Characterization of Fixed and Reconfigurable Multi-Core Devices for Application Acceleration

JASON WILLIAMS, CHRIS MASSIE, ALAN D. GEORGE, JUSTIN RICHARDSON, KUNAL GOSRANI, and HERMAN LAM
NSF Center for High-Performance Reconfigurable Computing (CHREC), University of Florida

As on-chip transistor counts increase, the computing landscape has shifted to multi- and many-core devices. Computational accelerators have adopted this trend by incorporating both fixed and reconfigurable many-core and multi-core devices. As more, disparate devices enter the market, there is an increasing need for concepts, terminology, and classification techniques to understand the device tradeoffs. Additionally, computational performance, memory performance, and power metrics are needed to objectively compare devices. These metrics will assist application scientists in selecting the appropriate device early in the development cycle. This article presents a hierarchical taxonomy of computing devices, concepts and terminology describing reconfigurability, and computational density and internal memory bandwidth metrics to compare devices.

Categories and Subject Descriptors: B.8.2 [**Performance and Reliability**]: Performance Analysis and Design Aids; C.4 [**Computer Systems Organization**]: Performance of Systems—*design studies*

General Terms: Measurement, Performance

Additional Key Words and Phrases: Computational density per watt, internal memory bandwidth

ACM Reference Format:
Williams, J., Massie, C., George, A. D., Richardson, J., Gosrani, K., and Lam, H. 2010. Characterization of fixed and reconfigurable multi-core devices for application acceleration. ACM Trans. Reconfig. Techn. Syst. 3, 4, Article 19 (November 2010), 29 pages. DOI = 10.1145/1862648.1862649. http://doi.acm.org/10.1145/1862648.1862649.

This work was supported in part by the I/UCRC Program of the National Science Foundation under Grant No. EEC-0642422.
Author's address: NSF Center for High-Performance Reconfigurable Computing (CHREC), Department of Electrical and Computer Engineering, University of Florida, Gainesville, Florida 32611; email: info@chrec.org.

1. INTRODUCTION

Although Moore's Law continues to hold true in that transistor counts on devices are doubling every 18 months, we have reached a point where we can no longer increase clock rates and instruction-level parallelism (ILP) to meet the insatiable demand for computing performance. Thus, large amounts of research are currently focused on how to best utilize all of the transistors on a chip. Over the last few years, multi-core devices have emerged as the leading technology to take advantage of increasing transistor counts. This *architecture reformation* is shifting the focus to exploiting explicit parallelism, rather than relying on ILP and higher clock rates to achieve high performance. The resulting *application reformation* is driving application developers to write explicitly parallel programs, rather than relying on automatic compiler optimizations for high performance. Multi-core devices are finding their way into new accelerator technologies that are used to augment the performance of traditional microprocessor-based systems.

Multi-core devices have at least two major computational components in a single package. Many-core devices have many (e.g., hundreds) of computational components in a single package. The demarcation between multi-core and many-core devices is still somewhat vague. We do not differentiate between multi-core and many-core devices and use the notation MC to refer to them collectively. In this article, we define two primary classes of MC architecture technology: *Fixed MC* (FMC) and *Reconfigurable MC* (RMC). FMC devices have a fixed hardware structure that cannot be changed after fabrication. A prime example of an FMC device is the Intel Xeon X3230 processor. It has four identical fixed processor cores on a single die [Intel Corp. 2008c; 2008e]. By contrast, RMC devices can change their hardware structure after fabrication to adapt to changing problem requirements. Multiple computational cores can be instantiated on the RMC fabric. The primary enabling technology in RMC is the field-programmable gate array (FPGA), but several other exciting technologies are entering the market in this category. Several subcategories are defined in Section 3 along with facets of reconfigurability.

In order to achieve near-optimal implementations given specific design goals and to reduce development time, a system designer must be able to analyze and evaluate appropriate computing devices and accelerator technologies early in the development cycle. However, comparing disparate computing technologies impartially and objectively has been a challenge throughout the history of computing. This is an even greater challenge considering the vast design space of FMC and RMC devices and the number and variety of available architectures. We propose several *computational density (CD)* metrics to facilitate comparing devices within and between architectural categories. These metrics provide the designer with relative performance information in terms of bit, integer, and floating-point operations, incorporating power consumption and memory constraints. Additionally, we have defined an *internal memory bandwidth (IMB)* metric. The IMB metric is used to analyze a device's on-chip memory access capabilities, which is a common bottleneck in many systems. In calculating IMB, we differentiate between block-based systems (BBS) and cache-based systems

(CBS). These contributions are intended to assist designers in rapid device exploration for efficient target device selection. We have not currently included metrics to describe a device's ease of programming or off-chip bandwidth.

The remainder of this article is organized as follows: Section 2 discusses background research on computing taxonomies and performance evaluation methodologies. Section 3 introduces a hierarchical MC computing taxonomy and eight reconfigurability factors. Section 4 discusses the methods used to calculate the CD and IMB metrics for each device type. An overview of the accelerator technologies considered in this study is presented in Section 5. Results and discussion of CD and IMB calculations are presented in Section 6. Finally, conclusions are rendered in Section 7.

2. RELATED WORK

Many researchers have previously surveyed the field of computing devices and computing characterization techniques. The literature includes several classification techniques and we build off of many of them in this paper. Previous works have used numerous criteria to classify both FMC and RMC accelerators. Originally intended to describe fixed architecture devices, Flynn's taxonomy is a common method used to describe a device's parallelism. It classifies accelerators as Single Instruction Single Data (SISD), Single Instruction Multiple Data (SIMD), Multiple Instruction Single Data (MISD), and Multiple Instruction Multiple Data (MIMD) [Flynn 1966]. Host/coprocessor coupling treats the accelerator as a coprocessor to a traditional microprocessor host and classifies the accelerator based on the level of integration. The coprocessor can be directly connected to the host processor, connected via the memory bus, or connected as I/O [Compton and Hauck 2002; Radunovic and Milutinovic 1998].

There are many other important classifiers in the literature targeting RMC accelerators. Device size is the amount of reconfigurable logic used for reconfigurable processing [Guccione and Gonzalez 1995]. The presence of on-chip memories and various memory configurations have also been used to classify devices [Guccione and Gonzalez 1995; Sawitzki and Spallek 1999]. Guccione and Gonzalez [1995] use device size and memory configuration to establish four categories of reconfigurable machines. A small reconfigurable device with no local memory is a Custom Instruction-Set Architecture. Large devices without local memory are Application-Specific Architectures. For devices with local memory, small devices are classified as Reconfigurable Logic Coprocessors, and large devices are Reconfigurable Supercomputers [Guccione and Gonzalez 1995].

Fault tolerance is important for some mission-critical applications for both fixed and reconfigurable architectures [Radunovic and Milutinovic 1998]. For networks of devices, reconfigurability of the device-to-device interconnect is an important classifier [Radunovic and Milutinovic 1998]. Methods of reconfiguration, such as parallel or serial loading of a bitstream, and support for dynamic and partial reconfiguration, can be used to categorize many RMC devices [Bondalapati and Prasanna 2002]. Vertical and horizontal microinstructions

are used to distinguish devices in Sima et al. [2002]. Vertical microinstructions only control one resource; horizontal microinstructions control multiple resources. The execution model refers to the operation of RMC resources when coupled with a host system as described in Compton and Hauck [2002]. RMC resources can operate simultaneously with host operation, or operation on the host can be suspended while the RMC resources are processing.

There are numerous previously researched characterizations that are particularly applicable to our taxonomy. Processing element (PE) granularity and heterogeneity of components [Compton and Hauck 2002; Radunovic and Milutinovic 1998] are common classifiers in the literature. The granularity of a device is based on the native granularity of its basic processing elements. A heterogeneous device has processing elements of different types or structures that are optimized to perform different tasks. Homogeneous devices contain only a single type of computational unit. We primarily focus on reconfigurability and heterogeneity in our taxonomy.

One of the primary challenges of RMC and exotic FMC device evaluation is acquiring computational performance metrics in terms that are comparable to traditional microprocessors. We leverage several related works on device performance characterization. Our CD metric is primarily an adaptation of work done by DeHon [1996]. It relates processing element width, the number of processing elements, and clock frequency to performance, normalized by die area and process technology [DeHon 1996]. Floating-point performance evaluation methods for RC architectures are explored in Strenski [2007] that use vendor tools and datasheet information to determine the maximum number of processing elements for a particular operation that can be supported in parallel. Again, the maximum achievable frequency is used to relate the number of parallel operations (for a given precision) to performance. We extend this methodology to common integer operations. Several performance comparisons are shown in Underwood and Hemmert [2004] that demonstrate the applicability of RC technologies to floating-point operations.

Memory performance plays a significant role in overall system performance. Several works discuss the increasing disparity between the improvement in processor speeds and the improvement in memory speeds. Using general application memory access behavior, Wulf and McKee [1995] show memory performance to be the dominant performance factor as the average memory access time exceeds the time to execute five instructions. They also predict the average number of cycles per access to be 98.8 by 2010. Sohi and Franklin [1991] illustrate through simulation results that low cache bandwidth hampers overall system performance; particularly as instruction issuing or parallel processing capabilities increase.

Burger et al. [1996] use simulation and execution time decomposition methods to show that memory bandwidth is a primary performance bottleneck due to aggressive latency-hiding techniques on several benchmarks. Furthermore, many latency-hiding techniques actually exacerbate memory bandwidth limitations. System performance is increasingly dictated by the operand transfer rate from external memory and the effectiveness of on-chip memory in preserving operands for reuse.

Saulsbury et al. [1996] suggest integrating simpler processors with DRAM memory. Tight integration between memory and simple single-scalar processors is shown to outperform high-end superscalar processors with traditional memory hierarchies and memory bandwidth limitations.

Although much of the literature focuses on the growing gap between processor performance and off-chip memory performance, the message is the same regardless of memory structure or location in the memory hierarchy: as processing performance continues to increase, especially via explicit parallelism, more stress is placed on the memory system to provide data at a fast enough rate to keep processing elements fully utilized. The emphasis on the memory bottleneck points out the need for additional methods and metrics to evaluate memory performance. Our IMB metric is proposed to quantitatively assess on-chip memory performance.

