Arithmetic Operations and their Energy Consumption in the Nios II Embedded Processor

David M. Cambre, Eduardo Boemo and Elías Todorovich
School of Computer Engineering / Universidad Autónoma de Madrid,
Ctra. de Colmenar Km 15, 28049 Madrid
email: davidm.cambre@gmail.es, eduardo.boemo@uam.es, etodorov@uam.es

Abstract

This paper reports the impact of different Nios II hardware and software options for arithmetic operations on its power and energy consumption. These options are evaluated on the Cyclone II and Stratix II FPGA families using a number of benchmark programs. This analysis is part of a more complete study oriented to characterize the power and energy consumption of an embedded processor like the Altera's Nios II. Results are based on physical measurements and show significant energy savings and higher performance in arithmetic operations when available arithmetic hardware suitable for these operations is included. However when the utilization of resources is taken into account, then setups with less hardware and more software for arithmetic computation can be more efficient.

1 Introduction

Performance, cost, and power can be combined in a number of ways in an embedded system specification of a nowadays embedded computer. However the design time, designer availability, and design cost (i.e. a productivity gap) can be a limit in the development of these systems. For this reason the research on automation of embedded computer design has received significant attention during the last years [12] [14]. This automation process needs powerful and efficient tools for design space exploration (DSE). In particular DSE tools require power estimation modules in order to obtain low-power embedded systems. The previous step to develop a power estimation tool for a configurable embedded-system is the analysis and characterization of the energy consumption of such a system.

When large FPGAs were launched into the market and stable software for developing embedded processing systems was released for these devices, the era of the embedded systems on FPGA and the System-on-a-Chip (SoC) on FPGA started. However EDA tools that map an application to a reconfigurable hardware are not mature at all. Again, these tools need power-aware modules for tasks like hardware/software partitioning. Interested readers can find a complete survey on modern reconfigurable embedded systems in [16] including some methods on low-power design.

Conventional studies to power modelling can be conducted on embedded systems in FPGA. However the specific FPGA features enable new possibilities in low power design. This paper studies one of these possibilities. As development tools for embedded systems offer different options related to the computation of arithmetic operations, it is helpful to give some recommendations about the energy implications of these decisions. For example, Altera Nios II cores offer hardware options for integer multiplication: embedded multipliers, logic elements (LE) and none (multiplication is done by software).

This paper presents a power and energy evaluation for different configurations of a Nios II FPGA-based embedded soft-core processor. Besides power, performance is also considered, as well as execution time and area. The results are based only on laboratory measurements. The study was done on two 90-nm FPGAs: Cyclone II and a Stratix II. In despite of this, the idea is to extract general conclusions for embedded processors implemented on programmable logic.

1.1 Related Work

Research in high-level power estimation techniques and low-power design for embedded systems have given some results like [13], a 32-bit processor architecture designed specifically for low-power and portable applications. Also a study ([18]) was conducted on power profiling of the Compaqs Itsy pocket computer [11]. The results suggest power optimization and power management strategies.

It also has been shown that reconfigurable computing can be an interesting platform to implement low-power designs. It is reported in [15], where a Xilinx Virtex XCV50E is used, that moving critical code segments (usually inner

978-0-7695-3474-9/08 $25.00 © 2008 IEEE
DOI 10.1109/ReConFig.2008.37

loops) to hardware results in average energy savings of 34% with an average speedup of 3 times. Another work in the same direction, but implementing these kernels in standard cell ASIC logic, is [10] with average speedups of 3 times while the average energy saving is 59%.

A closely related work is [9] where an energy evaluation for different data and instruction cache sizes on a Nios II is conducted. In another paper, [17], the tradeoffs between performance and power on embedded systems with configurable cache and bus are also studied.

2 Evaluation Setup

This paper evaluates the power and energy consumption of the Nios-II embedded processor on a Cyclone II and a Stratix II development kits from Altera [2] [3]. The software suite used for FPGA synthesis and implementation is the Quartus-II v7.2 [4]. The NIOS-II IDE v7.2 and the GCC compiler were used for editing and compiling the benchmark programs. The specific parts in the evaluation boards used in this work are the Cyclone II EP2C35F672C6N and the Stratix II EP2S60F672C3N respectively.

NIOS-II is a 32-bit RISC embedded processor that can be tailored to a broad range of applications [8]. There are three different types of Nios-II cores available for the user: small, standard and fast. The small and standard cores have a narrow set of configuration parameters and focuses on low demand computing applications.

