Accurate Power control and monitoring in ZYNQ boards

Arash Farhadi Beldachi, Jose L. Nunez-Yanez
Department of Electronic Engineering
University of Bristol
Bristol, UK
{arash.beldachi, j.l.nunez-yanez} @ bristol.ac.uk

***Abstract*—ZYNQ devices combine a dual-core ARM Cortex A9 processor and a FPGA fabric in the same die and in different power domains. In this paper we investigate the run-time power scaling capabilities of these devices using of-the-shelf boards and proposed accurate and fine-grained power control and monitoring techniques. The experimental results show that both software and hardware methods are possible and the right selection can yield different results in terms of control and monitoring speeds, accuracy of measurement, power consumption, and area overhead. The results also demonstrate that significant power margins are available in the FPGA device with different voltage configurations possible. This can be used to complement traditional voltage scaling techniques applied to the processor domain to obtain hybrid energy proportional computing platforms.**

Keywords—Adaptive Voltage Scaling; Power analysis; FPGA; Xilinx;ZYNQ board;

I. INTRODUCTION

Energy and power efficiency in FPGAs has been estimated to be up to one order of magnitude worse than in ASICs [1] and this limits their applicability in energy constrained applications. According to device vendors, recent 28nm FPGAs consume 50% less power than previous generations [2] and this contributes to close this power gap. Additional power savings are possible if FPGAs can make use of techniques such as Adaptive Voltage Scaling (AVS) which results in significant reduction of the dynamic and static power by dynamically adjusting voltage and frequency in a closed-loop configuration. AVS is a power-saving technique that enables a device to regulate its own voltage and frequency based on workload, fabrication, and operating conditions and compares favourably with open-loop Dynamic Voltage and Frequency Scaling (DVFS). Our previous work [3, 4] presented a novel design flow and IP library that enable the integration of closed-loop variation-aware adaptive voltage scaling in commercial FPGAs. This approach adapts the operational point over a wide range of voltage and frequency levels at run-time adapting to temperature, process and workload changes automatically. The investigations reveal that although the device has not been validated by the manufacture at below nominal voltage operational points; savings approaching one order of magnitude are possible by exploiting the margins available in the chip.

In [5], we extended the work of [3, 4] by presenting the additional blocks required to regulate voltage and frequency at run-time using state-of-the-art devices and leveraging the availability of the PMBus in off-the-shelf FPGA boards. In addition, we investigated the run-time power and performance scaling in 28nm Xilinx Virtex-7 devices and compared it with the work in [3] based on 65nm FPGAs. To do this, we implemented different test systems, with a varying number of test modules to consume different portions of the device, and evaluated the run-time power and performance scaling of the systems. The results reveal that the available voltage and frequency margins create a large number of performance and energy states with scaling possible at run-time with low overheads. Power savings of up to 64.98% are possible maintaining the original performance at a lower voltage. In this paper we investigate and compare software and hardware methods to control power and monitor energy in Xilinx ZYNQ-based FPGA boards. The methods are also applicable to other FPGA boards that use a programmable power supply accessible through a PMBus interface.

The rest of the paper is structured as follows. Section 2 presents different power control and monitoring methods in the hybrid CPU+FPGA boards. Section 3 uses these methods with a real design and explores the different trade-off in terms of performance, overheads, and accuracy. Finally, section 4 presents the final conclusions and future work.

II. VOLTAGE SCALING METHODS

A key point in this research is that many modern FPGA boards include Power Management Bus (PMBus) Controllers. The PMBus is an open standard power management protocol that facilitates the communication with power converters and other devices in a power system [6]. This technology means that software or hardware running in the device have access to a controllable power supply. This is the case with the latest evaluation kits (such as the KC705, VC707, ZC702, and ZC706) for Xilinx series 7 FPGAs that use the Texas Instrument (TI) UCD92xx PMBus controller. The TI UCD92xx series [7] is a family of digital power controller which supports a wide range of commands that allow an external host to configure, control, and monitor the controller

We acknowledge with gratitude EPSRC for their support with research grant ENPOWER (Elastic and Non-stationary POWER)

through an I2C electrical interface using the PMBus command protocol.

