Teaching Computer Architecture Using VHDL Simulation and FPGA Prototyping

Dr. Ronald J. Hayne, The Citadel

Ronald J. Hayne is a Professor in the Department of Electrical and Computer Engineering at The Citadel. He received his B.S. in Computer Science from the United States Military Academy, his M.S. in Electrical Engineering from the University of Arizona, and his Ph.D. in Electrical Engineering from the University of Virginia. Dr. Hayne's professional areas of interest include digital systems design and hardware description languages. He is a retired Army Colonel with experience in academics and Defense laboratories.

Teaching Computer Architecture using VHDL Simulation and FPGA Prototyping

Abstract

An Instructional Processor design example has been expanded to facilitate teaching of a Computer Architecture course. The system is modelled in VHDL and simulated using Xilinx design tools to demonstrate operation of the processor. A basic microcontroller is created by adding memory-mapped input/output (I/O). The system is implemented in hardware on a field programmable gate array (FPGA). The processor can then be interfaced with peripheral devices to demonstrate functional applications.

A key component of the Computer Architecture course is a student design project. The Instructional Processor provides the base design, which can be modified to adapt to a new set of specifications. Students must modify the appropriate processor components and integrate them into the data path. The control unit must also be redesigned to accommodate the new instructions. A sample program is then tested via simulation of the updated VHDL model.

The base processor is expanded by adding a serial communication interface, designed using a UART (universal asynchronous receiver transmitter). Next, a programmable timer and interrupt system are added to the processor architecture. The enhanced FPGA microcontroller is tested using a design example which gives students an in-depth look at both the internal details and external interfacing of a hardware-software system.

The expanded Instructional Processor has been successfully used to teach Computer Architecture using VHDL modelling and simulation, combined with FPGA implementation and testing. The design project serves as a good assessment of the students' understanding of key design concepts via their use of industry-standard tools. Feedback has been very positive that the course illustrates fundamental design concepts reinforced by actual functioning hardware.

Introduction

Teaching Computer Architecture involves use of many components including counters, registers, multiplexers, arithmetic logic units (ALUs), and memory. The design of a computer processor combines these building blocks into an integrated digital system. An Instructional Processor, which was developed as a design example in an Advanced Digital Systems course [1], [2], has been expanded to facilitate teaching of a Computer Architecture course. The system is modelled in VHDL and can be simulated using Xilinx design tools to demonstrate operation of the processor. A basic microcontroller is then created by adding memory-mapped I/O. The system can be synthesized and implemented in hardware on an FPGA. The processor can then be interfaced with multiple peripheral devices to demonstrate a variety of applications.

Several Computer Architecture courses exist which use hardware description language models and simulation, combined with FPGA implementation. Many of them are based on the open-source RISC-V system-on-chip (SoC) [3]. RVfpga is a freely available course which uses

Verilog, C, and assembly language to explore and modify the microarchitecture [4]. LeaRnV is a course whose objective is to train students in hardware-software co-design of integrated systems [5]. Alternatively ARM provides an Introduction to Computer Architecture kit, based on its commercial processor [6]. All of these options involve very complex architectures and higher-level system views. The goal of this course was to use a much simpler processor design with the ability to explore the in-depth details of the hardware-software system.

Instruction Set Architecture

The Instructional Processor has been designed to illustrate multiple operations and basic addressing modes. It is based on a three-bus organization of a 16-bit data path with a four-word register file (REGS). Key registers include the program counter (PC), instruction register (IR), memory data register (MDR), and memory address register (MAR). Other components consist of the ALU, subroutine STACK, and a 4K word by 16-bit MEMORY. The complete data path and memory map are shown in Figure 1.

