

Computational Density of Fixed and Reconfigurable Multi-Core Devices for Application Acceleration

Jason Williams, Alan D. George, Justin Richardson, Kunal Gosrani, Siddarth Suresh
NSF Center for High-Performance Reconfigurable Computing (CHREC)
Department of Electrical and Computer Engineering
University of Florida
Gainesville, Florida 32611
{jwilliams, george, richardson, gosrani, suresh}@chrec.org

Abstract

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, performance and power metrics are needed to objectively compare accelerators. These metrics will assist application scientists in selecting the appropriate device early in the development cycle. This paper presents a hierarchical taxonomy of computing devices, concepts and terminology describing reconfigurability, and computational density metrics to compare devices.

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 paper, 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 [1,2]. 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 sub-categories 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, and incorporate power consumption and memory constraints. These

contributions are intended to assist designers in rapid device exploration for efficient target device selection.

The remainder of this paper 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. An overview of the accelerator technologies considered in this study is presented in Section 4. Section 5 discusses the methods used to calculate the CD metrics for each device type. Results and discussion of CD 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) [3]. *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 [4,5].

There are many other important classifiers in the literature targeting RMC accelerators. *Device size* is the amount of reconfigurable logic used for reconfigurable processing [6]. The presence of on-chip memories and various memory configurations have also been used to classify devices [6,7]. Guccione and Gonzalez 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* [6].

Fault tolerance is important for some mission-critical applications for both fixed and reconfigurable architectures [5]. For networks of devices, reconfigurability of the device-to-device interconnect

is an important classifier [5]. *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 [8]. Vertical and horizontal microinstructions are used to distinguish devices in [9]. 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 [3]. 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* [4,5] 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. It relates processing element width, the number of processing elements, and clock frequency to performance, normalized by die area and process technology [10]. Floating-point performance evaluation methods for RC architectures are explored in [11] 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 parallel operations to performance. We extend this methodology to common integer operations. Several performance comparisons are shown in [12] that demonstrate the applicability of RC technologies to floating-point operations.

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 ongoing architecture and application reformations. The taxonomy proposed is used both as a means to classify devices and to help select the appropriate in-depth

Table 1. 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 \times 1024 deep memory block to a 64-bit \times 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

characterization methods described in Section 5. 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).

3. MC Taxonomy

We propose a hierarchical, tree structure to classify computing devices. 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 processing element 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 1. 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.

4. Accelerator Overview

In this section we describe the features of a variety of FMC and RMC devices so that the CD metrics described in Section 5 can be applied. We have included devices from both 90 nm and 65 nm process technologies. Table 4 provides a summary of classifications for the various devices.

Table 2. FMC Device Features

Device	Cores	Instructions Issued/Core	Datapath Width (bits)	Frequency (GHz)	Process Tech. (nm)	Power (W)
Cell BE	1+8	2+1	64/128	3.2	90	70
Tesla C870	128	2	32	1.35	90	120
Xeon 7041	2	3+1	64/128	3.0	90	165
Xeon X3230	4	4+1	64/128	2.66	65	95

Table 3. FPGA Device Features

Device	LUTs	DSPs	Max. Frequency (MHz)	Process Tech. (nm)	Min. Power (W)	Max. Power (W)
Stratix-II EP2S180	143,520	768	500	90	3.26	30
Stratix-III EP3SE260	203,520	768	550	65	2.11	25
Stratix-III EP3SL340	270,400	576	550	65	2.83	32
Virtex-4 SX55	49,152	512	500	90	1.00	10
Virtex-4 LX200	178,176	96	500	90	1.27	23
Virtex-5 SX95T	58,800	640	550	65	1.89	10
Virtex-5 LX330T	207,360	192	550	65	3.43	27

4.1. FMC Devices

Several FMC devices have been included in this study. Table 2 lists the devices and provides a summary of the key features needed to compute the CD metrics. These devices exhibit very few of the reconfigurability factors listed in Table 1, and are thus classified as FMC devices. This information was gathered from [13,16] for the Cell Broadband Engine (Cell BE), [17-19] for the Nvidia Tesla C870 graphics processing unit (GPU), [20,21] for the Intel Xeon 7041, and [1,2,22] for the Intel Xeon X3230. The Cell BE is a heterogeneous device since it has a traditional processing unit 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 2. The other devices are considered homogeneous because all of the sub-units are the same at the level of replication. The Xeon processors each have a single vector unit per core, again leading to the x+y and 64/128 notation. Note that the 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.