While much of the previous work focused on separately classifying fixed and reconfigurable architectures, an important distinction is that we focus on incorporating both paradigms into a single, MC taxonomy. A new taxonomy is needed due to the on-going architecture and application reformations. The taxonomy proposed is used both as a means to classify devices and to help select the appropriate in-depth characterization methods described in Section 4. Additionally, much of the previous focus has been on the computational performance of devices. Although computational performance is an important device selection criterion, we expand the selection process by incorporating power consumption, an issue of increasingly vital importance in both high-performance computing (HPC) and high-performance embedded computing (HPEC), and memory performance, to address potential memory bottleneck issues.

3. MC TAXONOMY

We propose a hierarchical, tree structure to classify computing devices as shown in Figure 1. The single-core version of this taxonomy is fairly trivial. Thus, the root of the tree is the MC category. The next level of the taxonomy differentiates between FMC and RMC devices. The basic definitions of FMC and RMC are as previously described. Devices can also be a hybrid of FMC and RMC, with segregated fixed and reconfigurable resources on a single die that operate in a mutually exclusive manner. At the lowest level we differentiate between heterogeneous and homogeneous architectures. As previously defined, heterogeneous devices contain multiple types of processing elements. Homogeneous devices contain only a single type of processing element. Finally, within each category there are devices with a variety of base PE granularities.

To further clarify and classify the differences between fixed and reconfigurable architectures, we introduce a set of reconfigurability factors that are summarized in Table I. Devices that exhibit zero or very few of the reconfigurability factors would be classified as fixed devices. Conversely, devices exhibiting many of the reconfigurability factors would be classified as reconfigurable devices. The next section describes the devices in this study and provides a summary of their classification according to this taxonomy.

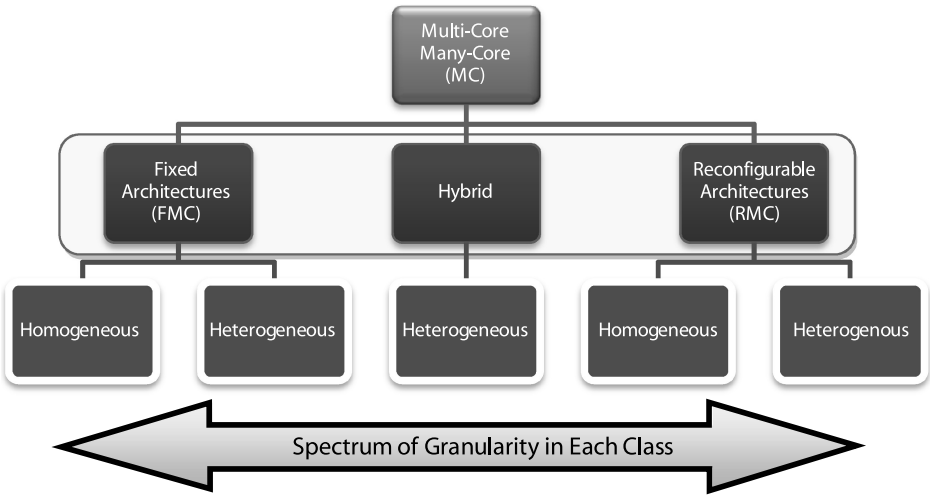

Fig. 1. MC taxonomy.

Table I. Reconfigurability Factors

Factor	Description	Example
Datapath	Device can change width or depth of datapath(s)	Reconfigure from four parallel datapaths with 3 pipeline stages to five parallel datapaths with 4 pipeline stages
Device Memory	Device can change width or depth of on-chip memory blocks	Reconfigure from a 32-bit × 1024 deep memory block to a 64-bit × 512 deep memory block
PE/Block	Device can change operation of PE/Block	Reconfigure PE from a Multiplier operation to a Multiply-&-Accumulate operation
Precision	Device can change numerical precision of PEs	Reconfigure PE from a 64-bit Multiplier to a 24-bit Multiplier
Interface	Device can change memory or I/O interface	Reconfigure memory interface from RLDRAM controller to a DDRII RAM controller
Mode	Device can change assignment of tasks to processing elements	Reconfigure from all PEs performing task A to two PEs performing task A and two PEs performing task B
Power	Device can cycle power of PEs for performance and power tradeoff	Reconfigure PEs from high-power, high-performance operation to low-power, low-performance operation
Interconnect	Device can change communication paths between PEs on chip	Reconfigure communication from bus interconnection topology to mesh interconnection topology between PEs

4. METRIC METHODOLOGY

In this section, we propose several metrics to compare devices within and between taxonomy categories. We evaluate bit-level, integer, and floating-point operations.

4.1 Bit-level Computational Density

Bit-level CD was originally proposed by DeHon [1996]. It describes the computational performance of a device on individual bits, normalizing by die area and process technology. We deviate from the original metric by omitting the normalization and instead group devices by process technology.

Bit-level CD can be defined in terms of device type. Equation (1) applies for FMC devices and coarse-grained RMC devices as

$$CD_{bit} = f \times \sum_{i} (W_i \times N_i), \tag{1}$$

where W_i is the width of element type i, N_i is the number of elements of type i or the number of instructions that can be issued simultaneously, and f is the clock frequency. Vector units are included in the equation above.

We now redefine this metric for FPGAs in terms of LUTs. Each LUT can implement at least one gate-level bit operation. Equation (2) pertains to FPGA technologies as

$$CD_{bit} = f \times \left[N_{LUT} + \sum_{i} (W_i \times N_i) \right], \tag{2}$$

where N_{LUT} is the number of LUTs, W_i is the width of element type i (such as DSP multiplier resources), N_i is the number of elements of type i, and f is the clock frequency.

These two equations give us an estimate of maximum bit-level CD in terms of the clock rate and parallelism (the N terms). It is important to note that these are theoretical peak values. One of the key advantages of FPGAs is that they have less overhead for bit-level computations, so achievable performance will be much closer to peak performance than it would be for coarser-grained devices [DeHon 1996]. We have conservatively chosen to not make a distinction between the number of 4- and 6-input LUTs. Depending on the functions being implemented, a 6-input LUT is not always 1.5 times more computationally powerful than a 4-input LUT.

4.2 Integer Computational Density

FMC and coarse-grained RMC devices typically contain ALUs or coarse-grained processing elements for integer computation. In this case, to determine the *integer CD*, we use Eq. (3) as

$$CD_{int} = f \times \sum_{i} \frac{N_i}{CPI_i}, \tag{3}$$

where N_i is the number of integer execution units or the number of integer instructions that can be issued simultaneously of element type i, CPI_i is the average number of clock cycles per integer instruction for element type i, and f is the operating frequency of the device. The summation over i in this equation takes into account architectures that support vector/SIMD integer instructions.

Integer addition, subtraction, and multiplication often require the same number of clock cycles for fixed architecture devices in terms of throughput. Integer dividers are more complex and require more clock cycles. Consequently, integer and floating-point performance is often reported in terms of addition and multiplication performance and not division. For consistency with previous practices, we will only consider addition and multiplication. Our methodology balances addition and multiplication performance for each integer CD metric by using an equal number of addition and multiplication operations such that the number of balanced operations is maximized.

For the Field-Programmable Object Array (FPOA), which is described in more detail in Section 5.2, integer CD can be calculated for integer widths that match multiples of the width of the basic block. Due to the heterogeneous nature of most RMC devices, the integer CD metric is a summation of the computational capacities of the various elements. The total number of each type of element is extracted from the device datasheet.

For FPGAs, a methodology similar to the one described by Strenski [2007] is used. This characterization is highly dependent on the performance of the IP cores. We assume that integer cores provided by the vendor are highly optimized and will provide a good basis for characterization. The parameters in the following procedure are available as part of the core documentation from the vendor or via experimentation using vendor tools. When using experimentation, typical methods to optimally balance high clock frequency and low resource utilization should be used. These methods are as follows:

(1) Determine the maximum amount of logic resources and the maximum amount of special on-chip resources (e.g., DSP multipliers), for the device.
(2) Assume 15% logic resource overhead for steering logic and memory or I/O interfacing.
(3) Determine the resource utilization and maximum achievable frequency for one instance of the core using DSP resources.
(4) Determine the resource utilization and maximum achievable frequency for one instance of the core utilizing logic-only resources.
(5) Determine the number of simultaneous cores, Ops_{DSP}, that can be instantiated until all DSP resources are exhausted.
(6) Using any remaining logic resources, determine the number of simultaneous logic-only cores, Ops_{logic}, that can be instantiated.
(7) The usable frequency f is the lower of the frequencies determined in Steps (3) and (4).

Thus, the integer CD is defined as:

$$CD_{int} = f \times (Ops_{DSP} + Ops_{logic}) \tag{4}$$

This formula represents peak CD without any consideration of potential performance limitations due to memory bandwidth or on-chip RAM resource restrictions for data buffering. Strenski [2007] describes a method to limit the

number of parallel operations based on the amount of available on-chip memory resources. Memory needs to be allocated to store two operands per operation. The operands can be overwritten with the result in memory. Dual-port memory configurations are used to increase the internal bandwidth. Thus, the *memory-sustainable CD* is limited by the size of the operands and the amount of parallel paths to on-chip memory.