This paper evaluates the fast core (Nios II/f) running at 85MHz. This core has configurable direct-mapping instruction and data cache sizes, hardware options for multiplication and division, barrel shifter and dynamic branch prediction. The fast core is the most advanced version of the Nios-II. It is recommended for intensive computation programs, but among the three cores it is the one that consumes most power and FPGA resources.

2.1 Experimental Setup

Fig. 1 shows the connection scheme to acquire the power consumed by the FPGAs and in some cases the DDR SDRAM on the development boards. In all the experiments the power consumption at the 5V line remained constant. In this way the 5V line is not taken into account in this work. Only power consumption on the 3.3V(FPGA I/O) and and 1.2V (FPGA core) are considered.

Power is supplied and measurements are taken by the Agilent DC Power Analyzer 6705A [1]. The voltmeter and ammeter measurement accuracies are 0.016%+1.5mV and 0.04%+160μA respectively. This leads in this work to an measurement error of 0.056%+0.24mW.

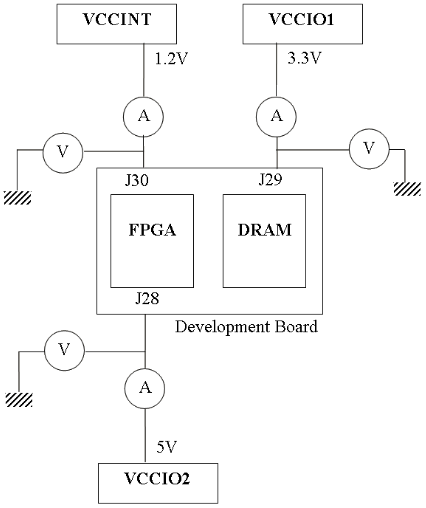

Figure 1. Experimental setup

2.2 Evaluated Arithmetic Options

There are several options for integer and fixed-point multiplication and division, and there is an independent option to use hardware for floating-point operations. In this way the user can balance area (embedded multipliers and logic elements (LE)), performance and energy consumption.

The Nios II/f core offers three options for integer multiplication and two options for integer division. The hardware options for multiplication are:

- DSP Block or Embedded Multipliers: Hard IP cores are included in the arithmetic logic unit (ALU) depending on the target devices. These IP cores can be DSP block multipliers (e.g. in Stratix II) or embedded multipliers (e.g. in Cyclone II).
- Logic Elements: The ALU uses LE-based multipliers.
- None: This option conserves logic resources by eliminating multiplication by specialized hardware elements. In this case the compiler generates routines that performs multiplication and division.

Turning on Hardware Divide includes LE-based divide hardware in the ALU.

Current versions of the Quartus II provide optional hardware for floating-point operations. This is done using a general technique known in Altera as custom instructions [7].

2.3 Benchmark Programs

There are three benchmark programs that are used in this work to evaluate both integer and floating-point options changing the data types of the operands: Matrix multiplication, vector multiplication and vector division.

Table 1. Hardware options evaluated

Hardware setup	None	EM/DSP	LE	HD	CI
EM		Y			
LE			Y		
None	Y				
EM+HD		Y		Y	
LE+HD			Y	Y	
None+HD	Y			Y	
CI+EM		Y			Y
CI+LE			Y		Y
CI+None	Y				Y
CI+EM+HD		Y		Y	Y
CI+LE+HD			Y	Y	Y
CI+HD+None	Y			Y	Y

Four additional programs evaluate integer operations: Dhrystone, finite impulse response filter, prime number computation, and factorial. Evaluation of floating-point operations is performed by the following programs: Fast Fourier transform (FFT) and Pi computation by the Gauss-Legendre algorithm.

Each program is evaluated with all the setups shown in Table 1, where EM/DSP stands for Embedded Multipliers or DSP Blocks, LE for logic elements-based multiplication, HD for Logic Elements-based division, and CI for floating-point custom instructions.

3 Results

The performed measurements are the power and execution times of the benchmarks for these two devices. Resources usage is obtained from the fitter report files generated by Quartus-II.