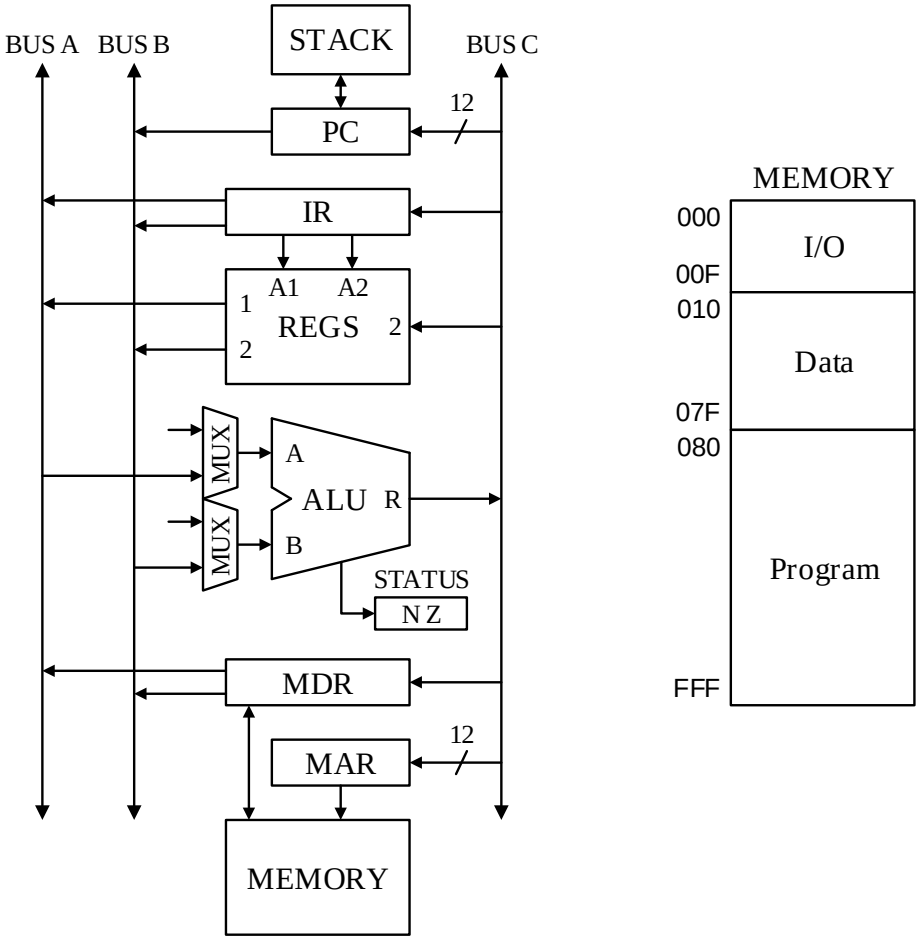


Figure 1. Data Path and Memory Map for the Instructional Processor.

The course starts with an introduction/review of VHDL by modelling key components of the processor architecture, the program counter and the multi-port register file. Students gain insights into VHDL design techniques and familiarity with the Xilinx design and simulation tools [7]. Next, more advanced sequential systems are designed via a control unit directing a dedicated data path. This technique is critical for complexity management and serves as the basis for the processor architecture. Throughout the VHDL lessons, the importance of specific modelling constructs and their relation to synthesized hardware are emphasized. This ensures that after successful simulation, the processor design will be able to be implemented and tested in FPGA hardware.

The data path of the Instructional Processor combines the previously developed VHDL components with an ALU, registers, buses, and memory. The instruction set architecture is designed with basic operations and addressing modes with sufficient complexity to demonstrate fundamental programming concepts such as data transfer, counting, indexing, and looping. Programs can be written in assembly language and converted to machine language manually or with the provided assembler [2].

The VHDL model interconnects the data path components with their appropriate control signals. The control unit, shown in Figure 2, generates the necessary sequence of commands to fetch, decode, and execute the machine language program. The system is then simulated with a sample test program to verify proper operation via register values, ALU results, control signals, and memory contents.

