100 base-T Ethernet connection to the control FPGA is present for out-of-band communications, monitoring and control.

Commercial machines with FPGA accelerators are also available. These include the Cray XD1, SRC SRC-7 and Silicon Graphics RASC blade. These all involve parallel organisations of high performance microprocessors tightly coupled to FPGA devices. In a manner similar to BEE2, nodes are interconnected via high speed switches and switching topologies can be altered via their configuration. One obstacle to computing applications has been the problem of providing a means to supply an FPGA-based coprocessor with data. This has been addressed by companies such as XtremeData and Nallatech by providing front side bus interfaces which have greatly improved latency and bandwidth over standard peripheral buses such as PCI express. The new strategy involves replacing a CPU board on a multiprocessor motherboard with a pin-compatible FPGA board.

As design complexity continues to increase, methods other than the traditional VHDL/Verilog register transfer level (RTL) approaches have become necessary. Tools such as Mentor Graphic's Catapult allow direct synthesis from C++ algorithms, greatly improving designer productivity. For DSP applications, tools to convert Simulink to synthesisable RTL have become mature, this path being particularly suitable for non-expert designers to build complex FPGA systems. Other new approaches include synthesis from SystemVerilog and SystemC.

3. Applications

FPGAs are suitable for an extremely diverse number of applications including: sorting and searching; signal processing; audio, video and image manipulation; cryptography; packet processing; random number generation and logic emulation. Important new markets are likely to include oil and gas exploration, financial engineering, bioinformatics, high definition video, software defined radio; automotive and mobile basestations. In this section, several examples of advanced applications of FPGA technology are presented. These examples are similar in that they are all large-scale applications requiring processing power surpassing processor and DSP-based solutions; production volume is limited; and their functionality is subject to change. These characteristics combine to make ASIC solutions less attractive.

3.1. Radio Astronomy

FPGAs are important in radio astronomy as they are able to meet the high performance requirements required while maintaining flexibility and relatively low cost. An interesting reported application in the field of radio astronomy is a billion-channel spectrometer used in the Search for Extraterrestrial Intelligence (SETI) project at the University of California at Berkeley and implemented on a BEE2 system [3] described earlier. A 16 Gbps, 800 MHz bandwidth input is passed through a 128 tap, 4 channel polyphase filter bank (PFB) on the control FPGA and split into 4 200 MHz bandwidth streams. Each stream is handled by a compute FPGA which implements a 256 million channel spectrometer with 0.745 Hz resolution. The spectrometer's processing includes an 8K channel PFB, data reordering, 32K point fast Fourier transform (FFT) and power spectrum computation. Each FPGA performs 29.4 GMACs (billion multiply-adds per second).

The FPGA-based system using 130 nm technology was compared with 130 and 90 nm DSP chips (Texas Instruments C6415-7E and C6415T-1G) and a 90nm microprocessor (Pentium 4 570) on the spectrometer and other DSP applications. Performance in terms of computational throughput per chip was a factor of 10 to 34 over the DSP chip in 130nm technology and 4 to 13 times better than the microprocessor. The FPGA was one order of magnitude better than the DSP and two orders of magnitude better than the microprocessor in terms of power efficiency. Compute throughput per unit chip cost was 20 to 307% better than the 90nm DSP and 50 to 500% better than the microprocessor.

3.2. Chip Multiprocessor Emulation

All major processor vendors are moving to chip multiprocessors (CMPs) but research in this area is difficult due to issues in efficiently prototyping such chips. FPGAs offer an excellent platform for rapid prototyping of this type of application as major architectural modifications can be easily made and tested. Moreover, software-based simulations, while having good flexibility are not suitable for testing large-scale parallel programs with large datasets. FPGAs offer two orders of magnitude improvement in performance over software simulation.

One of the concerns associated with CMPs is the difficulty in programming them. The standard technique is to use multiple threads that execute in parallel. Fine-grain locks are commonly used to control access to common data structures, but much care is required to avoid deadlocks, races and other associated bugs. Transactional memory (TM) wraps all code associated with potentially shared data in a transaction. All threads are executed speculatively without locks assuming that no concurrent access to data occurs. If that is the case at the end of the transaction's execution, the writes are committed to shared memory. Otherwise, the writes are rolled back and the transaction re-executed. TM combines the advantages of simplicity and high performance.

ATLAS was the first full system prototype of a CMP with hardware-based transactional memory support [4]. The BEE2 board was used for the implementation and 8 processor cores were prototyped. This was done using the PowerPC 405 processors available on the board along with specialised data caches connected to a common shared memory. Customised blocks required to support TM were implemented using FPGA resources. These included the data cache, shared memory, snooping bus, arbitration and checkpointing logic. Software support is provided via a library with application programming interface (API) to specify transaction boundaries together with a small runtime kernel.