There are two possible methods to communicate with the PMBus controller in these boards [8]. The first method employs the Fusion Digital Power Designer software package provided by TI [9]. This software package has several tools that are able to communicate with the UCD92xx series of controllers from a Windows-based host computer. This software package requires the use of a USB Interface Adapter EVM [10] to connect the PMBus (I2C) interface of the UCD92xx controller and the USB port in the host computer. The second method consists in using the PMBus (I2C) interface which is available on the boards. This is a more complex method since it requires creating custom code on the device to read and write properly formatted PMBus and UCD92xx commands. TI UCD92xx PMBus Command Reference Manual and the industry standard PMBus Specification for UCD92xx command codes, data formatting, and PMBus protocol are available on [11, 12], respectively to guide the designer in this task. In this work we focus on the second method because we need to access the PMBus interface internally to scale voltage dynamically and autonomously.

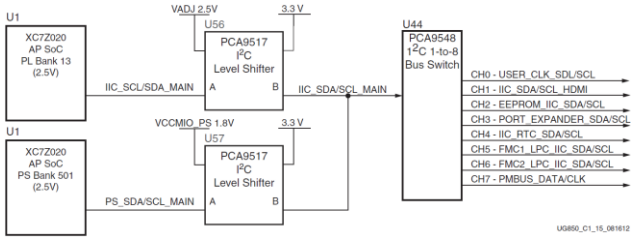

Fig. 1. The ZYNQ I2C Bus topology (Adapted from [13])

We have selected the Xilinx ZC702 (ZYNQ), which is a hybrid CPU+FPGA evaluation board, in this work. Fig.1 displays the I2C Bus topology on this board which has been adapted from [13]. As can be seen in Fig.1, the ZC702 board implements a single I2C port in the XC7Z020 ZYNQ SoC accessible through pins IIC_SDA_MAIN and IIC_SDA_SCL. The PMBus is connected to an I2C 1-to-8 Bus switch external to the device in which others I2C peripherals are also connected. There are two alternatives to communicate to the PMBus internally as both the PL and the PS banks have access to the PMBus data and clock signals. In the following sections we discuss the features of these two alternatives.

A. Programmable Logic Method (PLM)

This method is a hardware method that takes advantage of the DVS IP core presented by the authors in [5] which is implemented in the Programmable Logic (PL) part. Fig.2 shows the Dynamic Voltage Scaling (DVS) unit architecture. The DVS unit has three main components which are a MicroBlaze processor (MB); a register file implemented using a Dual-Port RAM (DPRAM) and an I2C IP core. These components are connected to a local AXI bus.

The DVS unit has full configuration and monitoring capabilities of the power rails connected to the PMBus. The DPRAM is used to receive the commands from the system processors (i.e. Cortex A9 processors). The commands control and record power, current, and voltage values. The MB is responsible for the execution of the commands, communicating with the PMBus via the I2C IP core and writing the results to the DPRAM. The need for a MB processor is mainly due to the relatively complexity of I2C communications that means that a state machine implementation will be complex to design and maintain for different boards with slight PMBus implementation differences.

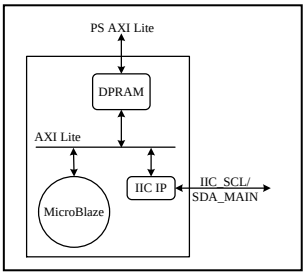

Fig. 2. The DVS IP Core

In the ZC702 board, the initialization code executed by the MB must set the 1-to-8 switch to point to the PMBus channel before communication with the voltage regulators is possible.

The initialization, configuration and monitoring code is written in C and compiled into an .elf file using the standard MB compiler. Note that .elf file is not an output of a C compiler but an input to describe memory mapping/layout for linker. In this board, JTAG has not access to the MB processor and therefore it is not possible to download the compiled C software (.elf file) directly. The .elf is made part of the bitstream as a firmware and it is automatically stored in the program memory when the device is configured. The DVS core is controlled with commands which are issued by the Cortex A9 processor present in the PS part [5].