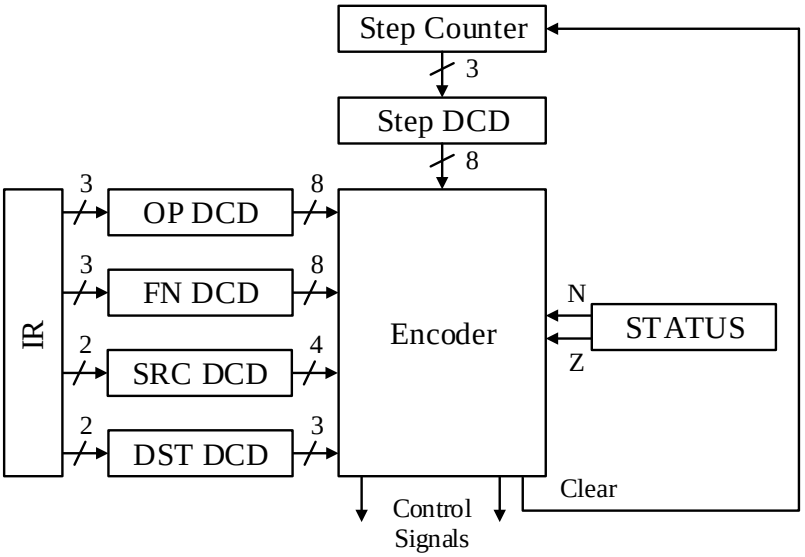


Figure 2. Control Unit Organization for the Instructional Processor.

The VHDL model provided to the students does not include the subroutine stack. Instead, design and integration of this system component is left as a homework exercise. Details of the STACK and internal Stack Pointer (SP) are shown in Figure 3. Subroutine call and return instructions use this stack to save and retrieve the 12-bit program counter addresses.

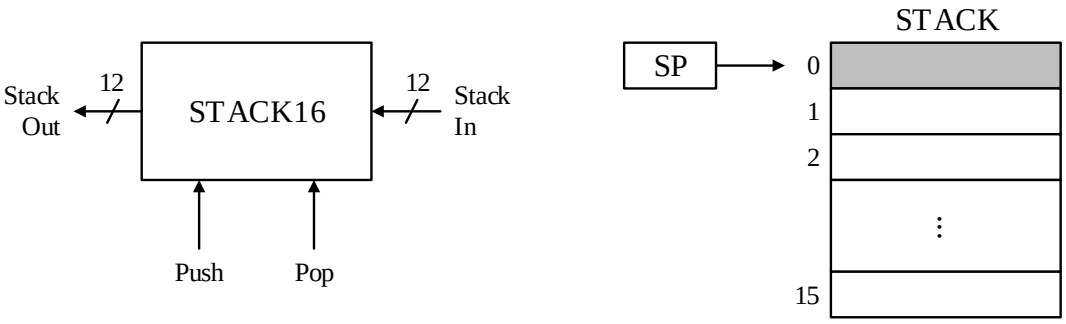
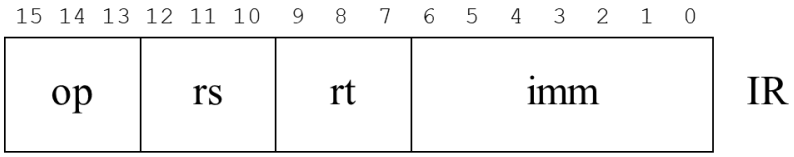


Figure 3. STACK16: 16 x 12-bit Subroutine Stack.

Design Project

A key component of the Computer Architecture course is a student design project. The Instructional Processor provides the base design, which can be modified to adapt to a new set of specifications. The register file is expanded to 8 words with 3 addresses, which allows implementation of a load and store architecture. The new register and immediate formats are similar to those used by the MIPS [8] and RISC-V [3] processors. A sample of the immediate instruction format for the design project is shown in Figure 4.



op	Assembly Language	Register Transfer Notation
001	`addi rt,rs,imm`	rt <= rs + imm
010	`andi rt,rs,imm`	rt <= rs and imm
011	`lw rt,imm(rs)`	rt <= MEM(imm + rs)
100	`sw rt,imm(rs)`	MEM(imm + rs) <= rt
101	`bra imm`	PC <= PC + imm
110	`bnz imm`	if Z = 0 then PC <= PC + imm
111		

Figure 4. Immediate Instruction Format for Design Project.

