Energy efficient Reconfigurable Computing with Adaptive Voltage and Logic scaling.

Jose Nunez-Yanez
University of Bristol, Electronic Engineering
Merchant Venturers Building, UK
j.l.nunez-yanez@bristol.ac.uk

Abstract— **This paper investigates a novel energy-proportional concept that combines closed-loop voltage scalability and run-time hardware reconfiguration. Voltage scaling is based on in-situ detectors that allow the device to detect valid working voltage and frequency pairs at run-time. The combined approach named AVLS (Adaptive Voltage and Logic Scaling) enables the adaptation of capacitance, voltage and frequency to obtain power and energy savings based on workload, process and operating conditions in a closed-loop configuration. The technique is applied to a reconfigurable motion estimation processor that can be configured with a variable number of execution units and it is used as a test vehicle. The results demonstrate that the proposed voltage scaling can obtain up to 85% reduction in energy compared with nominal voltage operation at the same frequency. This efficient energy point is obtained at a voltage of 0.62 V and frequency of 56 MHz compared with running the core at the same frequency and nominal 1 V. The addition of logic scalability means that if enough device resources are available a parallel configuration with six execution units operating at 0.62 V reduces energy by up to 95% compared with a single execution unit operating at 1 V and the same frequency.**

Index Terms— **FPGA, energy efficiency, DVFS, AVS.**

1. INTRODUCTION

Energy and power efficiency in Field Programmable Gate Arrays (FPGAs) has been estimated to be up to one order of magnitude worse than in ASICs [1] and this limits their applicability in energy constraint applications. Since FPGAs are fabricated using CMOS transistors power can be divided into two main categories, dynamic power and static power. Fig 1 shows a simplified relation of power with voltage, frequency and capacitance.

$$\text{Power} = \alpha C V^2 F + g1V^3 + g2V^5 \quad (1)$$

Where α = Activity factor < 1, C = Capacitance, V = Supply voltage, F = Frequency, g1 = sub-threshold leakage and g2 = gate leakage. The first term of the equation represents dynamic power and the other two static power. This equation shows how voltage affects both static and dynamic power. It is apparent that lowering the supply voltage in CMOS circuits reduces both dynamic and static power but it also has a cost in increased circuit delay. As a result, voltage scaling is often combined with frequency scaling in order to compensate for the variation of circuit delay. An example of this is Dynamic Voltage and Frequency Scaling (DVFS) which is a technique that uses a number of pre-evaluated voltage and frequency operational points to scale power, energy and performance. With DVFS, margins for worst case process and environmental variability are still maintained since it operates in an open-loop configuration. However, worst case variability is rarely the case. For that reason, in Adaptive Voltage Scaling (AVS), run-time monitoring of performance variability in the silicon is used together with system characterization to influence the voltage and the frequency on the fly in a closed-loop configuration. This paper applies AVS to commercial FPGAs and its contributions can be summarised as follows:

1. We present a power adaptive architecture that includes run-time support for voltage, frequency and logic adaptation in commercial FPGAs.
2. We demonstrate the power and energy savings possible using as a test case a reconfigurable motion estimation processor with a variable number of execution units.

The rest of the paper is structured as follows. Section 2 describes related work. Section 3 presents the hardware platform that includes an off-chip voltage scaling circuit that can be controlled directly by the FPGA to regulate its own power. Section 4 presents the AVLS management IP with the voltage, frequency and logic scaling units. Section 5 presents and discusses the results focusing on power and energy measurements. Finally, section 6 presents the conclusions and future work.