For the FPGA calculations presented in Section 6, we attempted to maximize the number of parallel operations while trying to balance the number of addition and multiplication operations. The term "parallel operations" as used in this article does not equate to GOPs (operations per second) or CD. Parallel operations refers to the number of basic operations (bit operations, adds, multiplies) that can be performed in parallel for a given precision. Clock frequency is still needed in conjunction with the number of parallel operations to determine the computational performance of a device. This balance of addition and multiplication operations can be achieved by iterating through combinations of DSP and logic resources allocated to addition or multiplication operations using the methods previously discussed. We focus on parallel additions and multiplies here because these operations typically have a single-cycle throughput (the case when throughput is not one cycle can be accounted for) and several other performance metrics (e.g., LINPACK benchmarks) have historically focused on these operations. The frequency used for all calculations is the lower of the multiplier and adder frequencies. This method is applicable to both integer and floating-point calculations. Single instantiations of a core provide a reasonable estimate for achievable frequency since to be conservative we are not necessarily assuming all parallel operations are constituents of a pipeline and we use the lowest achievable frequency calculated for each precision. Although the achievable frequencies used here may be optimistic compared to achievable frequencies on real applications, the memory-sustainability limitations on the number of parallel operations caused by the wide and flat structure assumed in this methodology are pessimistic. When taken together, we feel these imprecisions balance each other to provide a reasonable performance estimate. The simplifying assumptions in our methodology enable rapid device comparison independent of specific algorithm requirements.

4.3 Floating-Point Computational Density

In most cases, *floating-point CD* can be determined at the device level using similar methods as shown above for integer CD. Coarse-grained devices use the same model as integer CD, Eq. (3), inserting the number of floating-point units or simultaneous floating-point instructions that can be issued for N_i, and the number of cycles per floating-point instruction for CPI_i. The same constraints regarding division apply; only addition and multiplication operations are considered as part of this metric. As with integer CD, an equal number of addition and multiplication operations are used in finding the maximum number of parallel operations.

Again, floating-point CD for FPGAs is calculated using the same procedure we used for integer operations by repeating the calculation using floating-point

computational cores. The same iterative procedure as integer CD is used to determine the maximum number of parallel operations that employ an equal number of addition and multiplication operations. The number of parallel floating-point operations of these devices is typically much less than the number of parallel integer operations since there is more resource utilization in each floating-point computational core. Consequently, the memory limitations noted previously could have a much greater impact on integer operations than floating-point operations in terms of memory-sustainable CD.

4.4 Power Consumption

Power consumption is also an important device characteristic, for HPEC and HPC alike. Power consumption can be a challenging metric to compute for RMC devices. Reconfigurable devices can have much lower power consumption from peak values since only configured portions of the chip are active. A detailed analysis of static and dynamic power is beyond the scope of this article. Within a metric we hold frequency constant. Therefore, for reconfigurable architectures, we assume that power scales linearly with resource utilization up to the maximum power consumption specified in vendor documentation in the results that follow. For FPGAs, we also scale the maximum power by the ratio of achievable frequency to maximum frequency. Using vendor tools and documentation, we are able to estimate the static power of a configured FPGA, which is used as the starting point in the linear approximation. It is difficult to estimate the error introduced by scaling the maximum power consumption based on frequency and resource utilization. An error estimate would be challenging due to the measurement of FPGA power consumption in isolation from other devices on the board or system (e.g., RAM, interface controllers, etc.).

We feel that based on the linear dependence of dynamic power to frequency and that power increases with resource utilization, our method for estimating power over a range of parallelism is reasonable. The CD per Watt (CD/W) metrics are calculated by taking the CD for each level of parallelism and dividing by the power consumption at that level of parallelism.

4.5 Internal Memory Bandwidth

Internal Memory Bandwidth is a crucial metric due to the impact that the memory subsystem can have on overall system performance. IMB is especially critical for devices with high computational capabilities (e.g., high CD) which require high internal memory performance to keep processing elements busy. Idle PEs are not performing computational work and are wasting energy. The IMB metric seeks to quantify memory performance by describing the rate at which data can be transferred from on-chip memories to processing elements for both block-based systems (BBS) and cache-based systems (CBS).

BBSs typically have a very large number of blocks of memory accessible to the PEs. Block memory acts similar to scratchpad memory; there is no inherent cache structure. Control and design issues such as data segmentation, distribution, replacement, and coherency are left to the user to implement.

BBSs can be found in both FMC and RMC devices. Equation (5) is used to calculate IMB for block-based systems as

$$IMB_{block} = \sum_i \frac{N_i \times P_i \times W_i \times f_i}{8 \times CPA_i}, \tag{5}$$

where N_i is the number of block memories of type i, P_i is the number of ports for memory of type i, W_i is the width of memory of type i, f_i is the operating frequency of memory type i, dividing by eight converts from bits per second to bytes per second, and CPA_i is the number of cycles per access for memory of type i. For devices that do not produce fixed-frequency designs (e.g., FPGAs), f_i is variable up to the achievable frequency of the design, otherwise, f_i is constant.

CBSs generally have multiple levels of cache available to the PEs. There is a hardware structure in place that implements cache design features and control options such as associativity, line size, replacement algorithms, coherency, etc. CBSs can be found in RMC and FMC devices, although they are much more prevalent in FMC devices. RMC devices that do not natively implement a CBS could be configured to do so, but this option is not considered here for simplicity. Implementing a CBS structure on a BBS RMC device would require additional complex control logic that would diminish the advantage that simpler block-based systems have for some application scenarios. The method for calculating IMB for CBS is shown in Eq. (6) as

$$IMB_{cache} = \%hitrate \times \sum_i \frac{N_i \times P_i \times W_i \times f_i}{8 \times CPA_i}, \tag{6}$$

where *%hitrate* is a scale factor to account for variable hit-rates, N_i is the number of block memories of type i, P_i is the number of ports for memory of type i, W_i is the width of memory of type i, f_i is the operating frequency of memory of type i, dividing by 8 converts from bits per second to bytes per second, and CPA_i is the number of cycles per access for memory of type i. IMB is calculated separately for each level of cache, thus the hit-rate scale factor is not included in the summation.

IMB does not include the data rates between computational circuitry and registers. We assume that registers are internal structures present in both BBSs and CBSs that are separate from either block or cache memory. The goal of IMB is to evaluate a device's capability to keep PEs fed; registers are excluded from IMB because they are typically not the bottleneck to PE utilization. Although IMB is not explicitly integrated into the CD and CD/W metrics, CD does include some notion of memory-sustainability as noted in Section 4.2.

5. ACCELERATOR OVERVIEW

In this section, we describe the features of a variety of FMC and RMC devices so that the CD and IMB metrics described in Section 4 can be applied. We have included devices from 130-nm, 90-nm, 65-nm, 45-nm, and 40-nm process

Table II. Device Classification

Device	FMC	RMC	BBS	CBS	Hetero.	Homo.
Arrix FPOA		✓	✓		✓	
ECA-64		✓	✓		✓	
MONARCH	✓	✓	✓		✓	
Stratix-II S180		✓	✓		✓	
Stratix-III SL340		✓	✓		✓	
Stratix-III SE260		✓	✓		✓	
Stratix-IV SE530		✓	✓		✓	
TILE64		✓		✓		✓
Virtex-4 LX200		✓	✓		✓	
Virtex-4 SX55		✓	✓		✓	
Virtex-5 LX330T		✓	✓		✓	
Virtex-5 SX95T		✓	✓		✓	
Athlon 64 X2 6400+	✓			✓		✓
Atom N270	✓			✓		✓
Cell BE	✓		✓		✓	
CSX600	✓		✓			✓
MPC7447	✓			✓		✓
MPC8640D	✓			✓		✓
Opteron 8360 SE	✓			✓		✓
PowerXCell 8i	✓		✓		✓	
Tesla C870	✓		✓			✓
Xeon 7041	✓			✓		✓
Xeon X3230	✓			✓		✓

technologies. Table II provides a summary of classifications for the various devices.

5.1 FMC Devices

Several FMC devices have been included in this study. Tables III and IV list the devices and provide a summary of the key features needed to compute the CD and IMB metrics. These devices exhibit very few of the reconfigurability factors listed in Table I, and are thus classified as FMC devices. This information was gathered from ClearSpeed Technology PLC [2007] for the ClearSpeed CSX600 accelerator, Freescale Semiconductor, Inc. [2006; 2005] for the Freescale MPC7447 PowerPC, AMD, Inc. [2008] and X-bit Laboratories [2006] for the AMD Athlon X2 6400+, Chen et al. [2007] and Wang [2005] for the Cell Broadband Engine (Cell BE), Freescale Semiconductor, Inc. [2006; 2008] for the Freescale MPC8640D PowerPC, Nvidia Corp. [2007; 2006; 2008] for the Nvidia Tesla C870 graphics processing unit (GPU), Intel Corp. [2008b; 2000] for the Intel Xeon 7041, X-bit Laboratories [2007] and Hester [2006] for the AMD Opteron 8360 SE, Chen et al. [2007] and IBM Corp. [2008] for IBM's PowerXCell 8i, [Intel Corp. 2008c; 2008e; 2006] for the Intel Xeon X3230, and Intel Corp. [2008d; 2008a] and Shimpi [2008] for Intel's Atom[1] N270 processor. The Cell BE is a heterogeneous device since it has a traditional processing unit

[1]Detailed information on the L1 cache structures was not available for Atom. We have assumed the Atom's L1 cache interfaces are similar to the Core microarchitecture since both Atom and Core use similar L2 interfaces.