3.1 Cyclone-II

Cyclone II FPGAs are manufactured on 300-mm wafers using TSMC's 90-nm low-k dielectric [6], and are oriented to low cost and low power consumption. Fig. 2 shows the power consumption measurement for the FPGA Cyclone-II EP2C35. Measurements are arranged by means of a group of bars for each evaluated processor setup, where white and grey bars are related to benchmarks with integer and floating point operands respectively. From left to right, bars within each group represent the power consumption of a benchmark program:

Table 2. Benchmarks and preferred Hardware configurations

Benchmark	EM/DSP	LE	HD	CI
Dhrystone	X	X		
Matrix multiplication	X	X		
Vector multiplication	X	X		
Vector division			X	
Factorial	X	X		
FIR	X	X		
Prime number search			X	
Matrix multiplication				X
Vector multiplication				X
Vector division				X
FFT	X	X	X	X
PI				X

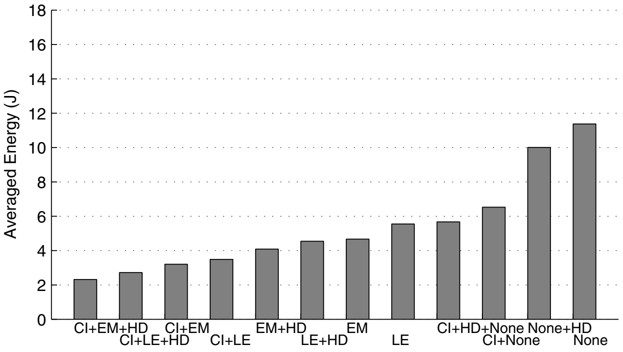

Figure 4. Energy efficiency VS arithmetic setup in Cyclone-II

- Integer operations: Dhrystone; matrix multiplication; vector multiplication; vector division; factorial; FIR signal filter; and prime number search on the first one-hundred integer numbers.
- Floating point operations: matrix multiplication; vector multiplication; vector division; FFT; and first one-hundred iterations of Pi digit calculation.

Table 2 contains the benchmarks labeled with capital letters and the hardware setups that are expected to perform the best, due to the arithmetic instructions found in the corresponding program.

Fig. 2 shows that the power consumption variation of the different running programs range from 300mW to 393mW, this amounts to 24% of the overall FPGA core consumption. This result suggests that the processor core, with its instruction fetch, decode, and execute cycles, dominates the dynamic power consumption of these systems.

On the other hand, Fig. 3 shows the results from the energy point of view. The different hardware setups impact on the execution time of the benchmark programs. Thus, there is an opportunity for significant energy sav-

