PicoBlaze using Xilinx ISE: Implemented on the Digilent Nexys Board

EEL5707 Final Project Repot, December 6th, 2010
F. Fleming, Student Member IEEE

Abstract –This document outlines the insight the author has gained from the final project of EEL 5707, "ASIC System Design I", at Florida State University under professor Dr. Uwe Meyer-Baese. The work investigates embedding the "PicoBlaze™ microprocessor onto an academic friendly development board, the Digilent® Nexys, which contains a Xilinx XC3S200 field programmable gate array (FPGA). While various design tools exist for creating one's own embedded microprocessor (EμP) (ex. LISA [1]), this work utilizes the PicoBlaze® which is pre-existing and widely supported. To showcase the use of EμPs, the author will compile a sum of products (SOP) "C"-code program looping 15 times onto the EμP. This would otherwise be time very time-consuming and extensive to do in an assembly language without the multiply instruction. This document is outlined as follows: Section I introduces the topic, further details the problem, and provides the Gantt chart of the work outline; Section II covers the necessary background theory while providing a literary review of the associated hardware and software required; section III details the development process for embedding the SOP program onto the Nexys board; Section IV defines the limitations of the designs, outlines future work, and the intended research applications and objectives; and finally Section V concludes.

I. Introduction

As technology has progressed, application specific integrated circuits (ASIC) now cover a much broader range of circuit types and topologies than traditionally defined. Initially, ASIC circuits were considered "hand designed cell based circuits" (encompassing both fully custom and standard cell/gate array semi-custom designs), thus giving rise to the title "application specific." With recent advances in field programmable gate arrays (FPGA) ASIC designs have began incorporating custom-programmable functionalities via programmable logic, wired design or even memory. It is easy to see when compared to transistor-transistor logic (TTL) design that both traditional and custom programmable ASIC reduce cost, size, weight and power; and improve throughput and security [1]. While traditional cell-based ASIC designs still have their merits over modern ASIC and FPGA designs: (1) lower cost for high quantity yields and (2) typically contain up to ten times more gates per same die size (this is intuitive as they are application specific designs), the growth FPGA technologies and the flexibility of programmable circuits create quite the design tradeoff. The reprogrammable nature allows for a reduction in development time and less debugging, as circuits are now programmable. Embedded microprocessors (EμP) are a useful example which exploits the convenience and power of programmable circuits. By sacrificing some of the available FPGA's chip space, a single design may benefit from the parallel nature of FPGA's, the

efficiency of application specific designs, and the convenience and support of using a µP as a digital signal processor (DSP). Note that as FPGAs have silicon space restrictions, the EµP instruction set must be reduced to fit the specific circuit's application, thus the larger DSPs that power desktops PCs are not a plausible EµP.

A very small, space efficient EµP, the "PicoBlaze", consumes only 96 slices of silicon, contains 16 general purpose registers, supports both VHDL and Verilog languages, all while still delivering 87 MHz (43-66 millions of instructions per second) of processing speed [2]. Furthermore, the design is open-source and intended for academic uses on the Spartan-3, Virtex-II, and Virtex-II PRO device families. Thus, this work intends to embed the PicoBlaze onto the XC3S200 FPGA of the Digilent Nexys board. While the details of the board and FPGA are not of concern, the interested reader is referred to [3] and [4], for the board and FPGA's documentation, respectively. It is informative to point out that the multiply instruction code is not defined for the PicoBlaze, thus to further exemplify the convenience of embedding microprocessors in ASICs the author will use a "C" compiler to create the assembly code of a sum of products program.

Figure 1 shows the projects scheduling over the seven allotted weeks to completion. Note that green indicates "on-time", red is "not completed", and purple is a later completion than planned for.