Table III. FMC Device Processing Features

Device	Cores	Instructions Issued/Core	Datapath Width (bits)	Frequency (MHz)	Process Tech. (nm)
CSX600	96	1	64	250	130
MPC7447	1	1+2	32/128	1000	130
Athlon 64 X2 6400+	2	3+1	32/64	3200	90
Cell BE	1+8	2+1	64/128	3200	90
MPC8640D	2	1+2	32/128	1000	90
Tesla C870	128	2	32	1350	90
Xeon 7041	2	3+1	64/128	3000	90
Opteron 8360 SE	4	3+2	64/128	2500	65
PowerXCell 8i	1+8	2+1	64/128	3200	65
Xeon X3230	4	4+1	64/128	2660	65
Atom N270	1	1+1	64/128	1600	45

Table IV. FMC Device Power and Memory Features

Device	Power (W)	On-Chip Memory
CSX600	10	I, D caches, 96 32-bit SRAM banks
MPC7447	10	L1-I, L1-D: 4 words/2 clock cycles, L2: 8 words/9 clock cycles
Athlon 64 X2 6400+	125	Each core: L1-I: 16 bytes/clock cycle, L1-D: 2 × 64-bit interface, L2: 2 × 64-bit interface
Cell BE	70	L1-I, L1-D, L2 (PPE), 8 128-bit LS banks (SPEs)
MPC8640D	14	Each core: L1-I, L1-D: 4 words/2 clock cycles, L2: 8 words/11.5 clock cycles
Tesla C870	120	16 shared memories each w/ 16 32-bit banks
Xeon 7041	165	Each core: L1-I: 3 μops/clock cycle, L1-D: 2 × 128-bit interface, L2: 256-bit interface
Opteron 8360 SE	105	Each core: L1-I: 32 bytes/clock cycle, L1-D: 2 × 128-bit interface L2: 2 × 128-bit interface, L3: variable
PowerXCell 8i	92	L1-I, L1-D, L2 (PPE), 8 128-bit LS banks (SPEs)
Xeon X3230	95	Each core: L1-I: 128-bit interface, L1-D: 2 × 128-bit interface, L2: 256-bit interface
Atom N270	3.3	L1-I: 128-bit interface, L1-D: 2 × 128-bit interface, L2: 256-bit interface

plus up to eight additional compute units. This structure of a processing unit with wider compute units leads to the 1+8 and 64/128 notation in Table III. The other devices are considered homogeneous because all of the sub-units are the same at the level of replication. The devices we have categorized as FMC at most only exhibit the Mode and Power reconfigurability factors. The majority of reconfigurability factors are not represented by this set of devices, leading to the FMC designation.

Additional background on FMC architectures is of interest for calculating CD and IMB. The AMD, Freescale, and Intel Xeon processors each have vector units for each core, again leading to the x+y and 64/128 notation. Note that the Athlon X2 6400+ and Xeon 7041 vector units have a throughput of one instruction every two clock cycles and that for 32-bit integer multiplication there is a 4x throughput reduction for the Tesla C870. The PowerPC processors (MPC7447 and MPC8640D) can issue one ALU and two vector instructions for

Table V. FPGA Device Processing Features

Device	LUTs	DSPs	Max. Frequency (MHz)	Process Tech. (nm)
Stratix-II EP2S180	143,520	768	500	90
Stratix-III EP3SE260	203,520	768	550	65
Stratix-III EP3SL340	270,400	576	550	65
Stratix-IV EP4SE530	424,960	1,024	600	40
Virtex-4 SX55	49,152	512	500	90
Virtex-4 LX200	178,176	96	500	90
Virtex-5 SX95T	58,880	640	550	65
Virtex-5 LX330T	207,360	192	550	65

Table VI. FPGA Device Power and Memory Features

Device	Min. Power (W)	Max. Power (W)	On-Chip Memory
Stratix-II EP2S180	3.26	30	9 128-bit dual-port blocks @ 420 MHz, 768 32-bit dual-port blocks @ 550 MHz, 930 16-bit dual-port blocks @ 500 MHz
Stratix-III EP3SE260	2.11	25	48 72-bit dual-port blocks @ 600 MHz, 864 32-bit dual-port blocks @ 580 MHz
Stratix-III EP3SL340	2.83	32	48 72-bit dual-port blocks @ 600 MHz, 1,040 32-bit dual-port blocks @ 580 MHz
Stratix-IV EP4SE530	3.55	39	64 72-bit dual-port blocks @ 600 MHz, 1,280 32-bit dual-port blocks @ 600 MHz
Virtex-4 SX55	1.00	10	320 32-bit dual-port blocks @ 500 MHz
Virtex-4 LX200	1.27	23	336 32-bit dual-port blocks @ 500 MHz
Virtex-5 SX95T	1.89	10	488 72-bit dual-port blocks @ 550 MHz
Virtex-5 LX330T	3.43	27	648 72-bit dual-port blocks @ 550 MHz

integer operations, two FPU and one vector for single-precision floating-point operations, and three FPU instructions for double-precision floating-point. In IMB calculations, we assume that each μop is 32-bits since it is similar to a RISC instruction [Hennessy and Patterson 2007] for the Xeon 7041. Finally, we focus only on block memory for the devices included here that have both cache and block memories on-chip (e.g., Cell BE, etc.); cache resources are minor bandwidth contributors compared to block memory resources in these devices. For all FMC devices, on-chip memory operates at the core speed and the number of cycles per access is equal to one unless otherwise noted. Fixed-function ASICs, although not included here, would not exhibit Mode reconfigurability. We have excluded fixed-function ASICs because they tend to be proprietary and thus difficult to find data describing them.

5.2 RMC Devices

We have evaluated a variety of FPGA and non-FPGA RMC devices. All of these devices show evidence of numerous reconfigurability factors and are therefore considered RMC devices. Some of the key parameters for FPGA devices regarding the CD and IMB metrics are listed in Tables V and VI. Note that the Altera

Table VII. Maximum CD (in Billions of Operations per Second or GOPs)

Device	Bit-level		16-bit Int.		32-bit Int.		SPFP		DPFP	
	Raw	Sus.	Raw	Sus.	Raw	Sus.	Raw	Sus.	Raw	Sus.
Arrix FPOA	5120	5120	320	320	160	160	n/a	n/a	n/a	n/a
ECA-64	435	435	13	13	6	6	n/a	n/a	n/a	n/a
MONARCH	2048	2048	65	65	65	65	65	65	n/a	n/a
Stratix-II S180	75216	75216	442	442	119	119	53	53	11	11
Stratix-III SL340	154422	154422	932	918	213	213	96	96	26	26
Stratix-III SE260	119539	119539	817	778	203	203	78	78	39	39
Stratix-IV SE530	243866	243866	1099	829	349	346	133	133	68	68
TILE64	4608	4608	240	240	144	144	n/a	n/a	n/a	n/a
Virtex-4 LX200	89952	89952	358	116	72	42	69	46	16	16
Virtex-4 SX55	29184	29184	365	110	73	40	40	40	13	13
Virtex-5 LX330T	119117	119117	623	300	134	122	119	116	26	26
Virtex-5 SX95T	49280	49280	615	226	136	92	74	74	21	21
Athlon X2 6400+	2048	2048	70	70	45	45	26	26	13	13
Atom N270	307	307	14	14	8	8	8	8	5	5
Cell BE	4096	4096	205	205	115	115	205	205	19	19
CSX600	1536	1536	24	24	24	24	24	24	24	24
MPC7447	288	288	17	17	9	9	6	6	3	3
MPC8640D	576	576	34	34	18	18	12	12	6	6
Opteron 8360 SE	4480	4480	190	190	110	110	80	80	40	40
PowerXCell 8i	4096	4096	205	205	115	115	205	205	102	102
Tesla C870	11059	11059	346	346	216	216	346	346	n/a	n/a
Xeon 7041	1536	1536	42	42	30	30	30	30	24	24
Xeon X3230	4095	4095	128	128	85	85	85	85	64	64

Stratix-II FPGA uses 9 × 9-bit DSP multipliers. The Altera Stratix-III, Stratix-IV, and Xilinx Virtex-4 devices use 18 × 18-bit multipliers. The Xilinx Virtex-5 devices use 25 × 18-bit multipliers. As shown in the next section, the maximum frequency listed here is only used for the bit-level CD metric. The maximum achievable core frequency, which is the frequency for a single instance of a DSP or logic-only core after PAR, is used for the integer and floating-point metrics. Further details on determining maximum achievable frequency are provided in Section 4.2.

The values in Tables V and VI were acquired from [Altera Corp. 2007a; 2007b; 2008; BittWare, Inc. 2008; Xilinx, Inc. 2007; 2008]. Power consumption values were estimated using Altera's PowerPlay early power estimator and Xilinx's XPower power estimator spreadsheet tools. Maximum utilization for clock, logic, DSP, and on-chip memory resources, maximum clock frequency,

Table VIII. FPGA Achievable Frequency (in MHz)

Device	Bit-level	16-bit Integer	32-bit Integer	SPFP	DPFP
Stratix-II S180	500	410	420	286	148
Stratix-III SL340	550	400	273	329	195
Stratix-III SE260	550	400	273	354	344
Stratix-IV SE530	550	291	243	241	184
Virtex-4 LX200	500	344	249	274	185
Virtex-4 SX55	500	344	249	353	303
Virtex-5 LX330T	550	463	378	357	237
Virtex-5 SX95T	550	463	378	410	356

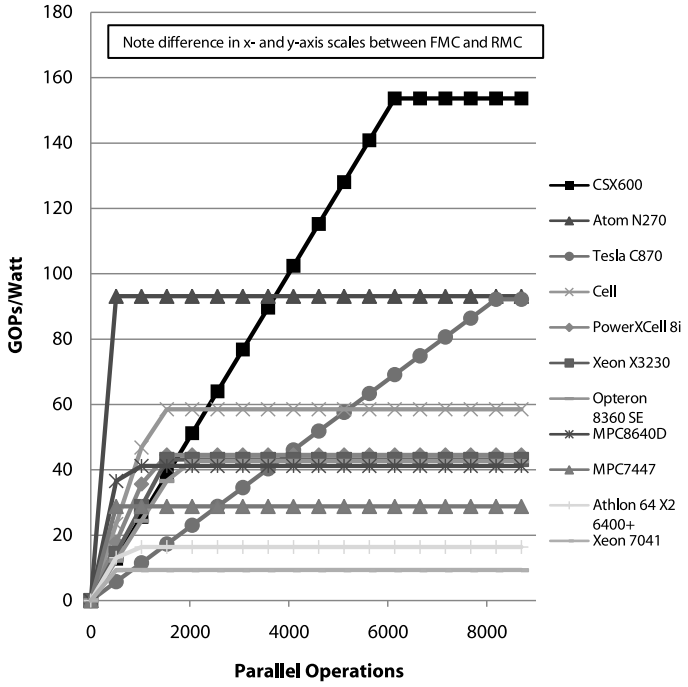

Fig. 2. Bit-level computational density per watt (FMC).

and a 12.5% switching rate were assumed in the spreadsheet tools to estimate maximum power consumption. All of the FPGA devices in this study display all eight of the reconfigurability factors.