2. RELATED WORK

In order to identify ways of reducing the power consumption in FPGAs, some research has focused on developing new FPGA architectures implementing multi-threshold voltage techniques, multi-Vdd techniques and power gating techniques [2-6]. Other strategies have proposed modifying the map and place&route algorithms to provide power aware implementations [7-9]. This related work is targeted towards

FPGA manufacturers and tool designers to adopt in new platforms and design environments. On the other hand, a user level approach is proposed in [10]. A dynamic voltage scaling strategy for commercial FPGAs that aims to minimise power consumption for a giving task is presented in their work. In this methodology, the voltage of the FPGA is controlled by a power supply that can vary the internal voltage of the FPGA. For a given task, the lowest supply voltage of operation is experimentally derived and at run-time, voltage is adjusted to operate at this critical point. A logic delay measurement circuit is used with an external computer as a feedback control input to adjust the internal voltage of the FPGA (VCCINT) at intervals of 200ms. With this approach, the authors demonstrate power savings from 4% to 54% from the VCCINT supply. The experiments are performed on the Xilinx Virtex 300E-8 device fabricated on a 180nm process technology. The logic delay measurement circuit (LDCM) is an essential part of the system because it is used to measure the device and environmental variation of the critical path of the functionality implemented in the FPGA and it is therefore used to characterise the effects of voltage scaling and provide feedback to the control system. This work is mainly presented as a proof of concept of the power saving capabilities of dynamic voltage scaling on readily available commercial FPGAs and therefore does not focus on efficient implementation strategies to deliver energy and overheads minimisation. A similar approach is also demonstrated in [11]. A dynamic voltage scaling strategy is proposed to minimise energy consumption of an FPGA based processing element, by adjusting first the voltage, then searching for a suitable frequency at which to operate. Again, in this approach, first the critical path of the task under test is identified, then a logic delay measurement circuit is used to track the critical point of operation as voltage and frequency are scaled. Significant savings in power and energy are measured as voltage is scaled from its nominal value of 1.2V down to its limit of 0.9V. Beyond this point, the system fails. The experiments were carried out on a Xilinx ML402 evaluation board with a XC4VSX35-FF668-10C FPGA fabricated in a 90 nm process and energy savings of up to 60% are presented.

The previously presented efforts are based on the deployment of delay lines calibrated according to the critical path of the main circuit. This calibration is cumbersome and it could lead to miss tracking due for, for example, the different locations of the delay line and the critical paths of the circuit having different temperature profiles. In-situ detectors located at the end of the critical paths remove the need for calibration. This technology has been demonstrated in custom processor designs such as those based around ARM Razor [12]. Razor allows timing errors to occur in the main circuit which are detected and corrected re-executing failed instructions. The lastest incarnation of Razor uses a highly optimized flip-flop structure able to detect late transitions that could lead to errors in the flip-flops located in the critical paths. The voltage supply is lower from a nominal voltage of 1.2V (0.13μm CMOS) for a processor design based on the Alpha microarchitecture observing approximately 33% reduction in energy dissipation with a constant error rate of 0.04%. The Razor technology requires changes in the microarchitecture of the processor and it cannot be easily applied to other non-processor based designs. It requires a specialized flip-flop structure that cannot be implemented in off-the-shelf FPGAs. Other work has shown the power and energy benefits of deploying voltage and frequency scaling with in-situ detectors in off-the-self FPGAs [13] based on the same controlled logic cell placement used in this work and also recently [14] that uses a recalibration technique to remove the variable delays introduced by a detector that with variable placement and routing. In this paper we extend previous research by adding the logic scaling variable. This means that we evaluate the power and energy synergies possible by deploying voltage scaling with user designs that can in addition be re-configured at run-time with different levels of complexity and performance. The presented approach uses the technology primitives and elements already available in the FPGAs and therefore does not require chip fabrication or redesign in order to be used.

3. PLATFORM DESCRIPTION

The research platform used is the Xilinx XUPV5-LX110T evaluation board (XUPV5) with a Virtex-5 XC5VLX110T FPGA manufactured in a 65nm process technology. The XC5VLX110T is conventionally powered by DC-to-DC power supplies that ensure fixed, stable and noise free supplies to three main voltage sources; VCCAUX, VCCO and VCCINT. VCCAUX provides power to the clock resources and clock primitives in the FPGA. VCCO provides power to the input and output banks of the device. VCCINT provides power to the logic resources of the device such as flip-flops, LUTs, configuration memory etc and as a result, heavily influences static and dynamic power. Static and dynamic power have approximately a quadratic and cubic dependency on voltage respectively.