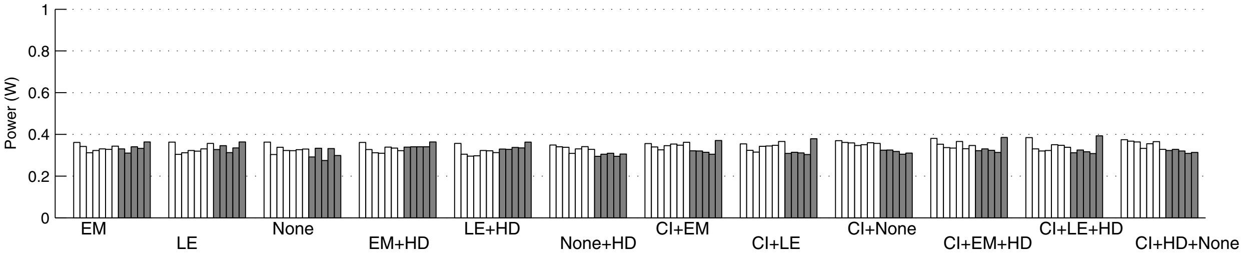

Figure 2. Power consumption VS hardware setup in Cyclone-II

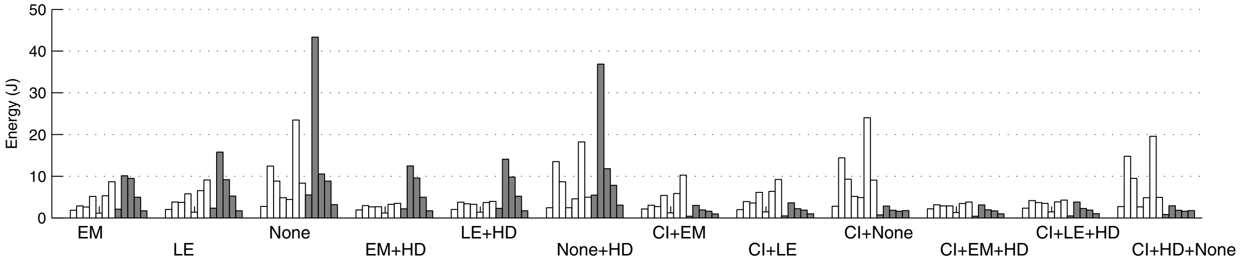

Figure 3. Energy consumption VS hardware setup in Cyclone-II

ings. The six groups on the right side of the figure include hardware-based custom instruction for the floating point operations. When the benchmarks include floating point operations without the specialized hardware (the six groups on the left side), they need more execution time and they are more energy-demanding. Using hardware-based custom instruction for the floating point operation in these cases leads to energy savings from 63% to 78% compared with the software implementation. Hardware multiplication and division is oriented to integer operations. However it has also been observed that including this hardware when floating point operations are involded, achieves energy savings from 9% to 48%. The results also show that benchmarks with integer multiplication and division instructions perform better when hardware for multiplication and division respectively is included. The energy savings range is between 12% and 63%. In these benchmarks, when custom instructions are included a 5% penalty in energy saving has been measured due to the unnecessary increase in resources allocation.

Efficiency for every processor setup has been evaluated as a function of the averaged energy. The average is calculated for each hardware setup over the whole set of benchmarks. Fig. 4 displays the energy efficiency for every setup evaluated in this paper. Processor setups are arranged from the most (left) to the less (right) efficient. Hardware configurations at the left side of the figure include more specialized hardware. It can be observed that efficiency decreases as additional hardware for arithmetic instructions included in the setup diminishes. In the figure the least efficient hardware setups include none or a very reduced set of hardware.

Table 3. Cyclone-II Resources usage

Setup	LE		H. Multip.	
EM	6,065	18%	4	6%
LE	6,437	19%	0	0%
None	5,791	17%	0	0%
HD + EM	6,515	20%	4	6%
HD + LE	6,640	20%	0	0%
HD + None	6,009	18%	0	0%
CI + EM	13,835	42%	11	16%
CI + LE	14,222	43%	7	10%
CI + None	13,568	41%	7	10%
CI + HD + EM	14,304	43%	11	16%
CI + HD + LE	14,326	43%	7	10%
CI + HD + None	13,777	41%	7	10%

The impact on usage of resources is shown in Table 3. The addition of custom instructions doubles the resources used by a hardware setup. Thus it must be considered by the designer whether the resources increase balances the obtained energy savings (up to 78% in the best case).

3.2 Stratix-II

The same evaluation process is conducted for the Stratix-II FPGA from Altera. Stratix-II is based on a 1.2V 90nm SRAM process [5]. These devices are aimed to maximize performance. Power consumption measurement are shown in Fig. 5. As has been observed in the Cyclone-II measurements, power consumption of the FPGA core ranges from

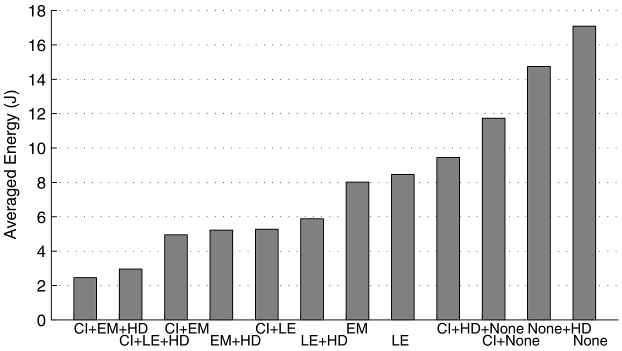

Figure 7. Energy efficiency VS arithmetic setup in Stratix-II

Table 4. Stratix-II Resources usage

Setup	LE		DSP Blocks	
DSP Blocks	3,984	8%	8	3%
LE	4,398	9%	0	0%
None	4,100	8%	0	0%
HD + EM	4,185	8%	8	3%
HD + LE	4,540	9%	0	0%
HD + None	4,144 2	8%	0	0%
CI + EM	10,721	21%	16	6%
CI + LE	11,076	22%	8	3%
CI + None	10,647	21%	8	3%
CI + HD + EM	11,039	23%	16	6%
CI + HD + LE	11,391	24%	8	3%
CI + HD + None	10,900	22%	8	3%

654mw to 831mW (24%). In the Stratix-II FPGA power consumption roughtly doubles the measured value in the Cyclone-II. The increase is due to a higher count of available resources in the Stratix-II.