ATLAS was compared with an optimised software simulator, TASSEL using a PowerPC G5 processor operating at 2 GHz. TASSEL uses direct execution of code on the G5 for benchmark initialisation which greatly improves performance. On a set of benchmarks including 3 scientific programs, a hashtable microbenchmark and a program that emulates a travel reservation system with database, improvement in execution time of ATLAS compared with TASSEL was $100\times$.

3.3. Particle Physics

The Compact Muon Solenoid (CMS) experiment is a general purpose detector designed for the Large Hadron Collider (LHC) at CERN. The goal of this experiment is to confirm or deny the existence of the Higgs boson, a particle predicted by the Standard Model of particle physics but which has not yet been observed. As of 2007, the experiment is still under construction.

FPGAs are used extensively in the CMS for pattern recognition, triggering and data analysis. The CMS silicon tracker measures the bending of charged particles by their bending in a strong magnetic field using finely segmented silicon microstrip detectors. An analogue ASIC is used to store data in an analogue pipeline until a trigger is received. The data are then transmitted serially at 40 MHz via analogue fibre optic links to Front End Driver (FED) cards. The CMS Silicon tracker consists of 9 million strips which are read out by 430 FEDs. The FED is a 9U VME64x card comprising: 8 front-end units each having: an optical receiver module serving 12 fibres, 6 dual 10-bit ADCs and 3 Virtex-II XC2V40 delay FPGAs. There are an additional 8 XC2V1500 front-end FPGAs for finding clusters and a XC2V1500 back-end FPGA for building events.

In the CMS, 40M particle collisions occur per second and an FPGA-based trigger selects 100K "interesting" events to be sent to the FED as analogue laser pulses [5]. The FED serves to: digitise the inputs which have a total data rate of 3.4 GB/s; perform pedestal and common mode noise subtraction; search for clusters of hits and format the output for further processing. The cluster data and event trigger information at 400 MB/s form the outputs of the FED.

3.4. Derivative Pricing

Monte Carlo (MC) simulation uses a large number of randomised trials to infer the probability distribution of the outcome. It can be used to price a large range of financial derivatives, with interest rate derivatives being particularly important because the time period being simulated is often very long. FPGAs may be an enabling technology to allow banks to develop models with larger numbers of factors than presently tractable. In this subsection, we briefly review an FPGA-based implementation of the BGM interest rate model [6].

Denote $F(t, t_n, t_{n+1})$ as the forward interest rate observed at time t for a period starting at t_n and ending at t_{n+1}. Suppose the time line is segmented by the reset dates $(T_1, T_2, ..., T_N)$ (called the standard reset dates) of actively trading caps on which the BGM model is calibrated. In the BGM framework, the forward rates $\{F(t, T_n, T_{n+1})\}$ are assumed to evolve according to a log-normal distribution. Writing $F_n(t)$ as the shorthand for $F(t, T_n, T_{n+1})$, the evolution follows the stochastic differential equation (SDE)

with d stochastic factors:

$$\frac{dF_n(t)}{F_n(t)} = \vec{\mu_n}(t)dt + \vec{\sigma}_n(t) \cdot d\vec{W}(t) \qquad n=1 \ldots N. \quad (1)$$

In this equation, dF_n is the change in the forward rate, F_n, in the time interval dt. The drift coefficient, $\vec{\mu_n}$, is given by $\vec{\mu_n}(t) = \vec{\sigma}_n(t) \cdot \sum_{i=m(t)}^{n} \frac{\tau_i F_i(t) \vec{\sigma}_i(t)}{1+\tau_i F_i(t)}$, where $m(t)$ is the index for the next reset date at time t ($t \leq t_{m(t)}$), $\tau_i = T_{i+1} - T_i$, and σ_n is the d-dimensional volatility vector. In the stochastic term (the second term on the right hand side of Equation 1), $d\vec{W}$ is the differential of a d-dimensional uncorrelated Brownian motion $\vec{W}$, and each component can be written as $dW_k(t) = \epsilon_k \sqrt{dt}$, where ϵ_k is a Gaussian random number drawn from a standardised normal distribution, i.e. $\epsilon \sim \phi(0, 1.0)$.

The MC simulation generates a large number of forward paths according to Equation 1 and performs post-processing to implement a number of derivatives, including caps, knock-out caps, swaps, swaptions, Bermudan bond options and flexi-caps. To implement the BGM application, a datapath that maximises performance is used (Figure 2). Instruction-level parallelism is employed by performing many of the operations in the same clock cycle; deep pipelining further increases parallelism while maintaining a high clock rate; a fast division algorithm; and a carefully chosen fixed-point implementation with variable wordlengths for different parts of the datapath used [6]. Together, these optimisations led to an implementation which was $25\times$ faster than an Intel Pentium-based one.

4. Conclusion

This paper reviewed some of the recent advances in FPGA technology. Case studies of applications that would be very difficult to implement using other technologies were given. As higher performance becomes necessary and ASIC costs, risk adversity and time to market pressures continue to increase, FPGAs will continue to grow in importance.

References

[1] V. Betz, J. Rose, and A. Marquardt, Eds., *Architecture and CAD for Deep-Submicron FPGAs*. Kluwer Academic Publishers, 1999.