We have also considered several new, alternative RMC technologies. MathStar's Arrix Field-Programmable Object Array (FPOA), model MOA2400D-10, has a clock rate of 1 GHz and was built on 90 nm process technology. The FPOA has 256 Arithmetic Logic Unit objects (ALUs) and sixty-four 16 × 16-bit Multiply-Accumulate (MAC) Objects. They also include 40-bit accumulators that can perform an operation every clock cycle. Power consumption is rated at 15.3 W at 25% utilization and 37.6 W for 100% utilization, for a 1-V core voltage [Mathstar, Inc. 2007a; 2007b]. We consider it a heterogeneous device.

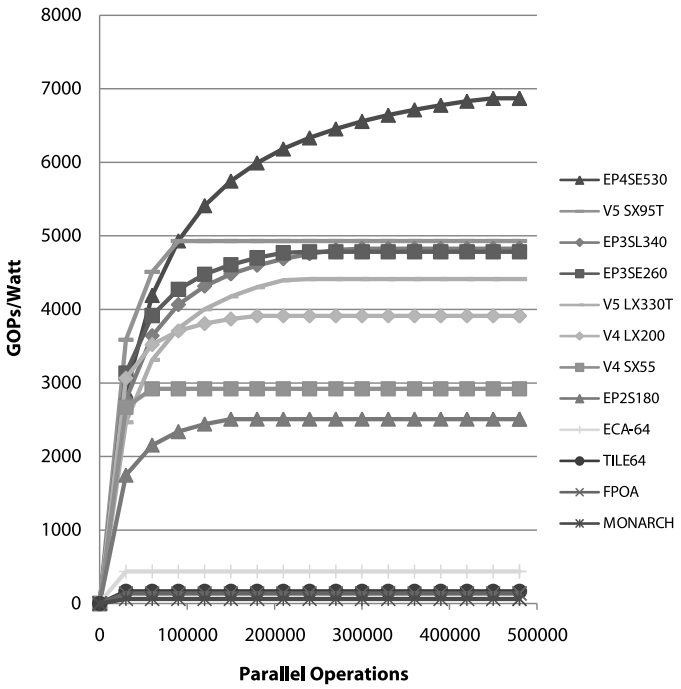

Fig. 3. Bit-level computational density per watt (RMC).

The FPOA is categorized as RMC because it represents the Datapath, Device Memory, Mode, Power, and Interconnect reconfigurability factors.

ElementCXI's ECA-64 is a heterogeneous, data-flow, reconfigurable processor built on 90-nm process technology with a 200-MHz clock. There is a variety of processing element types, supporting many parallel operations. The ECA-64 has published power consumption of up to one Watt at full utilization [ElementCXI, Inc. 2007a; 2007b]. The goal of the ECA-64 is to provide performance and fault resiliency and to replace many custom processors. The ECA-64 includes all eight reconfigurability factors and is classified as RMC.

The TILE64 processor from Tilera is a 64-core processor (at most 63 cores can be used for processing) with a reconfigurable mesh network. Each core is a full 32-bit processor, running at 750 MHz. Each core is a VLIW architecture that can issue three instructions per clock cycle. Instruction packing allows four 16-bit or five 8-bit integer operations to be processed simultaneously. Its idle power consumption is 5 W and maximum power consumption is 28 W. The TILE64 is built on 90-nm technology [Barton 2007; Tilera Corp. 2008]. The goal of the TILE64 is to provide supercomputing performance on a single chip. Custom pipelines can be set up among tiles, enabling Datapath reconfigurability on the TILE64. The collective L2 caches for each tile can also function as a unified L3 cache, demonstrating Device Memory reconfigurability. The TILE64 also incorporates the Mode, Power, and Interconnect reconfigurability factors.

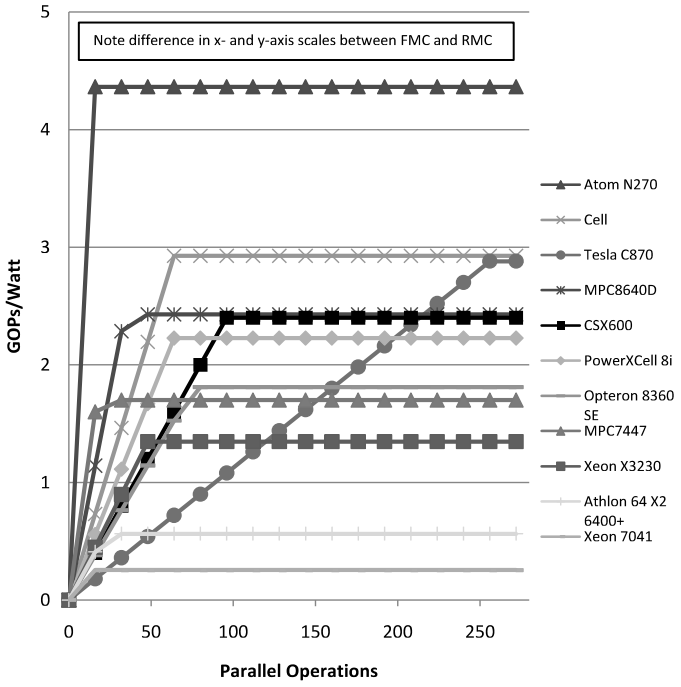

Fig. 4. 16-bit integer computational density per watt (FMC).

Finally, we consider a device that operates using fixed or reconfigurable resources in a somewhat mutually exclusive manner. The MONARCH polymorphous processor spans both FMC and RMC categories. It contains six RISC processors and a Field-Programmable Computing Array (FPCA) of coarse-grained elements. It operates at 333 MHz and has a standby power consumption of 6.7 W and a maximum power consumption of 33 W [Lewins et al. 2007]. The goal of the MONARCH processor is to provide the capability to adapt to changing application requirements. When employing the RISC processors, MONARCH only exhibits the Mode and Power reconfigurability factors. When using the FPCA, MONARCH can take advantage of Datapath, Device Memory, Mode, Power, and Interconnect reconfigurability. The differences in the two distinct modes of operation cause MONARCH to be classified as both an FMC and RMC device.

6. RESULTS AND DISCUSSION

In this section, we primarily focus on detailed results for memory-sustainable CD/W. Table VII summarizes both raw and memory-sustainable CD in GOPs. Memory-sustainable CD is defined as the CD that a device can support with its on-chip memory structure. Memory-sustainable CD is limited when there are not enough parallel paths to memory for the maximum number of parallel operations that can be processed. For each metric and each device, we calculate the maximum memory-sustainable CD. This enables us to determine the

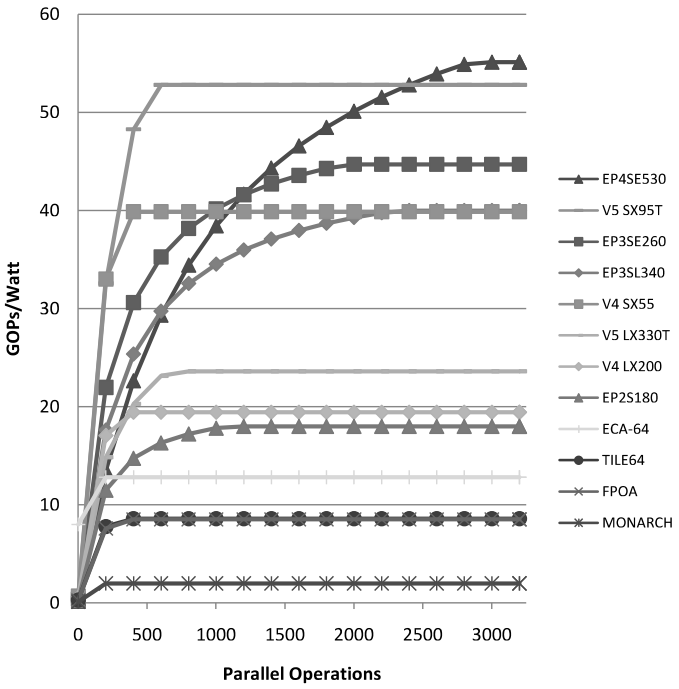

Fig. 5. 16-bit integer computational density per watt (RMC).

maximum amount of exploitable parallelism, which is the number of memory-sustainable parallel operations that can be processed. As indicated previously, clock frequency is held constant within a metric. Maximum clock frequency is used for the bit-level metrics (CD and CD/W) and achievable frequency is used for the remaining metrics as shown in Table VIII. We then examine the impact of varying parallelism to compare performance. We are using the process technology of a device to help group and compare devices both within their generation and across generations. For the integer and floating-point metrics, we adjust the maximum power consumption of FPGA devices by the ratio of achievable frequency to maximum frequency. There may be intuitive expectations for each metric. For the bit-level metrics, one might expect in general that the FPGAs would perform the best due to their fine-grained LUT-based architecture and low power consumption.

For the integer metrics, one might expect the coarse-grained reconfigurable devices, such as the Arrix FPOA, to be the best performers, due to the large number of coarse-grained processing elements that can be active simultaneously. For the single-precision floating-point (SPFP) metric, devices used primarily for graphics processing (Cell BE, Tesla C870) might be expected to perform the best. For double-precision floating-point (DPFP) CD/W, one might expect that the server-class microprocessors, often used as HPC cluster building blocks, would provide the best performance. However, in many cases, the results provided surprises and new insight.