In this system voltages can be scaled from s from 650mV to 1V and from 1V down to 650 mV. The IP Core is designed to maintain the voltage in this range to avoid damaging or cutting off the power supply of the board. This means that the IP core will automatically reject commands that indicate a voltage value out of these ranges.

TABLE I. COMPLEXITY OF THE DVS UNIT COMPONENTS

Resource	FF	Utilization	LUT	Utilization
Microblaze processor	972	0.9%	631	1.2%
I2C Controller	343	0.3%	468	0.9%

Table I shows the complexity of the DVS unit components after implementation in the device. As can be seen in this table, the unit is area efficient and it only consumes a small fraction of the available resources in the PL part. Nevertheless there is an overhead that could affect measurements done in the PL part directly. For this reason this method is more suitable for monitoring the PS part accurately as the voltage scaling and monitoring core does not consume power in the power supply of the PS part.

B. *Processing System Method (PSM)*

This method is a software method in which the Cortex A9 Processing Unit (PU) is responsible for monitoring and voltage scaling using executing routines implemented in C code. This code uses similar commands format to the PLM and gives the PU full configuration and monitoring capabilities of the power rails connected to the PMBus. Although this method does not have any area overhead, it cannot monitor the PS part accurately as the voltage scaling and power monitoring software consumes itself power. In the next sections we investigate the accuracy and overheads of each of these methods.

III. Power Scaling Analysis

As a test system, we have created a Linux based video processing system. Fig.3 shows the architecture of our test platform using our DVS IP core and LiquidMotion Processor (LMP) [14], which is an open source reconfigurable Application Specific Instruction Set Processor (ASIP) designed to execute user-defined block-matching motion estimation algorithms, in the PL part. This system runs a video processing algorithm, which takes advantage of openCV libraries for video input in the PS part and passes a reference and current frames to the PL part to calculate the motion vectors by the LMP. We have scaled the PL voltage and compared the power monitoring results to the power at nominal voltage with both methods described in the previous section. In this work we are focusing on VCCint and VCCPint which are supplying the PS and PL cores respectively and we do not consider other voltage consumers.

We have measured that the minimum safe voltage in this board for the test platform is 750 mV. The term of minimum safe voltage means the lowest voltage that the test platform is operating without any error or faults.

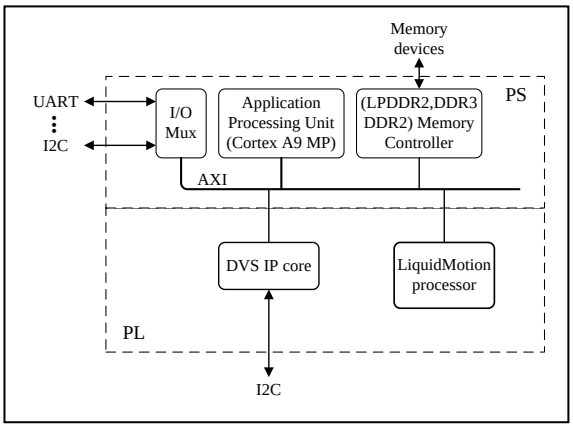

Fig. 3. The test platform architecture

A. *Area*

Table II shows the complexity of the test platform. This table shows that the test platform including the LMP processor occupies approximately a quarter and half of the available LUT and Slices, respectively.

TABLE II. The Test System Complexity

Resource	Registers	LUT	Slice	BRAM-36E1
System	10682	14642	6375	42
Available	106400	53200	13300	280
Utilisation	10%	27.50%	47.90%	15%

B. *Timing*

Table III displays the required time for the monitoring and voltage scaling. The times in this table are calculated since the task command is issued until the task has been finished. As can be seen in Table III, monitoring a parameter (i.e. voltage, current and or power) and scaling the PL voltage tasks is faster with PSM due to communication time between the system processors (i.e. Cortex A9 processors) and MB. On the other hand, if we take into account the large differences in frequency the PLM is overall more effective (PL_{freq}=100 MHz, PS_{freq}=666 MHz). Voltage scaling takes considerably longer to complete compared to monitoring and this is expected since writing a new voltage involves a sequence of commands to control the power rails.