4.2. 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 metrics are listed in Table 3. Note that the Altera Stratix-II FPGA uses 9x9-bit DSP multipliers. The Altera Stratix-III and Xilinx Virtex-4 devices use 18x18-bit multipliers. The Xilinx Virtex-5 devices use 25x18-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 is used for the integer and floating-point metrics. The values in Table 3 were acquired from [23-27]. Power

consumption values were estimated using Altera’s PowerPlay early power estimator and Xilinx’s XPower power estimator spreadsheet tools.

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 16x16-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 [29,30]. We consider it a heterogeneous device.

Element CXI’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

Table 4. Accelerator Classification

Device	FMC	RMC	Hetero	Homo
Arrix FPOA		✓	✓	
ECA-64		✓	✓	
MONARCH	✓	✓	✓	
Stratix-II S180		✓	✓	
Stratix-III SL340		✓	✓	
Stratix-III SE260		✓	✓	
TILE64		✓		✓
Virtex-4 LX200		✓	✓	
Virtex-4 SX55		✓	✓	
Virtex-5 LX330T		✓	✓	
Virtex-5 SX95T		✓	✓	
Cell BE	✓		✓	
Tesla C870	✓			✓
Xeon 7041	✓			✓
Xeon X3230	✓			✓

[31,32]. The goal of the ECA-64 is to provide performance and fault resiliency and to replace many custom processors.

The TILE64 processor from Tiler 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 [33,34]. The goal of the TILE64 is to provide supercomputing performance on a single chip.

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 [35]. The goal of the MONARCH processor is to provide the capability to adapt to changing application requirements.

5. CD 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.

5.1. Bit-level CD

Bit-level CD was originally proposed by DeHon [10]. 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:

$$CD_{bit} = f \times [\sum_i (W_i \times N_i)] \quad (1)$$

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:

$$CD_{bit} = f \times [N_{LUT} + \sum_i (W_i \times N_i)] \quad (2)$$

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 maximum bit-level CD in terms of the clock rate and parallelism (the N terms). It is important to keep in mind 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 [10].

5.2. Integer CD

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 Equation 3:

$$CD_{int} = f \times \sum_i \frac{N_i}{CPI_i} \quad (3)$$

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. 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.

For the FPOA, 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 is used [11]. 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:

$$CD_{int} = (Ops_{DSP} + Ops_{LOGIC}) \times f \quad (4)$$

This is the peak CD without any consideration of potential performance limitations due to memory bandwidth or on-chip RAM resource restrictions for data buffering. Strenski [11] 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. This balance

can be achieved by iterating through combinations of DSP and logic resources allocated to addition or multiplication operations using the methods previously discussed. 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.

5.3. Floating-point CD

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, Equation 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.

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, which is the maximum with 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.

5.4. 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 paper. Within a metric, we hold frequency constant. Therefore, for reconfigurable architectures, we assume that power

Table 5. CD (in billions of operations per second or GOPs)

Device	Bit-level		16-bit Int.		32-bit Int.		SPFP		DPFP	
	Raw	Sustain.	Raw	Sustain.	Raw	Sustain.	Raw	Sustain.	Raw	Sustain.
Arrix FPOA	6144	6144	384	384	192	192				
ECA-64	2176	2176	13	13	6	6				
MONARCH	2048	2048	65	65	65	65	65	65		
Stratix-II S180	63181	63181	442	442	123	123	53	53	11	11
Stratix-III SL340	154422	154422	933	933	213	213	96	96	26	26
Stratix-III SE260	119539	119539	817	817	204	204	73	73	22	22
TILE64	4608	4608	240	240	144	144				
Virtex-4 LX200	89952	89952	357	116	66	42	68	46	16	16
Virtex-4 SX55	29184	29184	365	110	71	40	31	31	7	7
Virtex-5 LX330T	150163	150163	606	300	131	122	119	116	26	26
Virtex-5 SX95T	48435	48435	599	226	221	92	82	82	15	15
Cell BE	4096	4096	205	205	115	115	205	205	19	19
Tesla C870	5530	5530	346	346	216	216	346	346		
Xeon 7041	1536	1536	42	42	30	30	30	30	24	24
Xeon X3230	4095	4095	128	128	85	85	85	85	64	64

scales linearly with resource utilization up to the maximum power consumption specified in vendor documentation in the results that follow. The CD per Watt (CDW) metrics are calculated by taking the CD for each level of parallelism and dividing by the power consumption at that level of parallelism.