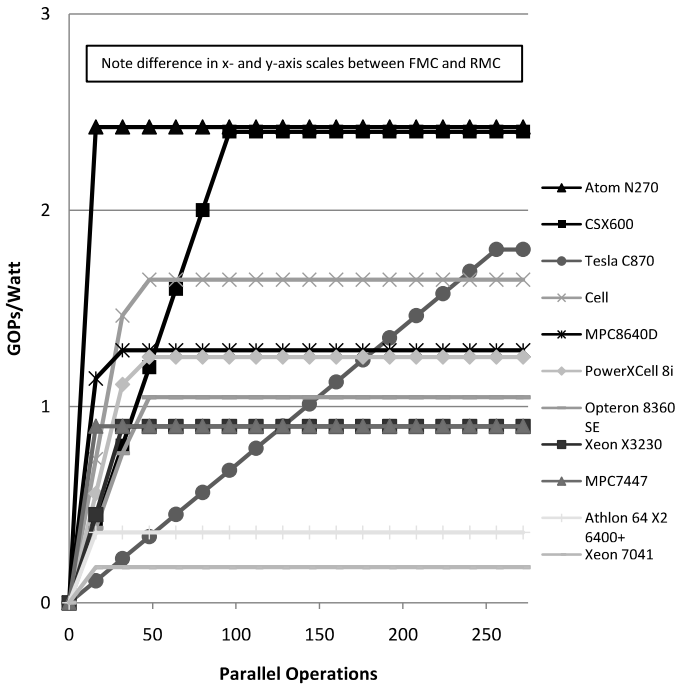

Fig. 6. 32-bit integer computational density per watt (FMC).

6.1 Bit-Level Computational Density per Watt

The bit-level CD/W performances for FMC and RMC devices are shown in Figures 2 and 3. Devices are listed in each chart legend in the order that they appear at the right-most point on the x-axis. The CD/W curves flatten when a device has reached its maximum number of parallel operations. Further levels of parallelism will not show an increase in performance. The points of interest along the x-axis can be limited not only by a device's physical characteristics but also by the algorithm that is to be executed on a device. If a software algorithm can only handle a certain level of parallelism, then that range becomes the point of interest. For this reason, the CD/W curves are extended beyond the physical reaches of the devices to show the intersections and comparisons between different devices.

The 40 nm Stratix-IV EP4SE530 FPGA has significantly more reconfigurable logic resources than all other devices, including the 65- and 90-nm FPGAs, and has the overall best bit-level CD/W performance. The best 65-nm performer is the Virtex-5 SX95T, with the other 65 nm RMC devices a close second. Within the 90 nm devices, the FPGAs also have significantly better performance than the other devices. For 90 nm, the Virtex-4 LX200 has the highest performance with the SX55 a close second. The Stratix-II S180 lags behind the other FPGAs since it has fewer logic resources than the LX200 and higher power consumption than the SX55.

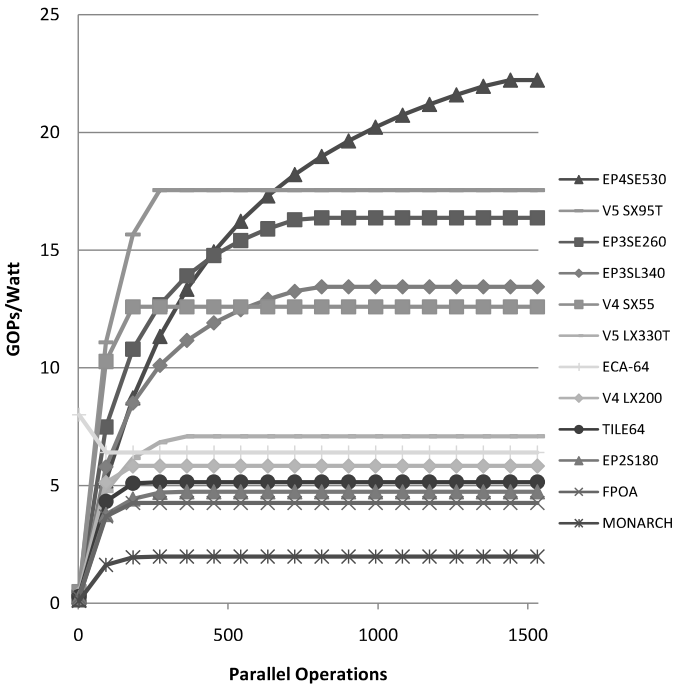

Fig. 7. 32-bit integer computational density per watt (RMC).

For this metric, the maximum frequencies listed in Table VIII were used. The FMC devices for all process technologies perform poorly in this metric due to the high overhead for bit operations on these devices and considerably higher power consumption.

6.2 16-Bit Integer Computational Density per Watt

Figures 4 and 5 show the CD/W metrics for both FMC and RMC devices, respectively. The Stratix-IV SE530 is the overall leader at high levels of parallelism, while the 65 nm Virtex 5 SX95T is the leader at almost all other levels of parallelism. For 90 nm devices the leader for almost all levels of parallelism is the Virtex-4 SX55, while the ECA-64 and Stratix-II EP2S180 also perform well. We scale power linearly up to the maximum resource utilization, which corresponds to the maximum number of raw parallel operations. Devices only have memory-sustainable operations instantiated, thus they will never reach their maximum power dissipation if full utilization of the chip cannot be sustained (e.g., the SX55 and SX95T). The large amount of low-power DSP resources and the power savings due to the limited number of memory-sustainable operations that can be instantiated give the SX55 and SX95T a large CD/W advantage within their respective technology nodes for moderate levels of parallelism. As the number of parallel operations increases, several Altera FPGAs show very

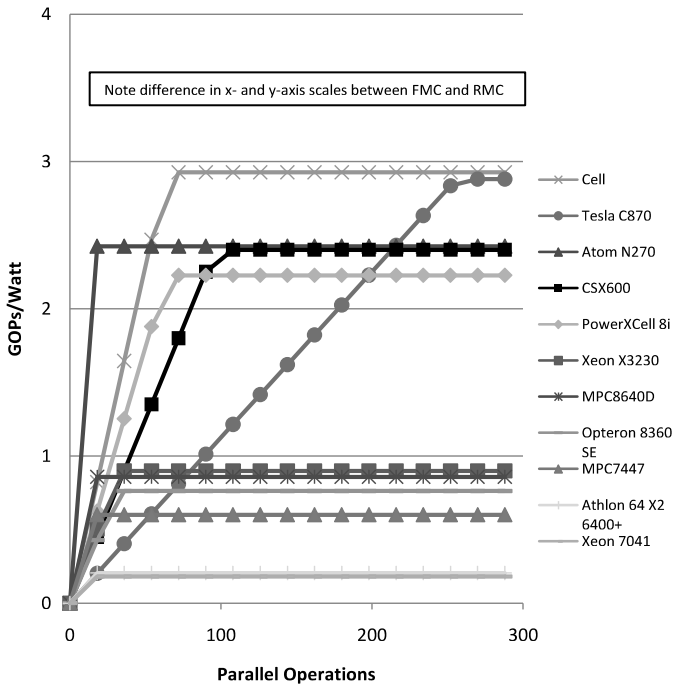

Fig. 8. Single-precision floating-point computational density per watt (FMC).

good performance since they are not as limited by memory-sustainability issues as the Xilinx devices. Again, the FMC devices tend to perform poorly in this metric due to their high, fixed power consumption.

6.3 32-Bit Integer Computational Density per Watt

Overall, the Stratix-IV SE530 leads this metric as shown in Figures 6 and 7 for high levels of exploitable parallelism. The Virtex 5 SX95T performs better at lower levels of parallelism, again due to the power consumption issues cited previously. For very large levels of parallelism, the Altera FPGAs show strong performance, with the Stratix-III SE260 nearing the performance of the SX95T. 32-bit CD/W is again an instance where some of the Xilinx FPGAs suffer in terms of sustainable performance due to memory capacity and hierarchy issues. The Virtex-4 LX200 has a raw maximum 32-bit integer CD of 72 GOPs, but can only sustain 42 GOPs, a 42% reduction. The Virtex-5 LX330T has a raw maximum CD of 134 GOPs but can only sustain 122 GOPs, a 9% reduction. Memory and buffering limitations lead to a reduction in overall CD and CD/W for these devices. The Altera FPGAs have a better balance between the number of parallel operations and the number of memory locations, and thus they do not exhibit this limitation. There is an interesting situation which can be seen in Figure 7. Even though the raw performance of the ECA-64 is relatively low, the power consumption is so low that it initially leads the CD/W metric for 90-nm devices. The negative initial slope for the ECA-64 is

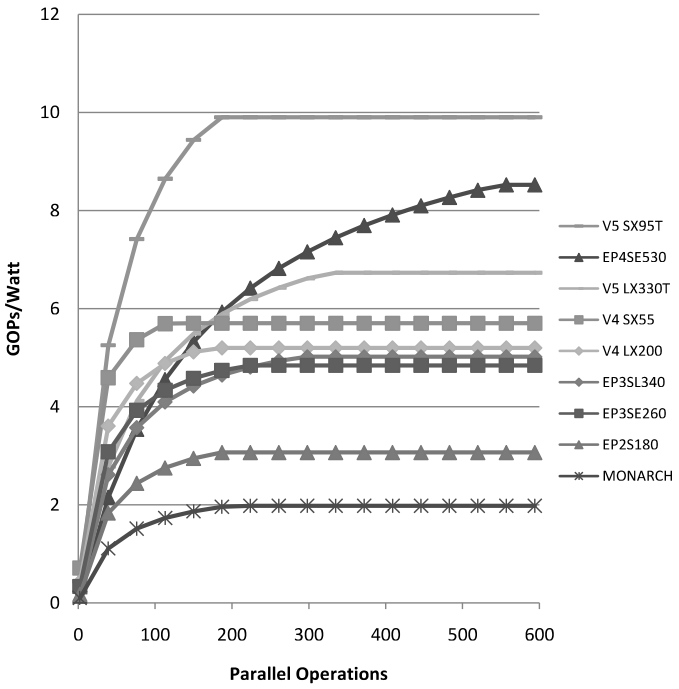

Fig. 9. Single-precision floating-point computational density per watt (RMC).

due to its less than one Watt power consumption at less than 100% resource utilization and that this metric is normalized to one Watt. For low levels of exploitable parallelism, the Stratix-II EP2S180 and TILE64 are good performers. As exploitable parallelism increases, the Virtex-4 SX55 becomes a better performer.