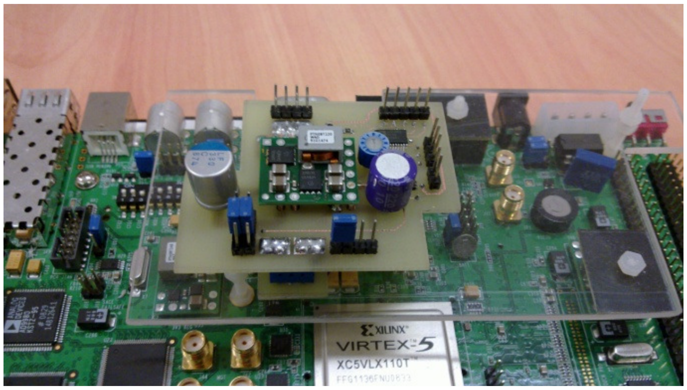


Figure 1. Voltage scaling PCB

To vary the power consumption of the FPGA, voltage scaling is applied to the VCCINT voltage source. To achieve this, the DC-to-DC module that supplies the VCCINT voltage to the FPGA was redesigned to provide variable voltage without affecting the other voltage sources to the device. This was accomplished by first designing a voltage scaling module on a printed circuit board (PCB), then the original DC-to-DC module that provides a fixed voltage to the VCCINT terminal of the FPGA was replaced by the voltage scaling PCB as shown in Fig.1. To permit the FPGA to control its own voltage, the control interface to the voltage scaling module - which uses the Serial Peripheral Interface (SPI) protocol - was connected to the general purpose I/O interface of the FPGA. Within the FPGA, a SPI slave controller was implemented. With this approach, a system configured in the FPGA can control its own voltage by adjusting the value of the digital potentiometer in the voltage scaling module through the SPI controller.

4. ADAPTIVE VOLTAGE AND LOGIC SCALING ARCHITECTURE

Fig. 2 shows the AVLS architecture. The AVLS management unit will initially configure a requested user design in the USER logic making use of the logic scaling unit and the ICAP port. The logic scaling unit (LSU) is formed by a dynamic reconfiguration controller described in [15] used to load new bitstreams into the dynamic FPGA fabric. The reconfiguration controller interfaces with the ICAP port which is the hardwired unit available in the FPGA device that enables access to the configuration fabric from within the device. The ICAP port offers a 32-bit interface and it can be clocked up to 100 MHz in the Virtex-5 device considered according to the FPGA manufacturer producing a maximum reconfiguration throughput of 400 Mbytes per second (32 bits per clock cycle). The ICAP port has also the capability of both reading and writing the configuration memory.

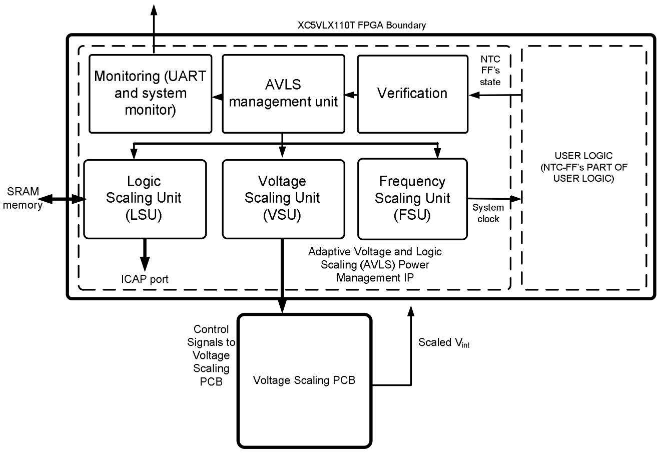

XC5VLX110T FPGA Boundary
Monitoring (UART and system monitor)
AVLS management unit
Verification
NTC FF's state
Logic Scaling Unit (LSU)
Voltage Scaling Unit (VSU)
Frequency Scaling Unit (FSU)
System clock
USER LOGIC (NTC-FF's PART OF USER LOGIC)
SRAM memory
ICAP port
Adaptive Voltage and Logic Scaling (AVLS) Power Management IP
Control Signals to Voltage Scaling PCB
Voltage Scaling PCB
Scaled V_{int}