6. Results and Discussion

In this section we primarily focus on detailed results for memory-sustainable CDW. Table 5 summarizes both raw and memory-sustainable CD. 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 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 CDW) and achievable frequency is used for the remaining

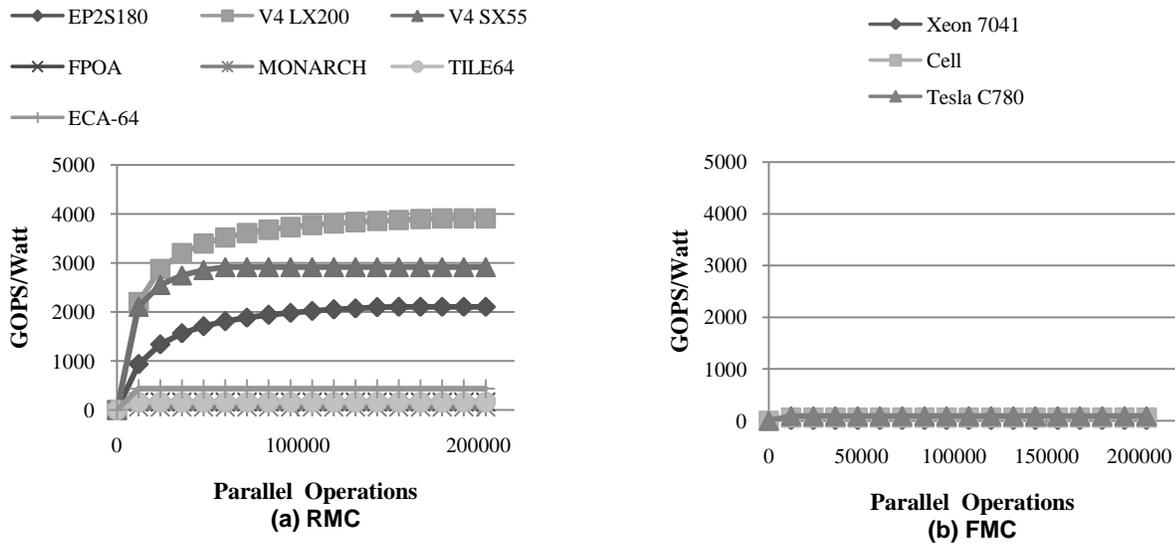


Figure 1. Bit-level CDW (90 nm)

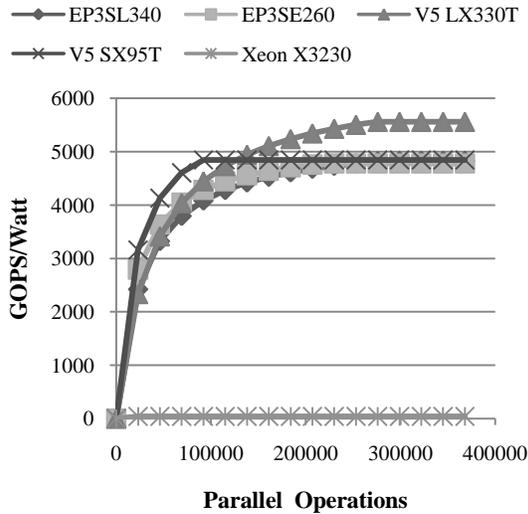
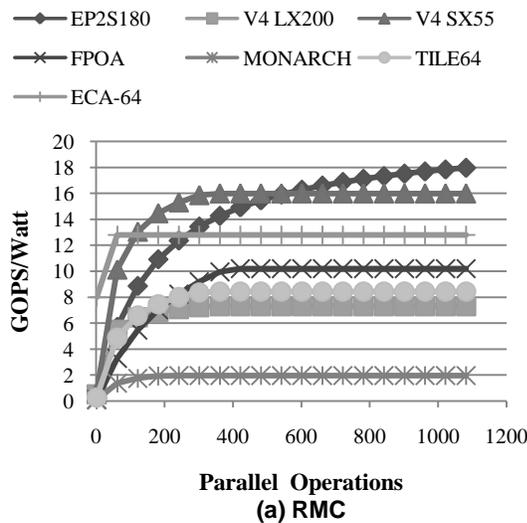


Figure 3. Bit-level CDW (65 nm)

metrics. We then examine the impact of varying parallelism to compare performance. 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) CDW, 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.