TABLE III. Monitoring and Voltage Scaling Timing

Timing	Monitoring 1 parameter	Voltage scaling
PLM	3.9 ms	50.3 ms
PSM	2 ms	6.3 ms

C. *Power and Voltage analysis*

As a test case to investigate the overheads of each method we have activated the test system, scaled the PL voltage (VCCint) from 1 to 0.75 V and monitored the PL and PS powers with both PLM and PSM.

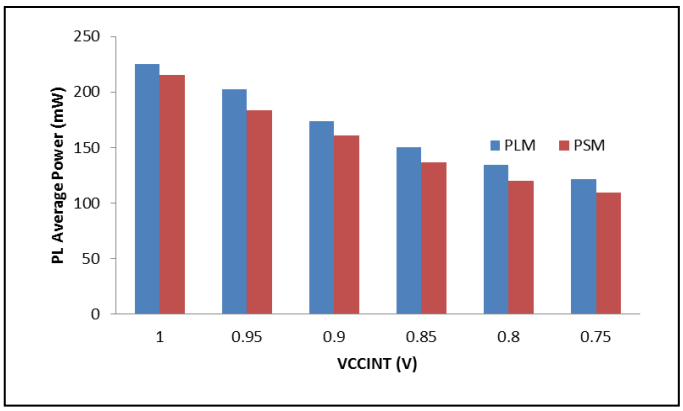

Fig. 4. Voltage scaling and monitoring the PL power

Fig.4 shows the average power when the PL and PS scale the voltage and monitor the power in the PL part. As can be seen in this figure, voltage scaling from nominal (1V) to 0.75 V helps to save the PL power up to 49%. In addition, PLM introduces an overhead in PL power of around 4.9-11.6%.

Fig.5 displays the average power when the PL and PS scale the PL voltage and monitor the power in the PS part. As can be seen in this figure, the PSM introduces a higher overhead in the PS power as it consumes 16-20% more power compared to the PLM.

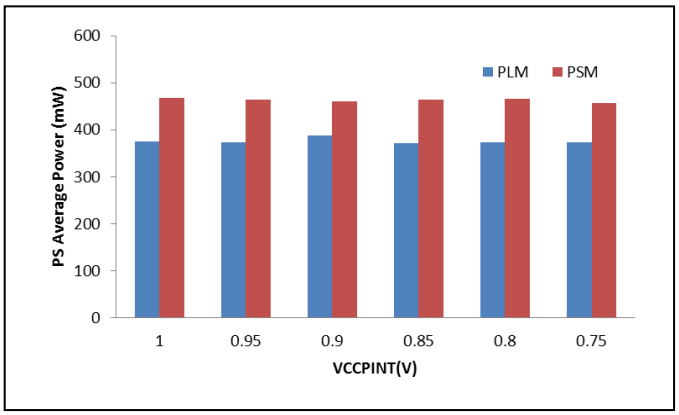

Fig. 5. Voltage scaling and monitoring the PS power

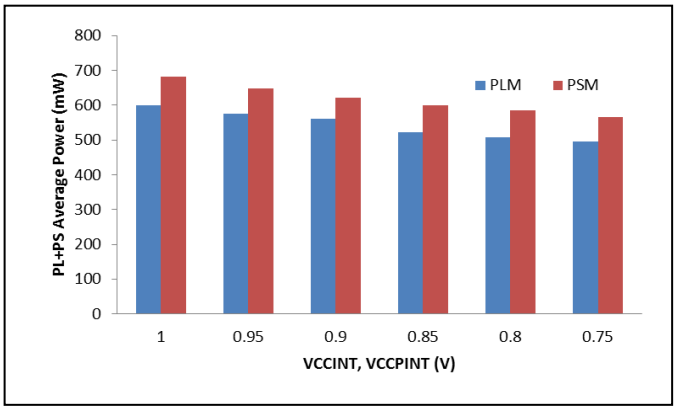

Fig. 6. Voltage scaling and monitoring the PL+PS power

Fig.6 shows the average power when the PL and PS scale the PL voltage and monitor the power in the PS+PL parts. As can be seen in this figure, the PLM hardware method is overall 12% more efficient compared to the PSM software method. In addition, scaling voltage from nominal to 0.75 V and using the PLM save 27.58% power compared to the PS method at nominal voltage.