Tasks		10/18 – 10/24	10/25 – 10/31	11/1 – 11/7	11/8 – 11/14	11/15 – 11/21	11/22 – 11/28	11/30 – 12/6
Study hardware/ software	Nexys I Board	X						
	XC3S200 FPGA	X						
	KCPSM3	X						
Setup development environment	ISE 12.3	X						
	Adept utility	-	X					
Programming*	Compile simple C-program			1/2	X			
	Peripheral control				3/4	X		
	Embed PicoBlaze					1/2	X	
	SOP example						1/2	3/4
	ADC				-	-		
	Develop metric for speed					-	-	
Deliverables	Presentation							X
	Final report							pending

Figure 1: Gannt Chart

II. Background Information

While it is assumed that the reader is familiar with sum-of-products it requires minimal theory to explain, thus is included for completion. As the name implies the product of two given functions, x(k) and f(k), are taken and them summed over various values of k. The value of k for this work is 15. Equation (1) clarifies this further.

$$\sum_{k=0}^{k=15} x(k) * f(k) \tag{1}$$

Aside from this minimal theory, the author was also unfamiliar with any of the necessary development tools required to program the Xilinx FPGA. Thus, a detailed literature and hardware search uncovered the specifications of the discontinued Digilent Nexys I board [3], the Digilent Adpet software used to transfer the compiled VHDL .bit file to the board through mini USB [5], the Xilinx IDE Design Suite needed for code synthesis, and the use of either the ISim Xilinx simulator or the third-party ModelSim MXIII Starter [6]. The necessary PicoBlaze files and C-Compiler (PCC) were given via the course website [1]. An extensive run-down of all of these data sheets is not included; the interested reader is deferred to these references.

III. Research and Development

Prior to embedding the EμP onto the Nexys board, various check-sums were required to ensure the project was developing in a concise, logical manner. Table I details the development process objectives for embedding the SOP program onto the Nexys board.

Table 1 - Developmental Agenda

1	Setup development environment
2	Completed Nexys & Adept software tutorials
3	Achieved peripheral control – Required Xilinkx ISE & Digilent Adept – Switch, button, LED, oscillator, and 7-SEG controlled
4	Completed Picoblaze (KCPSM3) tutorial – Including interrupts (not really much to see synthesized)
5	Use PCC for a sum of products (SoP) example – Either use peripheral or memory to show register values have correct value after SoP

The first two items including the setup of the development environment is covered nicely in the cited reference manuals through tutorials thus is excluded. The third is of great importance as peripheral control dictates whether or not the author has control of the board, allows for feedback during the programs runtime, and ensures the Adept software correctly transferred the code to the board.

To test the peripherals a simple program was writing that mapped the input to output as shown in Table II. Note that the switches, LEDs, clock oscillator, and 7-segment displays were tested.

Table 2 - Peripheral I/O Logical Mapping

Input	Output
Switches 0-3	LED 0-3
DIP Buttons 0-2	LED 4-6
Switch 7	7-segment display 0
If (Button 3 = '1') then (LED (7) <= CLK;)	LED 7

Figure 2 shows this code running on the board as Button 3 and Button 1 are held down, as switches 0, 1, and 3 are toggled. Note that the clock is working as desired (as shown by the "dim" light indicated by the arrow). Furthermore the seven segments are only being controlled to output the top "A" output for each 4 displays. The underlying code for this test is given in the attached .zip (example2IO).