6.1. Bit-level CDW

The 65 nm FPGAs have significantly more reconfigurable logic resources than all other devices, including the 90 nm FPGAs, and have the overall best bit-level CDW performance as shown in Figures 1 and 2. The best performer is the Virtex-5 LX330T, 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. For this metric the maximum frequencies listed in Table 3 were used. The FMC devices for both process technologies perform poorly in this metric due to the high overhead for bit operations on these devices and considerably higher power consumption.

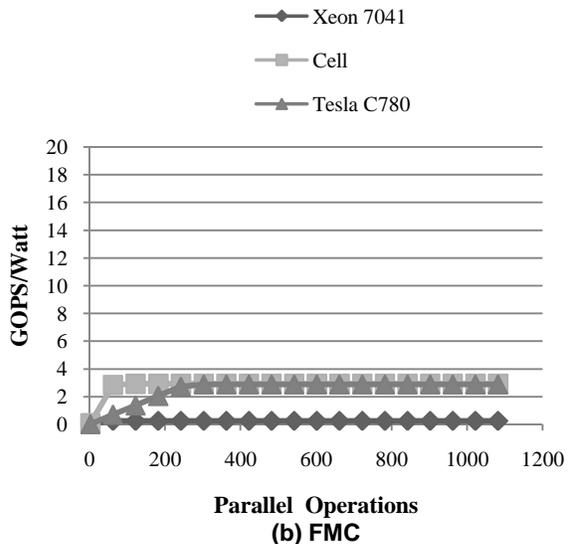


Figure 2. 16-bit Integer CDW (90 nm)

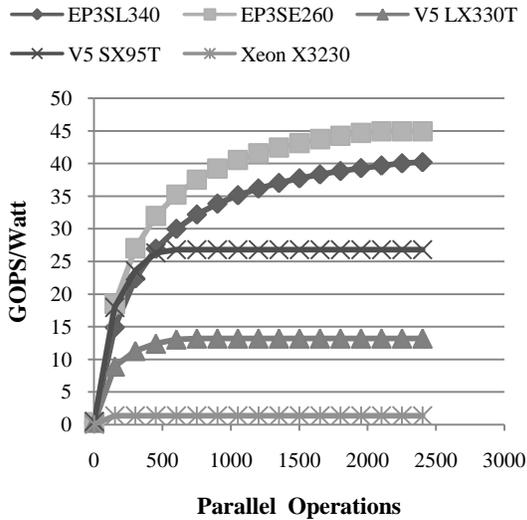


Figure 4. 16-bit Integer CDW (65 nm)

6.2. 16-bit Integer CDW

Figures 3 and 4 show the CDW metrics for 90 and 65 nm devices, respectively. For 90 nm devices the leader for almost all levels of parallelism is the Stratix-II EP2S180, although the ECA-64 and Virtex-4 SX55 perform well. The Stratix-III FPGAs are the clear overall leader, due to their high performance at high levels of parallelism and low power consumption. Again, the FMC devices tend to perform poorly in this metric due to their high, fixed power consumption. For this metric Stratix-II devices had an achievable

frequency of 410 MHz, Stratix-III devices achieved 400 MHz, Virtex-4 devices achieved 344 MHz and Virtex-5 achieved 463 MHz.

6.3. 32-bit Integer CDW

There is an interesting situation for 32-bit CDW, which can be seen in Figure 5. Even though the raw performance of the ECA-64 is relatively low, the power consumption is so low that it initially leads the CDW metric for 90 nm devices. The negative initial slope for the ECA-64 is 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. Overall, the Stratix-III SE260 leads this metric as shown in Figure 6 for high levels of exploitable parallelism. This is 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 66 GOPs, but can only sustain 41.5 GOPs, a 37% reduction. The Virtex-5 LX330T has a raw maximum CD of 131 GOPs, but can only sustain 122 GOPs, a 6% reduction. Memory and buffering limitations lead to a reduction in overall CD and CDW for these devices. Achievable frequencies were 420 MHz for Stratix-II devices, 273 MHz for Stratix-III devices, 249 MHz for Virtex-4 devices, and 378 MHz for Virtex-5 devices.