Fig. 6 shows the energy consumption measurements for the FPGA. Data is organized in the same fashion as above. In benchmarks using floating point operations, energy savings varies from 67% to 84% in setups where custom instructions are included and, as have been observed in Cyclone-II, energy savings range from 9% to 54% for setups with hardware integer multiplication and division. In integer based benchmarks, when additional arithmetic for integer operations is included, the energy savings achieve values from 16% to 77%. In those benchmarks, if custom instructions are included, a 5% penalty in energy saving occurs due to the increase in resources allocation.

Efficiency evaluation is shown in Fig. 7. Processor setups using the larger number of hardware options perform better than other solutions with less dedicated hardware. These results agree with the results in Cyclone-II.

Table 4 contains the usage of DSP Blocks and Logic Elements (LE) resources by the different processor setups. The addition of floating point custom instructions in a setup increases resources usage from 9% to 21%. This increase has also been observed in Cyclone-II. Difference in percentage values between Cyclone-II and Stratix-II is due to a higher count of resources in the Stratix-II FPGAs.

4 Conclusions

On average the Cyclone II device consumes 54% less power and energy than the Stratix II FPGA for the benchmark programs evaluated in this work. For both boards the power variation of the evaluated programs is aproximately 24% of the overall FPGA core consumption. On the other hand variation in execution times, and consequently the energy required for computation, represent an open oportunity in design space exploration.

When hardware for integer (and fixed point) multiplication and division is implemented, up to a 77% energy saving are achieved in programs with integer instructions. This hardware is also useful for floating point instructions: up to 54% energy savings are observed in this study.

When programs with floating point instructions are evaluated and additional hardware for floating point operations is implemented, then up to 84% energy saving is obtained compared to when these instructions are executed just by software. Thus, regards to energy efficiency the best choice is to include all the specific hardware for both integer and floating point instructions. However custom instructions for floating point operations double the processor core area (logic elements and embedded multipliers or DSP blocks).

If energy savings are evaluated only for the preferred hardware list according to Table 2, a 65% in Cyclone II and 75% in Stratix II overall energy savings are achieved. Overall energy saving obtained for a non-discriminant, without taking into consideration the proper hardware accelerator choices for the type of arithmetic operations performed in the benchmark, results are 47% in Cyclone II and 54% in Stratix II.

For efficiency evaluation it can be useful to consider together the resources allocation in the FPGA and the energy saving achieved. In this case, for the Cyclone-II FPGA, hardware setups with custom instructions consumes 70% less energy in average for floating point benchmarks than software based setups and, 40% less than setups with additional hardware for integer operations. But an increase of a factor of two appears in resources allocation compared to setups without custom instructions.

If a cost function is calculated as a function of resources allocation and energy consumption: $Cost = R^n \cdot E^m$, where R stands for resources usage, E stands for energy