6.4 Single-Precision Floating-Point Computational Density per Watt

Devices that are not intended for SPFP or DPFP operations and would likely perform poorly are not included in the SPFP and DPFP metrics. In addition, due to lack of vendor data, results for TILE64 are also excluded. Despite their significant performance advantage for raw CD performance for SPFP over other devices, the Cell, PowerXCell 8i, and GPU are extremely power-hungry and perform worse on CD/W than most of the RMC devices, as shown in Figures 8 and 9. The 65-nm FPGAs have a major performance increase over the previous generation devices, while maintaining good power efficiency, so that they achieve the best CD/W for all levels of parallelism, led by the Virtex-5 SX95T as shown in Figure 9. Although the Stratix-IV SE530 has twice the CD performance of the SX95T, the higher power consumption of the Stratix-IV SE530 causes it to be a close second in this metric. Even the extremely low-power Atom N270 does not have enough SPFP processing capability to overcome the performance per Watt advantage of most RMC devices, specifically the FPGAs.

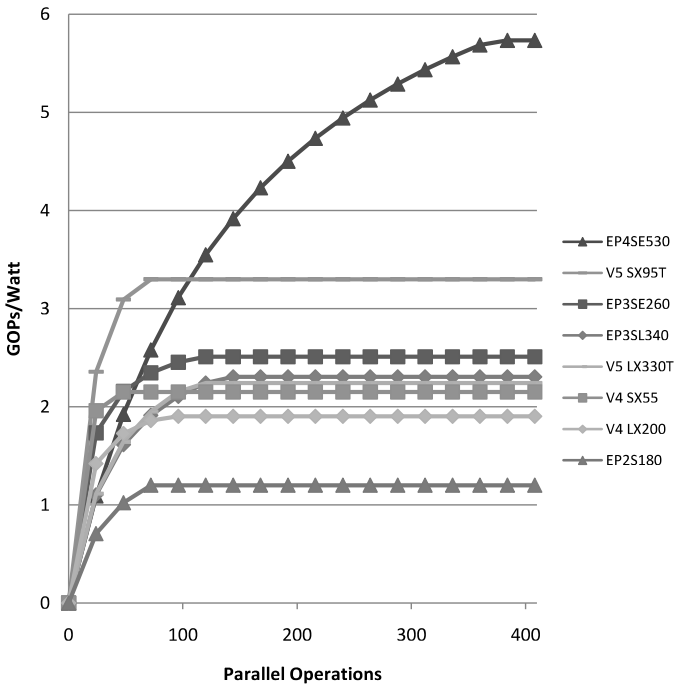

Fig. 10. Double-precision floating-point computational density per watt (RMC).

6.5 Double-Precision Floating-Point Computational Density per Watt

The Stratix-IV SE530 is the noticeable leader for all devices due to its large fabric and the area-intensive nature of DPFP cores as shown in Figure 10. The Virtex-5 SX95T had the highest CD/W score of all 65-nm devices, with the other 65-nm Altera and Xilinx FPGAs clustered together. For 90-nm devices, the Virtex-4 devices were the clear winners for all levels of parallelism for the power-normalized metric. Although it has higher performance, tighter integration of multiple cores on a single die, and improved power efficiency over previous generations, the Xeon X3230 (shown in Figure 11) continues to lag behind the 65-nm RMC devices in CD/W. The 65-nm PowerXCell 8i, a version of the Cell BE with improved DPFP performance, had the highest overall DPFP CD performance of all devices, but was hampered in CD/W by its 92 Watts of maximum power consumption. The CSX600 shows remarkable overall CD/W performance for a 130-nm device due to its relatively high CD score and low power consumption, as shown in Figure 11. Although it was the best raw CD performer of the 90-nm devices, the Xeon 7041 was the worst device in terms of CD/W due to its very high power consumption.

6.6 Internal Memory Bandwidth

Figures 12 and 13 show the IMB metric for cache-based systems and block-based systems, respectively. BBS devices, specifically FPGAs, tend to far

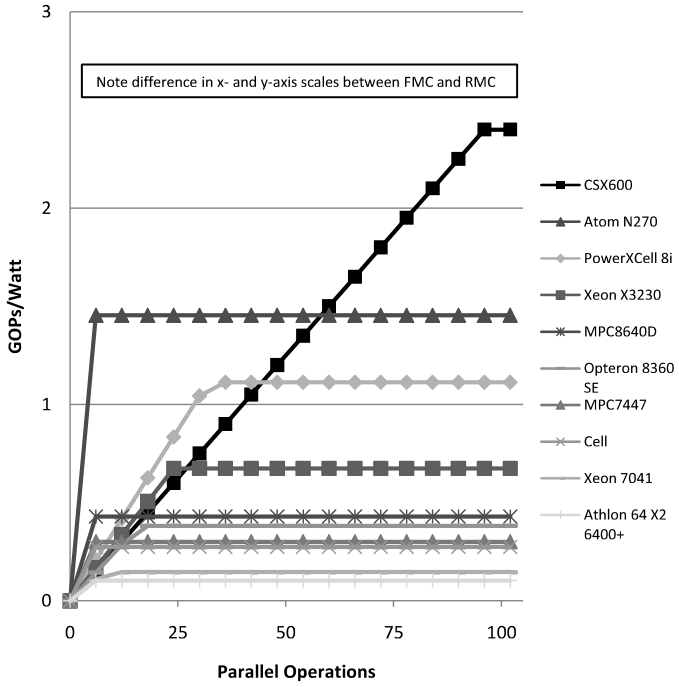

Fig. 11. Double-precision floating-point computational density per watt (FMC).

outperform CBS devices. The MPC7447, MPC8640D, and Atom N270 significantly lag BBS devices due to their limited parallel paths to on-chip memory coupled with their low memory-access frequency, relative to other CBS devices. The TILE64, having 64 L1 and L2 caches with a 750-MHz access frequency, has moderate parallel access to cache, but still cannot compete with any of the FPGAs. The Athlon, Opteron, and Xeon processors have multiple cores each having paths to cache. Their relatively low parallelism to memory is not overcome by their high memory-access frequency, and thus they also significantly lag behind many BBS devices. From Figure 13, we can see that FPGAs dominate this metric, even at relatively low achievable frequencies, due to their high level of parallel access to memory.

7. CONCLUSIONS AND FUTURE WORK

We have presented a taxonomy and a set of reconfigurability factors for classifying fixed and reconfigurable device accelerator technologies. These factors and taxonomy provide useful concepts and terminology to define characteristics of computing technologies. Additionally, we have presented a methodology to comparatively assess these technologies in terms of computational and memory performance and power consumption. Finally, we have shown the large variations in resulting data that can arise when this methodology is applied to disparate accelerator technologies.

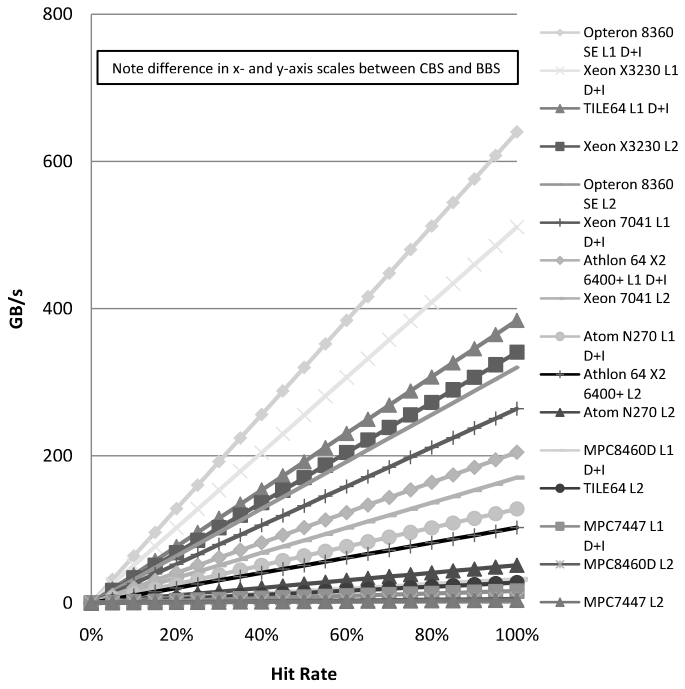

Fig. 12. Internal memory bandwidth (CBS).

As shown in Section 6, various devices show good computational performance depending on the level of exploitable parallelism and the size and type of operation considered. Although FMC devices tended to perform better in terms of raw floating-point CD, the RMC devices performed better when this metric was normalized by power consumption. RMC devices have a distinct advantage over FMC devices since they can adapt to a much wider range of problem requirements with better power efficiency. In this study, RMC devices showed strong CD performance for most precisions and showed a clear CD/W advantage in all cases. Specifically, FPGAs with many low-power DSPs tended to have very high CD/W scores, even for floating-point operations. We recognize there may still be specific situations, especially when high raw double-precision floating-point performance is required, when developers may want to choose an FMC device.