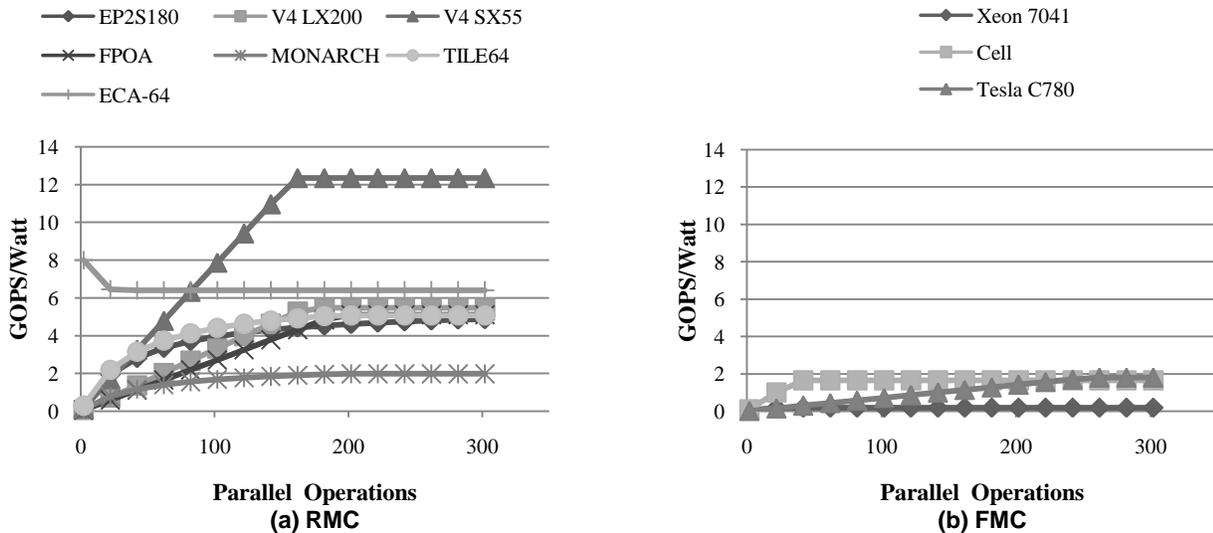


Figure 5. 32-bit Integer CDW (90 nm)

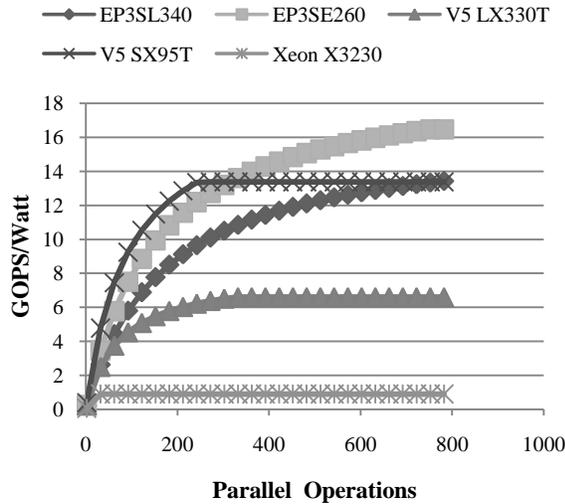


Figure 6. 32-bit Integer CDW (65 nm)

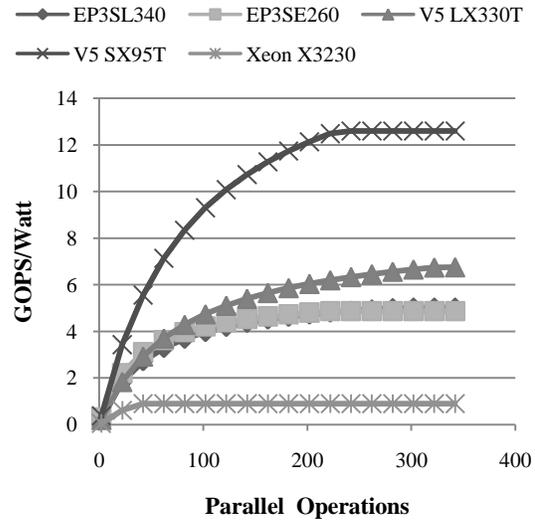


Figure 8. SPFP CDW (65 nm)