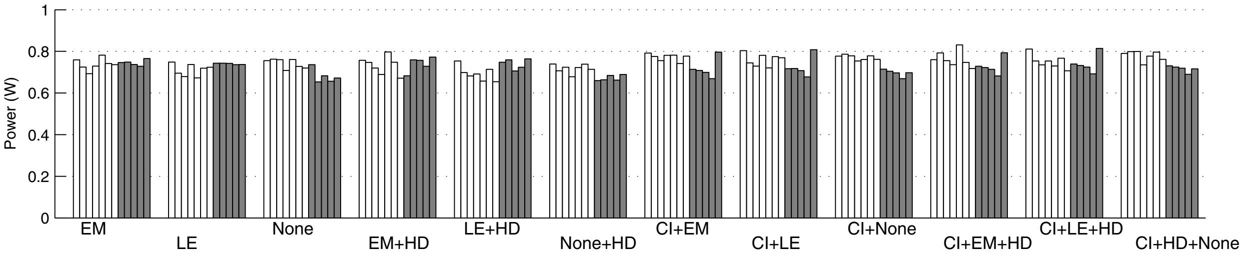

Figure 5. Power consumption VS arithmetic setup in Stratix-II

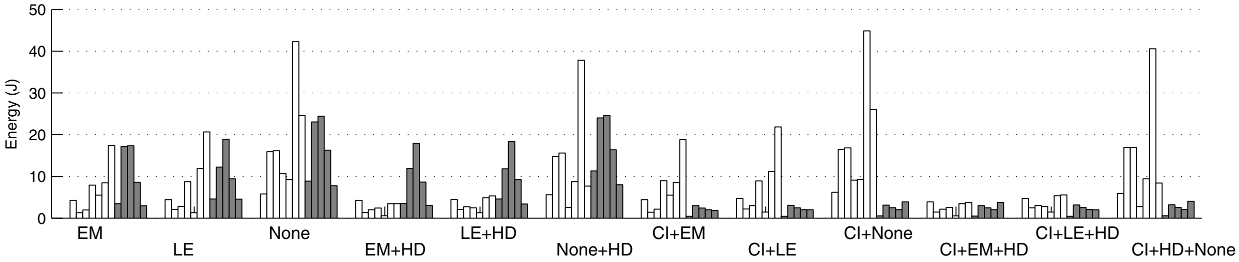

Figure 6. Energy consumption VS arithmetic setup in Stratix-II

consumption, n and m are the weights assigned by the designer after careful evaluation of design constrains. A cost function calculated this way helps the designer to achieve optimal hardware solutions that leverages energy and resources allocation.

5 Acknowledges

This work has been granted by the CICYT of Spain under contract TEC2007-68074-C02-02/MIC. EDA Tools and development boards were provided by Altera Corp. through University Program agreements.

References

[1] Agilent Technologies. *N6705A DC Power Analyzer, Modular, 600W, 4 Slots*, 2008.

[2] Altera corp. *Nios Development Board Reference Manual, Cyclone II Edition*, 2007.

[3] Altera corp. *Nios Development Board Reference Manual, Stratix II Edition*, 2007.

[4] Altera Corp. *Quartus II Handbook*, 2007.

[5] Altera corp. *Stratix II Device Family Data Sheet*, 2007.

[6] Altera corp. *Cyclone II Device Handbook*, 2008.

[7] Altera Corp. *Nios II Custom Instruction user Guide*, 2008.

[8] Altera Corp. *Nios II Processor Reference Handbook*, 2008.

[9] D. M. Cambre, E. Boemo, and E. Todorovich. Energy evaluation in the Nios II processor as a function of cache sizes. *IEEE SPL 08 (Southern Conference on Programmable Logic)*, pages 55–61, March 2008.

[10] M. D. Galanis, G. Dimitroulakos, and C. E. Goutis. Performance and energy consumption improvements in microprocessor systems utilizing a coprocessor data-path. *J. Signal Process. Syst.*, 50(2):179–200, 2008.

[11] W. Hamburgen, D. A. Wallach, M. A. Viredaz, L. S. Brakmo, C. A. Waldspurger, J. F. Bartlett, T. Mann, and K. I. Farkas. Itsy: stretching the bounds of mobile computing. *IEEE Computer*, 34(4):28–37, 2001.

[12] V. Kathail, S. Aditya, R. Schreiber, B. R. Raua, D. C. Cronquist, and M. Sivaraman. Pico: Automatically designing custom computers. *IEEE Computer*, 35(9):39–47, 2002.

[13] B. Moyer. Low-power design for embedded processors. *Proceedings of the IEEE*, 89(11):1576–1587, 2001.

[14] S. Pillement, O. Sentieys, and R. David. Dart: A functional-level reconfigurable architecture for high energy efficiency. *EURASIP Journal on Embedded Systems*, 2008.

[15] G. Stitt, F. Vahid, and S. Nematbakhsh. Energy savings and speedups from partitioning critical software loops to hardware in embedded systems. *Trans. on Embedded Computing Sys.*, 3(1):218–232, 2004.

[16] T. Todman, G. Constantinides, S. Wilton, O. Mencer, W. Luk, and P. Cheung. Reconfigurable computing: architectures and design methods. *Computers and Digital Techniques, IEE Proceedings*, 152:193–207, 2005.

[17] F. V. Tony D. Givargis and J. Henkel. Evaluating power consumption of parameterized cache and bus architectures in sytem-on-a-chip designs. *IEEE Trans. VLSI Syst.*, 9(4):500–508, 2001.

[18] M. A. Viredaz and D. A. Wallach. Power evaluation of a handheld computer. *IEEE Micro*, 23(1):66–74, 2003.