Block-based systems tended to have better memory performance than cache-based systems. FPGAs dominate the IMB metric due to their many parallel paths to on-chip memory and low overhead for memory accesses. The mismatch between raw CD and memory-sustainable CD for some devices demonstrates the importance of balancing computational capabilities and memory access capabilities in architecture design. CD and CD/W for devices such as the Virtex-4 SX55 and Virtex-5 SX95T could be further improved. If more parallel memory structures were implemented in these devices, which would increase their aggregate IMB, more parallel operations could be sustained, leading to

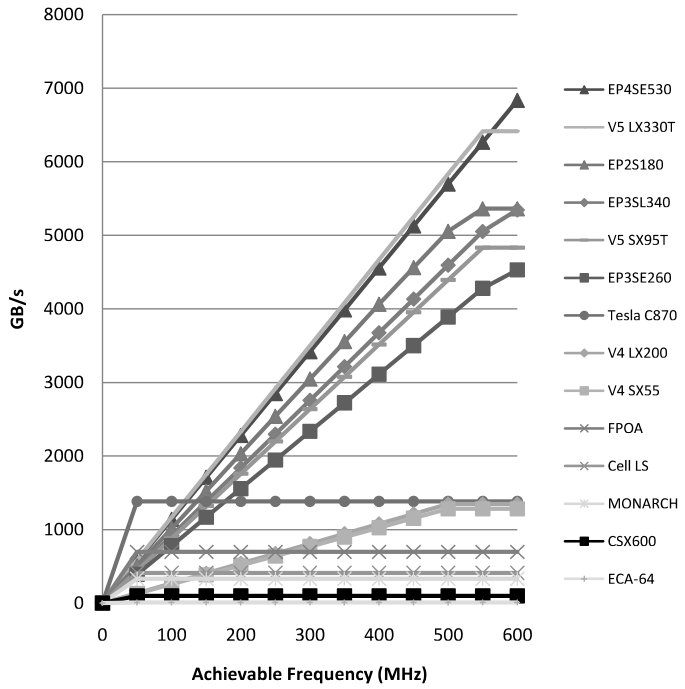

Fig. 13. Internal memory bandwidth (BBS).

higher CD scores and potentially higher CD/W scores. Memory-sustainability is rarely an issue for Altera FPGAs due to their better balance between computation and memory access capabilities, although 16-bit integer CD and CD/W could be slightly improved in the Stratix-IV SE530 with more parallel memory structures.

In general, the newer process technology devices performed better on CD/W than the older process technology devices. This demonstrates a key aspect of the architecture reformation: we have not reached the end of Moore's law, and explicit parallelism will allow us to fully utilize process technology advances and increasing transistor counts while achieving good power efficiency. We recognize the need for progress in the application reformation to enable programs to exploit the high level of parallelism presented here.

There are several other important metrics for overall system performance that are planned for future work. Off-chip memory bandwidth describes the ability of a device to keep its processing elements fed with data from external sources. The I/O capabilities and bandwidth are also important considerations in some systems. Finally, cost is another driving factor in device selection, but is difficult to assess due to the time-varying, volume-dependent nature of retail cost and other complex cost components (e.g., NRE and production costs). We also plan to explore and analyze application metrics and to develop a mapping between the application metrics and the metrics presented here.

ACKNOWLEDGMENTS

The authors gratefully acknowledge vendor equipment and/or tools provided by Altera, Mathstar, and Xlinx, as well as an MS graduate from our group, Sidarth Suresh, for his assistance with this work.

REFERENCES

Altera Corp. 2007a. *Stratix II Device Handbook*. Altera Corp.

Altera Corp. 2007b. *Stratix III Device Handbook*. Altera Corp.

Altera Corp. 2008. *Stratix IV Device Handbook*. Altera Corp.

AMD, Inc. 2008. Key architectural features amd athlon x2 dual-core processors. http://www.amd.com/us-en/Processors/ProductInformation/0,,30_118_9485_130415E13043,00.html.

BARTON, M. 2007. Tilera's cores communicate better. *Microprocess. Rep.*

BittWare, Inc. 2008. *B2-AMC Data Sheet*. BittWare, Inc.

BONDALAPATI, K. AND PRASANNA, V. K. 2002. Reconfigurable computing systems. *Proc. IEEE 90*, 7, 1201–1217.

BURGER, D., GOODMAN, J. R., AND KÄGI, A. 1996. Memory bandwidth limitations of future microprocessors. In *Proceedings of the 23rd Annual International Symposium on Computer Architecture (ISCA'96)*. ACM, New York, 78–89.

CHEN, T., RAGHAVAN, R., DALE, J. N., AND IWATA, E. 2007. Cell broadband engine architecture and its first implementation: A performance view. *IBM J. Res. Devel. 51*, 5, 559–572.

ClearSpeed Technology PLC 2007. *CSX600 Architecture Whitepaper*. ClearSpeed Technology PLC.

COMPTON, K. AND HAUCK, S. 2002. Reconfigurable computing: A survey of systems and software. *ACM Comp. Surv. 34*, 2, 171–210.

DEHON, A. 1996. Reconfigurable architectures for general-purpose computing. Tech. rep., Massachusetts Institute of Technology, Cambridge, MA.

ElementCXI, Inc. 2007a. ECA-64 Device architecture overview. ElementCXI, Inc.

ElementCXI, Inc. 2007b. ECA-64 Product brief. ElementCXI, Inc.

FLYNN, M. J. 1966. Very high-speed computing systems. *Proc. IEEE 54*, 12, 1901–1909.

Freescale Semiconductor, Inc. 2005. *MPC7450 RISC Microprocessor Family Reference Manual* Rev. 5. Freescale Semiconductor, Inc.

Freescale Semiconductor, Inc. 2006. *Altivec Technology Programming Environments Manual* Rev. 3. Freescale Semiconductor, Inc.

Freescale Semiconductor, Inc. 2008. *MPC8641D Integrated Host Processor Family Reference Manual* Rev. 2. Freescale Semiconductor, Inc.

GUCCIONE, S. AND GONZALEZ, M. J. 1995. Classification and performance of reconfigurable architectures. In *Proceedings of the 5th International Workshop on Field-Programmable Logic and Applications (FPL'95)*. Springer-Verlag, Berlin, Germany, 439–448.

HENNESSY, J. L. AND PATTERSON, D. A. 2007. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann, San Francisco, CA.

HESTER, P. 2006. 2006 technology analyst day. http://www.amd.com/us-en/assets/content_type/DownloadableAssets/PhilHesterAMDAnalystDayV2.pdf.

IBM Corp. 2008. PowerXCell 8i processor specifications. IBM Corp.

Intel Corp. 2000. Intel netburst architecture. http://www.intel.com/software/products/documentation/vlin/mergedprojects/analyzer_ec/mergedprojects/reference_olh/reference_hh/inbma.htm.

Intel Corp. 2006. *Inside Intel Core Microarchitecture*. Intel Corp.

Intel Corp. 2008a. *Intel Architecture Software Developer's Manual*, Vol. 1: Basic Architecture. Intel Corp.

Intel Corp. 2008b. Intel xeon processor 7041. http://processorfinder.intel.com/Details.aspx?sSpec=SL8UD.

Intel Corp. 2008c. Intel xeon processor x3230. http://processorfinder.intel.com/Details.aspx?sSpec=SLACS.

Intel Corp. 2008d. Mobile Intel Atom Processor N270 single core datasheet. Intel Corp.

Intel Corp. 2008e. Product brief Intel Xeon processor 3000 sequence. Intel Corp.

LEWINS, L., PRAGER, K., GROVES, G., AND VAHEY, M. 2007. World's first polymorphic computer - monarch. In *Proceedings of the 11th Annual High-Performance Embedded Computing Workshop*.

Mathstar, Inc. 2007a. *Arrix Family FPOA Architecture Guide*. Mathstar, Inc.

Mathstar, Inc. 2007b. *Arrix Family Product Data Sheet and Design Guide*. Mathstar, Inc.

Nvidia Corp. 2006. Nvidia GeForce 8800 GPU architecture overview. Nvidia Corp.

Nvidia Corp. 2007. *Nvidia CUDA Compute Unified Device Architecture Programming Guide*. Nvidia Corp.

Nvidia Corp. 2008. Nvidia tesla c870 specifications. http://www.nvidia.com/object/tesla_c870.html.

RADUNOVIC, B. AND MILUTINOVIC, V. M. 1998. A survey of reconfigurable computing architectures. In *Proceedings of the 8th International Workshop on Field-Programmable Logic and Applications (FPL'98)*. Springer-Verlag, Berlin, Germany, 376–385.

SAULSBURY, A., PONG, F., AND NOWATZYK, A. 1996. Missing the memory wall: The case for processor/memory integration. In *Proceedings of the 23rd Annual International Symposium on Computer Architecture (ISCA'96)*. ACM, New York, 90–101.

SAWITZKI, S. AND SPALLEK, R. G. 1999. A concept for an evaluation framework for reconfigurable systems. In *Proceedings of the 9th International Workshop on Field-Programmable Logic and Applications (FPL'99)*. Springer-Verlag, Berlin, Germany, 475–480.

SHIMPI, A. L. 2008. Intel's silverthorne unveiled: Detailing baby centrino. http://www.anandtech.com/showdoc.aspx?i=3230&p=3.

SIMA, M., VASSILIADIS, S., COTOFANA, S., VAN EIJNDHOVEN, J. T. J., AND VISSERS, K. A. 2002. Field-programmable custom computing machines - A taxonomy. In *Proceedings of the 12th International Conference on Field-Programmable Logic and Applications (FPL'02)*. Springer-Verlag, Berlin, Germany, 79–88.

SOHI, G. S. AND FRANKLIN, M. 1991. High-bandwidth data memory systems for superscalar processors. *SIGOPS Operat. Syst. Rev. 25* (Special Issue), 53–62.

STRENSKI, D. 2007. Fpga floating point performance – a pencil and paper evaluation. *HPC Wire*.

Tilera Corp. 2008. TILE64 processor product brief. Tilera Corp.

UNDERWOOD, K. D. AND HEMMERT, K. S. 2004. Closing the gap: CPU and FPGA trends in sustainable floating-point blas performance. In *Proceedings of the 12th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'04)*. IEEE Computer Society, Los Alamistos, CA, 219–228.

WANG, D. T. 2005. ISSCC 2005: The cell microprocessor. http://www.realworldtech.com/page.cfm?ArticleID=RWT021005084318&p=2.

WULF, W. A. AND MCKEE, S. A. 1995. Hitting the memory wall: Implications of the obvious. *SIGARCH Comput. Architect. News 23*, 1, 20–24.

X-bit Laboratories 2006. AMD's next generation microarchitecture preview: From k8 to k8l. http://www.xbitlabs.com/articles/cpu/display/amd-k8l.html.

X-bit Laboratories 2007. AMD K10 micro-architecture. http://www.xbitlabs.com/articles/cpu/display/amd-k10.html.

Xilinx, Inc. 2007. Virtex-4 family overview. Xilinx, Inc.

Xilinx, Inc. 2008. Virtex-5 family overview. Xilinx, Inc.

Received October 2008; revised February 2009, May 2009; accepted June 2009