6.4. SPFP CDW

Devices that are not intended for SPFP or DFPF operations and would likely perform poorly are not included in the SPFP and DFPF 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 90 nm devices, the Cell and GPU are extremely power-hungry and perform worse on CDW than most of the RMC devices, as shown in Figure 7. 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 CDW for all levels of parallelism, led by the Virtex-5 SX95T as shown in Figure 8. Stratix-II devices had an achievable frequency of 286 MHz, Stratix-III devices achieved 329 MHz, Virtex-4 devices achieved 274 MHz, and Virtex-5 devices achieved 357 MHz.

6.5. DFPF CDW

Although it was the best raw CD performer of the 90 nm devices, the Xeon 7041 was the worst device in terms of CDW due to its very high power consumption, as shown in Figure 9 (a). For 90 nm devices, the Virtex-4 devices were the clear winners

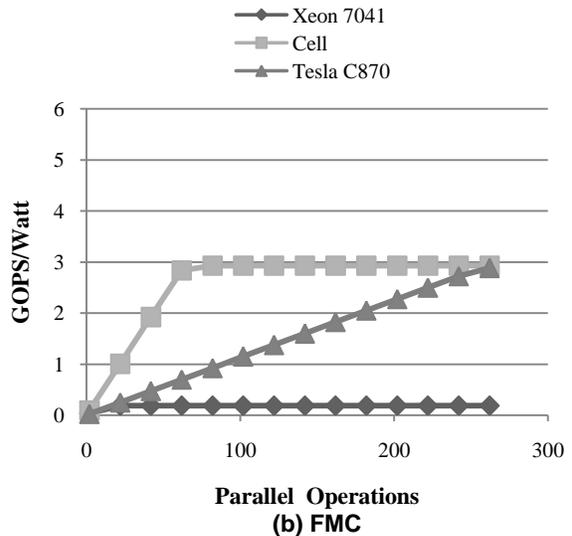
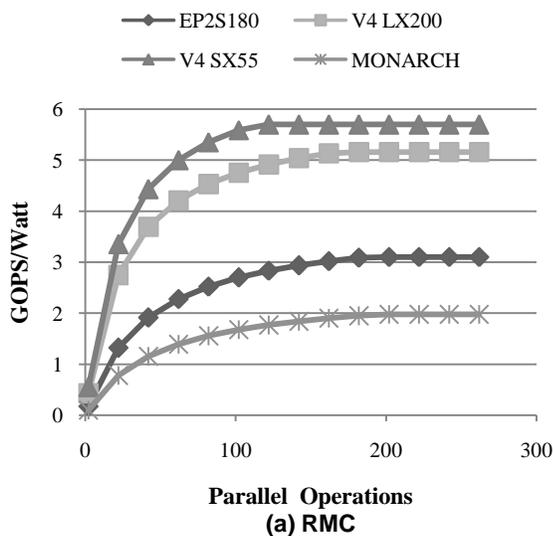


Figure 7. SPFP CDW (90 nm)