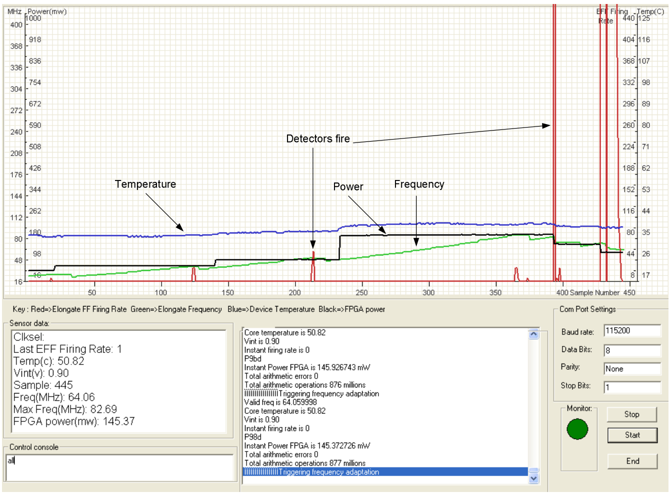
Figure 2. AVLS architecture

designed to provide fast partial reconfiguration services with minimal overheads and maximum autonomy. All partial bitstreams are stored in external SRAM memory. This is a reasonable approach to follow since the FPGA device does not have enough on-chip memory to do this and the available on-chip memory is required for the data processing itself. This external memory provides a good trade-off between memory size, transfer overheads and on-chip resource utilisation. With this approach, the on-chip resources are used to control the transfer of bitstreams from the SRAM memory to the ICAP (Internal Configuration Access Port). A simple reconfiguration protocol is used to maximise autonomy. The reconfiguration controller receives the memory location and the size of the bitstream of interest from the AVLS management unit. Once the start signal is given, it retrieves the bitstream and performs partial dynamic reconfiguration autonomously. Once the reconfiguration process has completed, the AVLS management unit is informed with an interrupt and adaption of voltage and frequency can start. The complexity of this AVLS IP is small and summarized in Table 1. The BRAM used in AVLS IP stores the frequency table parameters for the Digital Clock Managers. The power required by this additional logic is negligible.

Table 1. AVLS complexity

	FFs	LUTs	BRAMs
AVLS IP (without ICAP controller)	312	216	1
LSU Unit (ICAP controller)	349	229	1

To detect valid voltage and frequency working points the user design has been embedded with in-situ timing detectors located at the end-points of the critical paths. These detectors fire when the voltage and frequency point is near generating a timing failure. The verification module receives the outputs from the in-situ detectors and proceeds to AND all these outputs to detect any timing violations. In the current setup once a voltage level has been setup the management unit finds the highest frequency that can be supported with that voltage. This is done by the FSU unit generating multiple frequencies until the detectors in the user logic start firing. A frequency generation ROM memory forms part of the FSU. This ROM contains values for the Digital Clock Managers (DCM_ADV) also part of the FSU that generate the clock for the user logic. The outputs obtained from this memory are written by the state machines part of the FSU using the reconfiguration port available in the DCM_AVD block and new frequencies are generated at run-time. Once the DCM_ADV's have locked the clock is driven into the user logic. Once the frequency reaches a value that causes timing violations these are reported by the detectors and the state machine stops increasing the frequency until a new higher voltage is configured in the system. An additional optional unit present in Fig.2 is the monitoring unit that instantiates the system monitor IP block available in the FPGA device to monitor internal variables such as temperature, voltage and verification status. These monitored parameters are not used to make decisions on frequency and voltage points but to inform aPC-based monitoring software of system status. A UART is part of the monitoring unit used to output this state to a host PC. A screen-shot of the monitoring software running in the host PC is shown in Fig. 3. The upper line in Fig. 3 corresponds to the chip temperature. The next line is power and six different power zones can be identified that correspond to operating voltages 0.75, 0.80, 0.85, 1.0, 0.95 and 0.9 starting from the left. The system finds an optimal working frequency for each of this voltage points automatically as shown in the frequency line that can be seen increasing or decreasing (third line from the top). Finally the bottom line in the figure shows the detectors firing when the optimal working frequency is located. It is important to note that during all the experiments reported in this paper no functional errors are found in the user logic application. The outputs generated by the user logic are compared with the expected outputs and they agree always. This means that this technique as deployed in this paper does not imply any loss of accuracy.