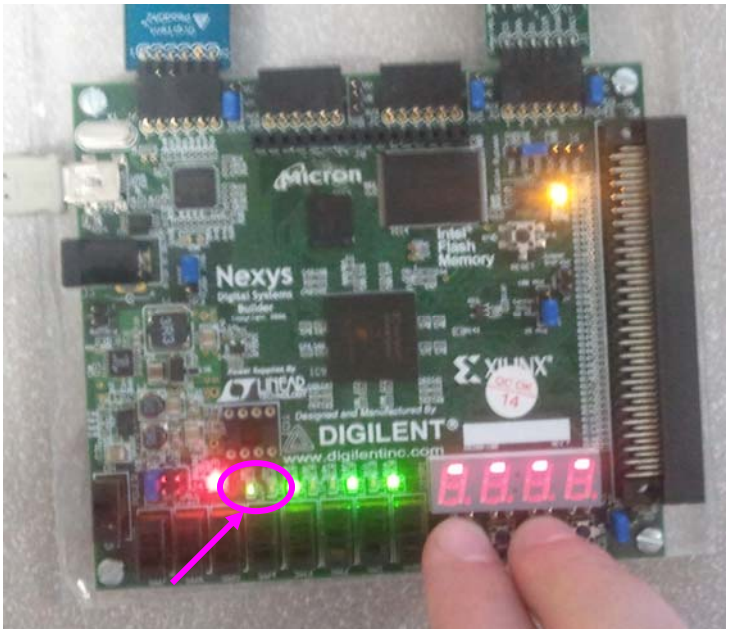

Figure 2: Peripheral Operation

Next, the PicoBlaze tutorials were completed to enlighten the author with instructional knowledge of compiling the EμP. The step has no real deliverable, but note that even interrupts were tested (just without any feedback/acknowledgement from the device). The summary of the procedure to embed C code onto the board is shown in Table III.

Table 3 - C-code to embedded EμP

1	pccomp -c prog.c
2	KCPSM3 prog.asm > out.txt
3	Compile prog.vhd via ISE
4	Load prog.bit via Adept

Finally, the SoP was ready to be coded and transferred to the board. First two functions were arbitrarily defined to be *k* and *k+2*, respectively. Equation 2 fills in the variables for the SOP equation. Next, a MATLAB script was ran to verify both the solution and as a pre-curser to writing the C-Code, both of which can be found in the attached .zip file (example3sop). The SOP code was successfully loaded onto the development board, but without any I/O feedback. The difficulty faced was the necessary C-code to control the peripheral. This was necessary since the user has no knowledge of the EμP memory state.

$$\sum_{k=0}^{k=15}(k+2)*k=\sum_{k=0}^{k=15}(k^2+2k)=1480 \qquad (2)$$

IV. Limitations, Future Developments, and Research Applications

While the author is confident in the successful implementation of the SOP code (due to its simplicity), it is typically bad style to blindly accept such results without simulation or final verification. Therefore, the main limitation of this project is the lack of feedback from the Nexys board that verifies the value 1480 is stored somewhere in memory. Furthermore, a simulation using either of the two simulation tools described in Section II would further strengthen the development process.

While, further developments are out of the scope of this project, it is interesting the mention possible future works regarding embedding the PicoBlaze. Primarily, one could use the analog to digital converters attached via the 4 6-pin headers, as shown in Figure 3, in conjunction with the PicoBlaze. Shown is a signal generator outputting an analog sinusoid of 60 Hz 3.3V signal into one of the ADC ports. A PicoBlaze EμP could be instantiated and "C"-coded to properly sample the analog waveform and apply one of many digital protocols to turn the analog signal into an array of square waves (roughly translating into a digital signal).