Overall, the PSM or PLM methods should be considered in tandem since each one is better suited for some tasks depending if scaling or monitoring is taking placed and the required overall performance. PSM is faster thanks to its faster clock speed, does not have an area overhead and measures the PL accurately. PLM method consumes less power and measures the PS accurately. Using partial reconfiguration techniques a dynamic power and monitoring strategy could be implemented in which PLM and PSM are enabled selectively and temporally.

IV. Conclusion

Our previous work in [3] has investigated the capability of standard FPGA devices to operate out of their nominal ranges with over and under scaling of frequency and voltage. The work presented in [3] was based on Virtex-5 devices fabricated using a 65 nm process. Further investigations in [4, 5] have revealed these margins are still present in modern 28 nm FPGAs that have the same nominal voltage of 1 V. The results showed that the margins available make these chips a good platform for energy proportional computing.

In this paper, we have investigated accurate power control and monitoring in Xilinx ZYNQ-based FPGA boards. To do this, we have introduced two different methods for monitoring and voltage scaling use the PS and PL parts to optimize the speed and accuracy. We have examined both methods in a real application test system. The results reveal that voltage scaling from nominal (1V) to 0.75 V reduces PL power up to 49% for the considered test cases. Selecting a voltage scaling and monitoring method is dependent on different design strategies defined by the required control and monitoring speeds, accuracy of measurement, power consumption and area overhead.

Future work involves further validation of the proposed techniques in commercial application and the introduction of dynamic technique that can use partial reconfiguration to selectively activate the PLM method to replace the PLS method when it is required by the overall power management strategy.

Acknowledgment

We acknowledge with gratitude EPSRC for their support with research grant ENPOWER (Elastic and Non-stationary POWER).

References

[1] Kuon, I.; Rose, J., “Measuring the Gap Between FPGAs and ASICs,” Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on , vol.26, no.2, pp.203,215, Feb. 2007.

[2] Available:http://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf. [Accessed 20 May 2014].

[3] Nunez-Yanez, J., “Adaptive Voltage Scaling with in-situ Detectors in Commercial FPGAs,” Computers, IEEE Transactions on , vol.PP, no.99, pp.1,1, 0, doi: 10.1109/TC.2013.73.

[4] Nunez-Yanez, J.L.; Beldachi, A., “Run-time Power and Performance Scaling for Reconfigurable Computing”, in 8th HiPEAC Workshop on Reconfigurable Computing, Vienna, Austria, 2014.

[5] Beldachi, Arash Farhadi; Nunez-Yanez, Jose L., “Run-time power and performance scaling in 28nm FPGAs”, IET Computers & Digital Techniques, 2014.

[6] Available: http://pmbus.org/index.php. [Accessed 20 May 2014].

[7] Available: http://www.ti.com/lit/ug/sluu490/sluu490.pdf. [Accessed 20 May 2014].

[8] Available: http://www.xilinx.com/support/answers/37561.html. [Accessed 20 May 2014].

[9] Available:http://focus.ti.com/docs/toolsw/folders/print/fusion_digital_power_designer.html. [Accessed 20 May 2014].

[10] Available: http://focus.ti.com/docs/toolsw/folders/print/usb-to-gpio.html. [Accessed 20 May 2014].

[11] Available: http://focus.ti.com/lit/ug/sluu337/sluu337.pdf. [Accessed 20 May 2014].

[12] Available: http://pmbus.org/specs.html. [Accessed 20 May 2014].

[13] Available:http://www.xilinx.com/support/documentation/boards_and_kits/zc702_zvik/ug850-zc702-eval-bd.pdf. [Accessed 20 May 2014].

[14] Available: http://opencores.org/project,motion_estimation_processor. [Accessed 20 May 2014].