Key : Red=>Elongate FF Firing Rate Green=>Elongate Frequency Blue=>Device Temperature Black=>FPGA power

Sensor data:

```
Clksel:
Last EFF Firing Rate: 1
Temp(c): 50.82
Vint(v): 0.90
Sample: 445
Freq(MHz): 64.06
Max Freq(MHz): 82.69
FPGA power(mw): 145.37
```

Control console: all

```
Core temperature is 50.82
Vint is 0.90
Instant firing rate is 0
P9bd
Instant Power FPGA is 145.926743 mW
Total arithmetic errors 0
Total arithmetic operations 876 millions
||||||||||||||||Triggering frequency adaptation
Valid freq is 64.059998
Core temperature is 50.82
Vint is 0.90
Instant firing rate is 0
P98d
Instant Power FPGA is 145.372726 mW
Total arithmetic errors 0
Total arithmetic operations 877 millions
||||||||||||||||Triggering frequency adaptation
```

Baud rate: 115200
Data Bits: 8
Parity: None
Stop Bits: 1

Figure 3. Monitoring software view at different voltages

5. POWER, PERFORMANCE AND COMPLEXITY ANALYSIS

A test system based around a reconfigurable motion estimation processor has been built to explore the effects of combining voltage, frequency and logic scalability in the same FPGA chip Full details of this processor core that can use a variable number of execution units are described in [16]. Three different motion estimation processor configurations with 1, 3 and 6 execution units are considered. The corresponding netlists for each of the hardware configurations are processed and 100 critical paths are protected with in-situ detection logic. Fig. 4 shows the complexity and performance of the resulting implementations identified with the elongate name compared with the original designs after adding the in-situ detectors. The total number of slices where the logic is mapped increases approximately 10% compared the original design. The additional flip-flops are designed to map the logic tightly in the slice so the number of additional slices required is moderate. The additional logic effects in circuit delay are low. Although it is reasonable to expect an increase due to the presence of the slower flip-flops associated to the in-situ detection logic the variation is mainly due to the place&routing algorithms and it could change with different tool versions, constraints or designs. Fig. 5 shows the valid frequency and voltage points for the three hardware configurations considered. The minimum stable voltage is 0.62 volts. Lower voltages create problems in the lock signal of the DCM_ADV blocks so they have not been used. For this voltage the circuit auto-detects a valid working frequency at around 50 MHz. The original design frequency as reported by the tools after static timing analysis are approximately 159, 156 and 131 MHz at nominal voltage of 1.0 V for me1, me3 and me6 respectively. The critical paths of the more complex configurations are longer since the final stage in the pipeline which needs to decide which is the winning point calculated by each of the execution units gets progressively more complicated. Additional the place and routing tools tend to obtain worse solutions as the complexity of the designs increase due to routing congestions and this also contributes to the decrease in performance. The maximum valid working frequency detected at run-time is higher achieving 240 MHz at 1.0 V for the me1 design. This suggests that the margins needed to compensate for voltage, temperature and process variations can be effectively exploited to obtain both higher performance designs and lower power/energy profiles in commercial FPGAs. Fig. 6 analyzes the power consumption for the different configurations. Each frequency point uses an optimal voltage point. All these values correspond to power measured in the board which is the case for all the experiments reported in this paper. The operating frequency range extends from approximately 50 Mhz to 240 MHz. The low frequency point could be lower since the DCM_ADV blocks can generate a valid frequency down to 22 MHz but the system locates the 50Mhz frequency as the most energy efficient (higher frequency) for the 0.62 V operating point. As expected the configuration with lower complexity (me1) results in lower power. It is reasonable to expect this result since for the same frequency and voltage only capacitance is left which is significantly lower for me1 compared with me3 or me6.This experiment shows the important reduction in power or increase in performance that can be achieved with the voltage and frequency scaling approach in a standard FPGA device. For example, in Fig.6 the maximum performance point of 168 MHz requires 4.64 mW/MHz, a medium performance point of 100 MHz requires 3.3 mW/MHz while a low performance point of 54 MHz requires 2.38 mW/MHz. It also shows the more complex hardware configurations need more power so if absolute power was the only design objective the configuration with just one execution unit will be the one selected. An important consideration is that wider configurations with more execution units reduce total execution time and this has the potential of reducing energy as long as the increase in power does not offset the reduction in time. To further clarify these results the final experiment compares the three hardware configurations when the computation is always active. We consider that the motion estimation processor is receiving constant requests that it must complete within a time period. In this experiment there are no idle states and the system is always active.