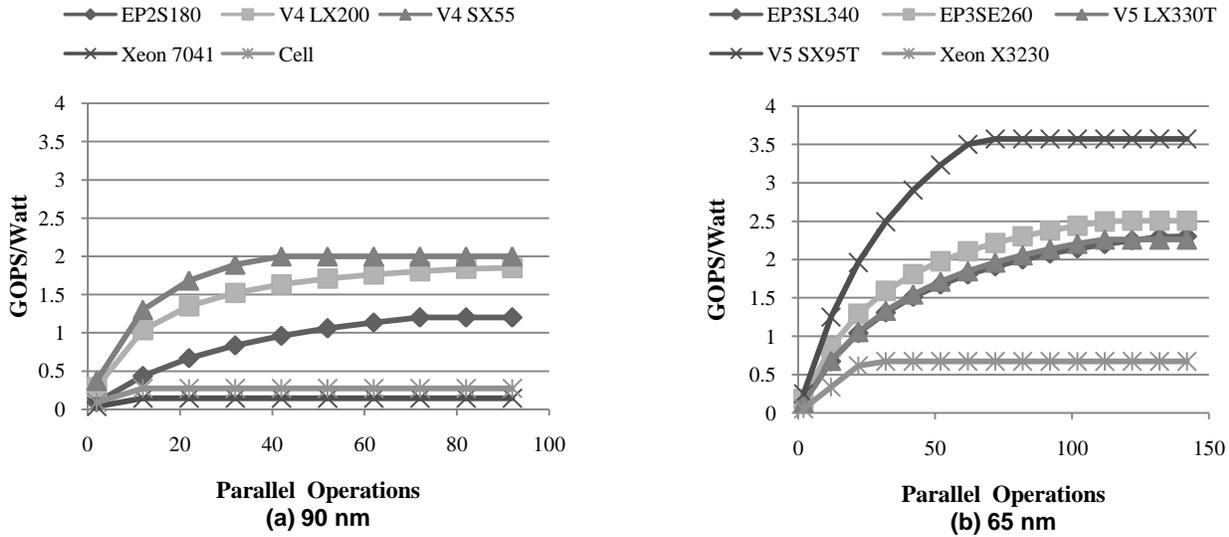


Figure 9. DPFP CDW

for all levels of parallelism for the power-normalized metric. Several of the 65 nm FPGAs performed very well in terms of CDW. The Virtex-5 SX95T had the highest CDW score of all devices, with the other 65 nm FPGAs clustered together. Although it has higher performance, tighter integration of multiple cores on a single die, and improved power efficiency over previous generations, the Xeon X3230 is shown in Figure 9 (b) to continue to lag the 65 nm RMC devices in CDW. Achievable frequencies were 148 MHz for Stratix-II devices, 195 MHz for Stratix-III devices, 185 MHz for Virtex-4 devices, and 237 MHz for Virtex-5 devices.

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 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. These metrics can be used to assist developers in choosing appropriate devices early in the development cycle to avoid wasted design time and unnecessary hardware purchases.

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. Developers may select an FMC device when raw floating-point computational performance is the primary concern, or they may select an RMC device when power efficiency or integer computation is a larger concern. In general, the 65 nm devices performed better on CDW than the 90 nm devices. This demonstrates a key aspect of the architecture reformation: we have not reached the end of Moore's law, but explicit parallelism will allow us to fully utilize process technology advances and increasing transistor counts. 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. On-chip and off-chip memory bandwidth describe the ability of a device to keep its processing elements fed with data. The I/O capabilities and bandwidth are also important considerations in some systems. Finally, cost is another driving factor in device selection. We also plan to explore and analyze application metrics and to develop a mapping between the application metrics and the metrics presented here.

8. Acknowledgments

This work was supported in part by the I/UCRC Program of the National Science Foundation under Grant No. EEC-0642422.

The authors also gratefully acknowledge vendor equipment and/or tools provided by Altera, Mathstar, and Xilinx.