[2] J. Vuillemin, B. Patrice, R. Didier, M. Shand, T. Herve, and B. Philippe, "Programmable active memories: Reconfigurable systems come of age," *IEEE Transactions on VLSI Systems*, vol. 4, no. 1, pp. 56–59, 1996.

[3] C. Chang, J. Wawrzynek, and R. W. Brodersen, "BEE2: A high-end reconfigurable computing system," *IEEE Des. Test*, vol. 22, no. 2, pp. 114–125, 2005.

[4] S. Wee, J. Casper, N. Njoroge, Y. Tesylar, D. Ge, C. Kozyrakis, and K. Olukotun, "A practical fpga-based framework for novel cmp research," in *FPGA '07: Proceedings of the 2007 ACM/SIGDA 15th international symposium on Field programmable gate arrays*. New York, NY, USA: ACM, 2007, pp. 116–125.

[5] C. Foudas, R. Bainbridge, D. Ballard, I. Church, E. Corrin, J. Coughlan, C. Day, E. Freeman, J. Fulcher, W. Gannon, G. Hall, R. Halsall, G. Iles, J. Jones, J. Leaver, M. Noy, M. Pearson, M. Raymond, I. Reid, G. Rogers, J. Salisbury, S. Taghavi, I. Tomalin, and O. Zorba, "The CMS tracker readout front end driver," *IEEE Transactions on Nuclear Science*, vol. 52, no. 6, pp. 2836–2840, Dec 2005.

[6] G. L. Zhang, P. H. W. Leong, C. H. Ho, K. H. Tsoi, C. C. C. Cheung, D.-U. Lee, R. C. C. Cheung, and W. Luk, "Reconfigurable acceleration for Monte Carlo based financial simulation," in *Proc. International Conference on Field Programmable Technology (ICFPT)*, 2005, pp. 215–222.

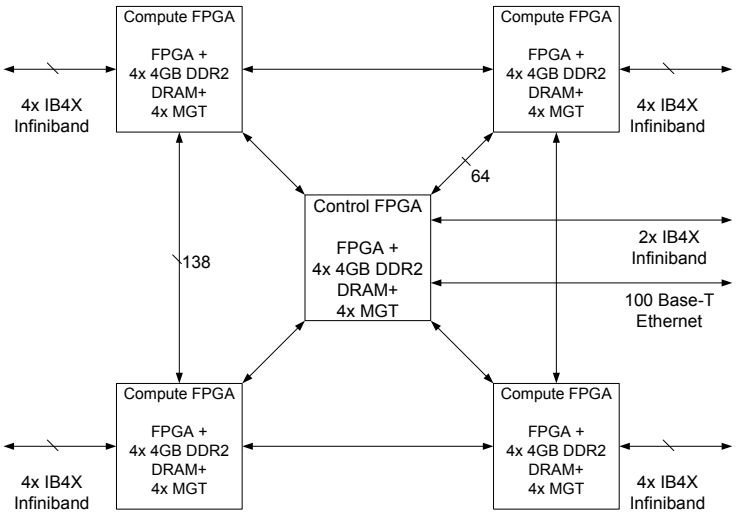

Figure 1. *BEE2 Compute Module block diagram.*

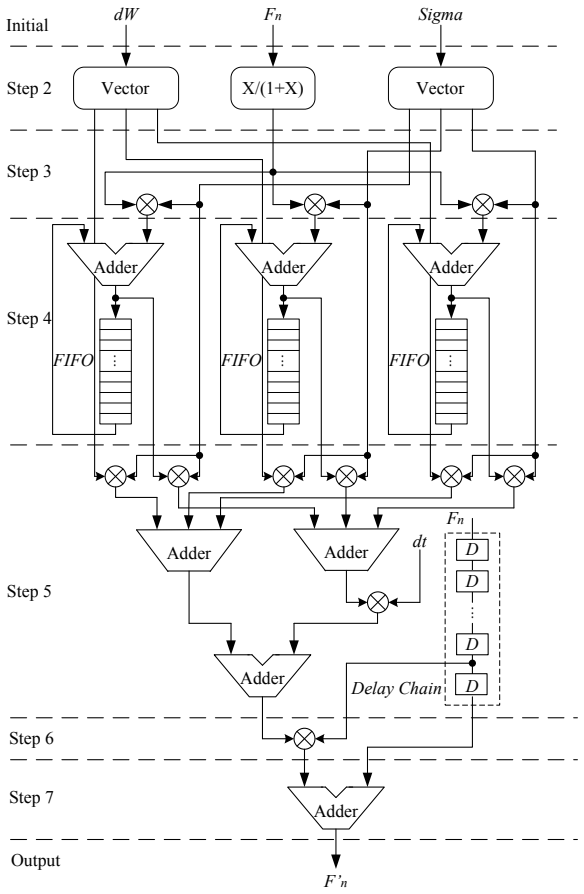

Figure 2. A block diagram of the BGM datapath implementation.