.

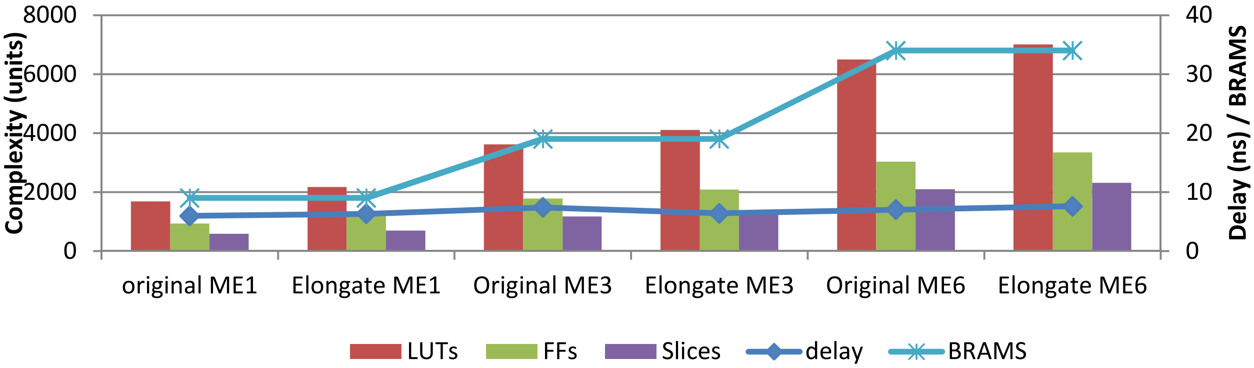


Figure 4. Comparison of voltage scaling enable configurations

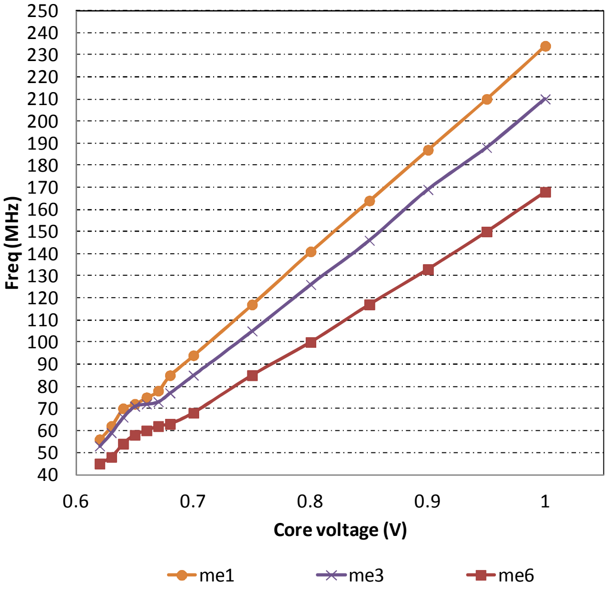


Figure 5. Frequency and voltage analysis

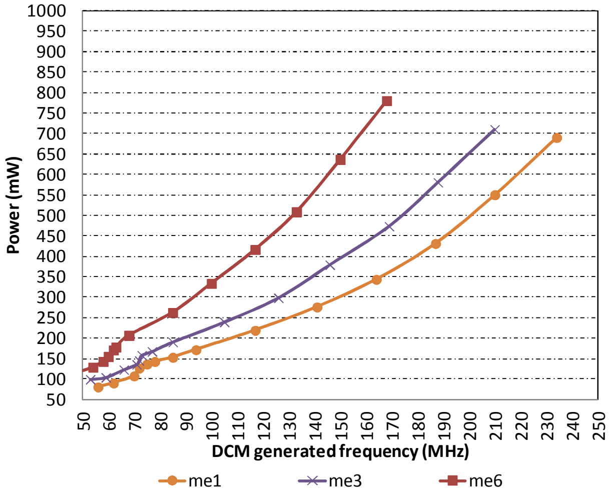


Figure 6. Measured static and dynamic power distribution.

To meet the computation requests the me1 configuration needs to run at 245 MHz and this requires a full voltage of 1 V. Operating at this rate the power is 720 mW. If it is possible to deploy the next hardware configuration of me3 then a frequency of 83 MHz is required to achieve the required performance. The 83 MHz frequency only requires a 0.7 voltage and this reduces power to 190 mW. Finally, if it is possible to deploy the me6 hardware configuration then only 42 MHz are required. The minimum voltage of 0.62 V is sufficient for this configuration and this translates into a power requirement of 100 mW. These results are illustrated in Fig 7 that summarises how the higher performance achieve by the parallel configurations means that a low voltage is required resulting in significant power savings. In this case and since time is constant for all the configurations (always active) the energy savings are equivalent to the power savings.

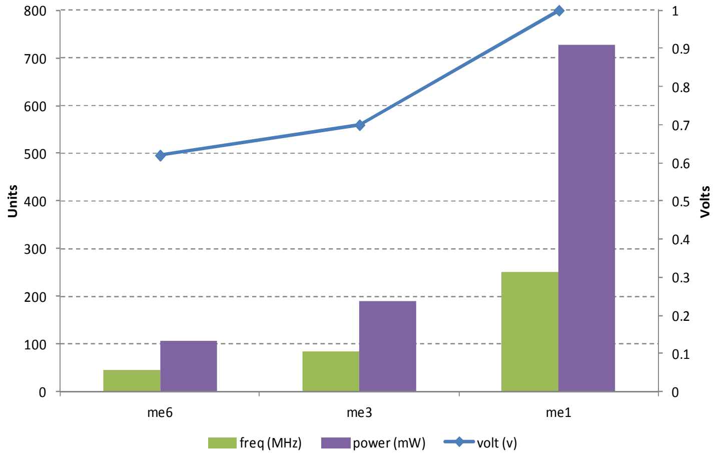


Figure 7. Energy for always on tasks in AVLS

6. CONCLUSION AND FUTURE WORK

This paper has investigated a novel power and energy efficient concept named AVLS that uses voltage and logic scaling in modern FPGAs. Voltage scaling is based on the integration of in-situ detectors coupled to the critical paths of the design to create a robust architecture that removes the need of delay line calibration. Although the FPGA devices employed have not been validated by the manufacturer at below nominal voltage operational points, the investigation shows that savings approaching one order of magnitude are possible by exploiting the margins and overheads available in the devices. Logic scaling is obtained with partial dynamic reconfiguration that enables changing the hardware configuration at run-time. The investigation is based on a reconfigurable motion estimation processor that can be implemented with a variable number of execution units. The results show that adaptive voltage and logc scaling is possible in commercial FPGAs and that, as expected, parallel configurations with reduced voltage and frequency can reduce energy and power by up to 85%. Future work involves using the technology with other FPGA devices manufacture in different process nodes (e.g 40 nm and 28 nm) to investigate the margins that exist at lower feature sizes and also exploring how this run-time voltage, logic and frequency mapping can be incorporated into an energy-aware operating system.