9. References

- [1] Intel Corp., "Intel Xeon Processor X3230," Intel Corp. Web site, retrieved April 17, 2008, <http://processorfinder.intel.com/details.aspx?spec=slacs>.
- [2] Intel Corp., *Product Brief Intel Xeon Processor 3000 Sequence*, 2008.
- [3] M.J. Flynn, "Very High-Speed Computing Systems," *Proceedings of the IEEE*, vol. 54, no. 12, Dec. 1966, pp. 1901-1909.
- [4] K. Compton and S. Hauck, "Reconfigurable Computing: A Survey of Systems and Software," *ACM Computing Surveys*, vol. 34, no. 2, June 2002, pp. 171-210.
- [5] B. Radunovic and V.M. Milutinovic, "A Survey of Reconfigurable Computing Architectures," *FPL '98: Proceedings of the 8th International Workshop on Field-Programmable Logic and Applications*, vol. 1482, no. 1, Aug. 1998, pp. 376-385.
- [6] S. Guccione and M. Gonzalez, "Classification and Performance of Reconfigurable Architectures," *FPL '95: Proceedings of the 5th International Workshop on Field-Programmable Logic and Applications*, vol. 975, no. 1, Aug. 1995, pp. 439-448.
- [7] S. Sawitzki and R.G. Spallek, "A Concept for an Evaluation Framework for Reconfigurable Systems," *FPL '99: Proceedings of the 9th International Workshop on Field-Programmable Logic and Applications*, vol. 1673, no. 1, Aug. 1999, pp. 475-480.
- [8] K. Bondalapati and V.K. Prasanna, "Reconfigurable Computing Systems," *Proceedings of the IEEE*, vol. 90, no. 7, Jul. 2002, pp. 1201-1217.
- [9] M. Sima, et al., "Field-Programmable Custom Computing Machines - A Taxonomy -," *FPL '02: Proceedings of the Reconfigurable Computing Is Going Mainstream, 12th International Conference on Field-Programmable Logic and Applications*, vol. 2438, no. 1, Sept. 2002, pp. 79-88.
- [10] A. DeHon, *Reconfigurable Architectures for General Purpose Computing*, PhD thesis, MIT AI Lab, Sept. 1996.
- [11] D. Strenski, "FPGA Floating Point Performance -- a pencil and paper evaluation," *HPCWire*, Jan. 12, 2007, <http://www.hpcwire.com/hpc/1195762.html>.
- [12] K.D. Underwood and K.S. Hemmert, "Closing the Gap: CPU and FPGA Trends in Sustainable Floating-Point BLAS Performance," *12th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'04)*, Napa, CA, Apr. 2004, pp. 219-228.
- [13] T. Chen, et al., "Cell Broadband Engine Architecture and its First Implementation--A Performance View," *IBM Journal of Research & Development*, vol. 51, no. 5, Sept. 2007, pp. 559-572.
- [14] J.A. Kahle, et al., "Introduction to the Cell Multiprocessor," *IBM Journal of Research & Development*, vol. 45, no. 4/5, Jul./Aug. 2005, pp. 589-604.
- [15] "IBM produces cell processor using new fabrication technology," Mar. 2007, retrieved Nov. 2007, <http://www.xbitlabs.com/news/cpu/display/20070312121941.html>.
- [16] D. Wang, "ISSCC 2005: the Cell Microprocessor," *Real World Technologies*, Feb. 2005, retrieved Jan. 2008, <http://www.realworldtech.com/page.cfm?ArticleID=rwt021005084318&p=2>.
- [17] Nvidia Corp., *Nvidia CUDA Compute Unified Device Architecture Programming Guide*, 2007.
- [18] Nvidia Corp., *Nvidia GeForce 8800 GPU Architecture Overview*, 2006.
- [19] Nvidia Corp., "Nvidia Tesla C870 Specifications," Nvidia Corp. Web site, retrieved Apr. 2008, http://www.nvidia.com/object/tesla_c870.html.
- [20] Intel Corp., "Intel Xeon Processor 7041," Intel Corp. Web site, retrieved Apr. 2008, <http://processorfinder.intel.com/Details.aspx?sSpec=SL8UD>.
- [21] Intel Corp., "Intel NetBurst Architecture," Intel Corp. Web site, Jan. 2000, retrieved Apr. 2008, <http://softwarecommunity.intel.com/articles/eng/3084.htm>.
- [22] Intel Corp., *Inside Intel Core Microarchitecture*, 2006.
- [23] Altera Corp., *Stratix II Device Handbook*, 2007.
- [24] Altera Corp., *Stratix III Device Handbook*, 2007.
- [25] BittWare, Inc., *B2-AMC Data Sheet*, 2008.
- [26] Xilinx, Inc., *Virtex-4 Family Overview*, 2007.
- [27] Xilinx, Inc., *Virtex-5 Family Overview*, 2008.
- [28] Nallatech Ltd., *H100 Series Product Brochure*, 2007.
- [29] Mathstar, Inc., *Arrix Family FPOA Architecture Guide*, 2007.
- [30] Mathstar, Inc., *Arrix Family Product Data Sheet & Design Guide*, 2007.
- [31] Element CXI, Inc., *ECA-64 Device Architecture Overview*, 2007.
- [32] Element CXI, Inc., *ECA-64 Product Brief*, 2007.
- [33] Tiler Corp., *TILE64 Processor Product Brief*, 2008.
- [34] M. Barton, Tiler's Cores Communicate Better, *Microprocessor Report*, Nov. 5, 2007.
- [35] Raytheon Company, *World's First Polymorphic Computer - MONARCH*, 2006.