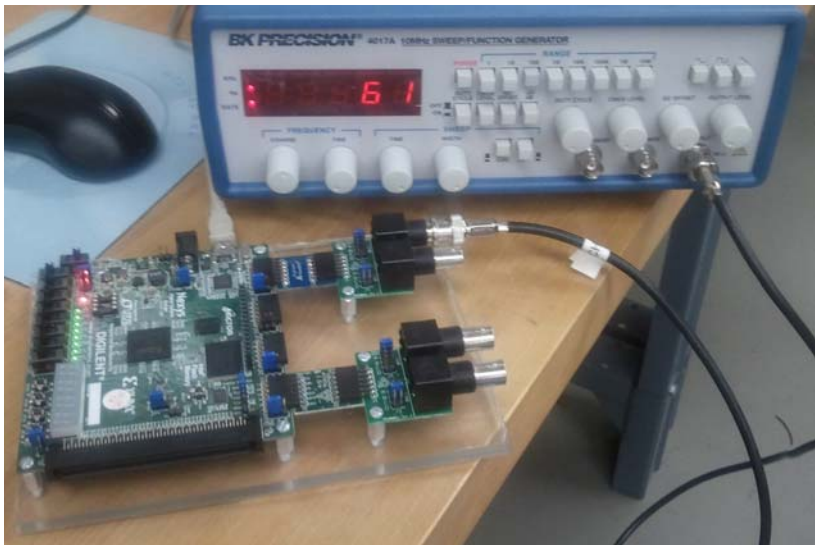

Figure 3: Nexys ADC Modules

Prior to completing such a test one needs to finish the necessary output verification of the SOP code. This would not only allow the satisfaction of results but also allow peripheral access through C-code (which is highly advocated if one was the write an ADC subroutine). A useful demonstration of this ADC could be recreating the familiar, in-class homework 4; where students use simulation to read in a sinusoidal and output the magnitude for power quality applications. Also available are various Universal Asynchronous Receiver/Transmitter (UART) tutorials for the PicoBlaze, which would provide a nice starting point for such an ADC application.

A DATA2MEM application is available [1] that allows the compiled VHDL .bit file to be reconfigured without restarting the steps outlined in table 3. Developing a nice demo to showcase the usefulness, or instantiating various interrupts beyond what is outlined in the product tutorials would prove worthy projects.

Finally, it is important to point out how the author intends to use the knowledge obtained from this project to further his research in electric drives, controls, and power systems. While the novel research in such a field does not center on an FPGA or ASIC, they prove as superior tools over commercial electric drive controllers as the researcher has more control of modulation strategies and the control algorithms implemented. Thus an FPGA will be used to take various input (voltage, current, and encoder) signals, convert them to a digital protocol, and use them in a novel control algorithm to study electric drive non-linearites in renewable system applications. Ultimately the FPGA will output digital signals to IGBT gate driver cards to trigger switching events in an electric converter; separately the FPGA provide DC voltage references to a DC motor controller. Furthermore, within the VHDL program could be a nonlinear model (ex. machine, electronics devices, etc) whose terminal characteristics would provide the control algorithm a reference. In doing this it is necessary to investigate the methods to solve the underlying equations. Questions then arise if a softcore processor could provide a viable non real-time solution? Thus, the knowledge gained throughout this class of using both Altera and Xilinx products as well as EμP will greatly help answer such questions.

V. Conclusion

In conclusion the author has demonstrated the necessary skills to fully research a given topic given only hardware and design goal. Furthermore, copious knowledge pertaining to Xilinx development tools, the PicoBlaze (KCPSM3) EμP, C-to-assembly compilers, USB .bit transfer clients, and project scheduling was obtained. An adequate review of conventional cell-based ASIC vs. reprogrammable ASIC was given, the developmental process shown, and the limitations and applications of the technology discussed.

Acknowledgements

Fletcher Fleming would like to thank Dr. Uwe Meyer-Baese for supplying the required hardware (Nexys Board and cable) to carry out the project. Furthermore, extensive use of his class notes and book "Digital Signal Processing with Field Programmable Gate Arrays" assisted greatly.

References

[1] U. Meyer-Baese, *ASIC System Design I* class notes, Florida State University, Fall 2010

[2] Ken Chapman, "*KCPSM3 (PicoBlaze) 8-bit Micro Controller for Spartan-3, Virtex-II and Virtex-IIPRO*", Xilinx Ltd., October 2003

[3] Digilent Nexys® documentation, "https://www.digilentinc.com/Products/Detail.cfm?NavPath=2,398,824&Prod=NEXYS"

[4] Xilink XC3S200 Datasheet, "http://www.google.com/url?sa=t&source=web&cd=3&sqi=2&ved=0CCUQFjAC&url=http%3A%2F%2Fwww.xilinx.com%2Fsupport%2Fdocumentation%2Fdata_sheets%2Fds099.pdf&ei=El38TNjuJ468sQPX7YD3DQ&usg=AFQjCNGNPW8rXbFgsa6jCR2pSmuFJy6maQ&sig2=OXYsgid5zhqjpM8xGWMPbg"

[5] Digilent Adept Utility, "https://www.digilentinc.com/Products/Detail.cfm?NavPath=2,66,828&Prod=ADEPT2"

[6] Xilinx Free Webpack Utilities, http://www.xilinx.com/tools/webpack.htm,

[7] U. Meyer-Baese, *Digital Signal Processing with Field Programmable Gate Arrays 3rd edition*, Springer 2007