7. REFERENCES

1. Kuon, I. and Roise, J. 2007. Measuring the gap between fpgas and asics. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 26, 2, 203 – 215.
2. Rahman, A., Das., Tuan T., and Rahut, A. 2005. Heterogeneous routing architecture for low-power FPGA fabric. In Custom Integrated Circuits Conference, 2005. Proceedings of the IEEE 2005. pp. 183 – 186.
3. Ryan, J. and Calhoun, B. 2010. A sub-threshold fpga with low-swing dual-vdd interconnect in 90nm cmos. In Custom Integrated Circuits Conference (CICC), 2010 IEEE. pp. 1 –4.
4. Li, F., Lin, Y., and He, L. 2004. Vdd programmability to reduce fpga interconnect power. In Computer Aided Design, 2004. ICCAD-2004. IEEE/ACM International Conference on. pp. 760 – 765.
5. Li, F., Lin, Y., He, L., and Cong, J. 2004. Low-power fpga using pre-defined dual-vdd/dual-vt fabrics. In Proceedings of the 2004 ACM/SIGDA 12th international symposium on Field programmable gate arrays. FPGA '04. ACM, New York, NY, USA, 42–50.
6. Raham A. and Polavarapuv, V. 2004. Evaluation of low-leakage design techniques for field programmable gate arrays. In Proceedings of the 2004 ACM/SIGDA 12th international symposium on Field programmable gate arrays. FPGA '04. ACM, New York, NY, USA, 23–30.
7. Lamoureux, J. and Wilton, S. . On the interaction between power-aware fpga cad algorithms. In Computer Aided Design, 2003. ICCAD-2003. International Conference on. 701 – 708.
8. Lamoureux, J. and Wilton, S. 2007. Clock-aware placement for FPGAs. In Field Programmable Logic and Applications, 2007. FPL 2007. International Conference on. 124 –131.
9. Gayasen, A., Tsai, Y., Vijaykrishnan, N., Kandemir, M., Irwin, M. J., and Tuan, T. 2004. Reducing leakage energy in fpgas using region constrained placement. In Proceedings of the 2004 ACM/SIGDA 12th international symposium on Field programmable gate arrays. FPGA '04. ACM, New York, NY, USA, 51–58.
10. Chow, C., Tsui, L., Leong, P., Luk, W., and Wilton, S. 2005. Dynamic voltage scaling for commercial FPGAs. In Field-Programmable Technology, 2005. Proceedings. 2005 IEEE International Conference on. 173 –180.
11. Chouliaras,, V., and Gaisler, J. 2007. Dynamic voltage scaling in a FPGA-based system-on-chip. In Field Programmable Logic and Applications, 2007. FPL 2007. International Conference on. pp. 459 –462.
12. S. Das, et al., Razor II, IEEE J. Solid-State Circuits, pp. 32--48, Jan. 2009.
13. Nunez-Yanez J, "Adaptive Voltage Scaling with in-situ Detectors in Commercial FPGAs", IEEE transactions in computers preprint, DOI http://doi.ieeecomputersociety.org/10.1109/TC.2013.73
14. Joshua M. Levine, Edward Stott, and Peter Y.K. Cheung. 2014. Dynamic voltage & frequency scaling with online slack measurement. In Proceedings of the 2014 ACM/SIGDA international symposium on Field-programmable gate arrays (FPGA '14)
15. Nabina A, "Dynamic Reconfiguration Optimisation with Streaming Data Decompression," Field Programmable Logic and Applications (FPL), 2010 International Conference on , vol., no., pp.602,607, Aug. 31 2010-Sept. 2 2010.
16. Hung, E.; Vafiadis, G., "Cogeneration of Fast Motion Estimation Processors and Algorithms for Advanced Video Coding," Very Large Scale Integration (VLSI) Systems, IEEE Transactions on , vol.20, no.3, pp.437,448, March 2012.