Students must modify the appropriate processor components and integrate them into the data path. The control unit also needs to be redesigned to accommodate the new instructions. The sample test program, which was used to test the base Instructional Processor, adds all the elements of a data array (X) to create a SUM. This program must be converted to the new assembly language (in red), as shown in Figure 5. It is then encoded in machine code, and tested via simulation of the updated VHDL model.

```
.data
SUM
N 3
X 7, -8, 10
.program
START:  lw   r1,N(r0)
        addi r2,r0,X
        MOVE 0,R0
LOOP:   ADD  [R2],R0
        ADD  1,R2
        ADD  -1,R1
        BNZ  LOOP
        MOVE R0,[SUM]
STOP:   BRA  STOP
```

Figure 5. Assembly Language Test Program.

Microcontroller Expansion

While the students are working on their design projects, the class continues with expansion of the base processor design. A basic microcontroller is created by adding memory-mapped I/O and the system is synthesized to an FPGA. A pulse-width-modulation (PWM) motor control application is used to test the resulting hardware. Next, a serial communication interface is designed using a UART [9].

During the final weeks of the course, a programmable timer and interrupt system are added to the processor architecture [10]. Details of the components and interfaces are shown in Figure 6.

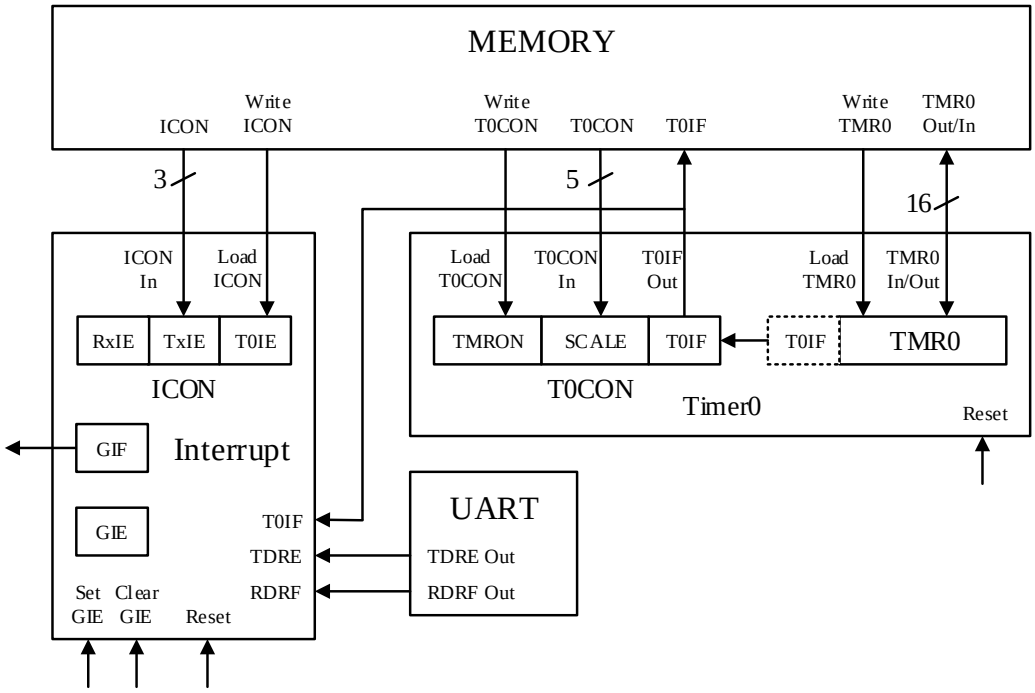


Figure 6. Interrupt System with Timer, UART, and Memory Interface.

The enhanced microcontroller is then tested on a BASYS 3 FPGA board [11], using a time-multiplexed seven-segment display and a serial RFID (radio frequency identification) card reader [12], as shown in Figure 7. The design example gives students an in-depth look at both the internal details and external interfacing of a hardware-software system.

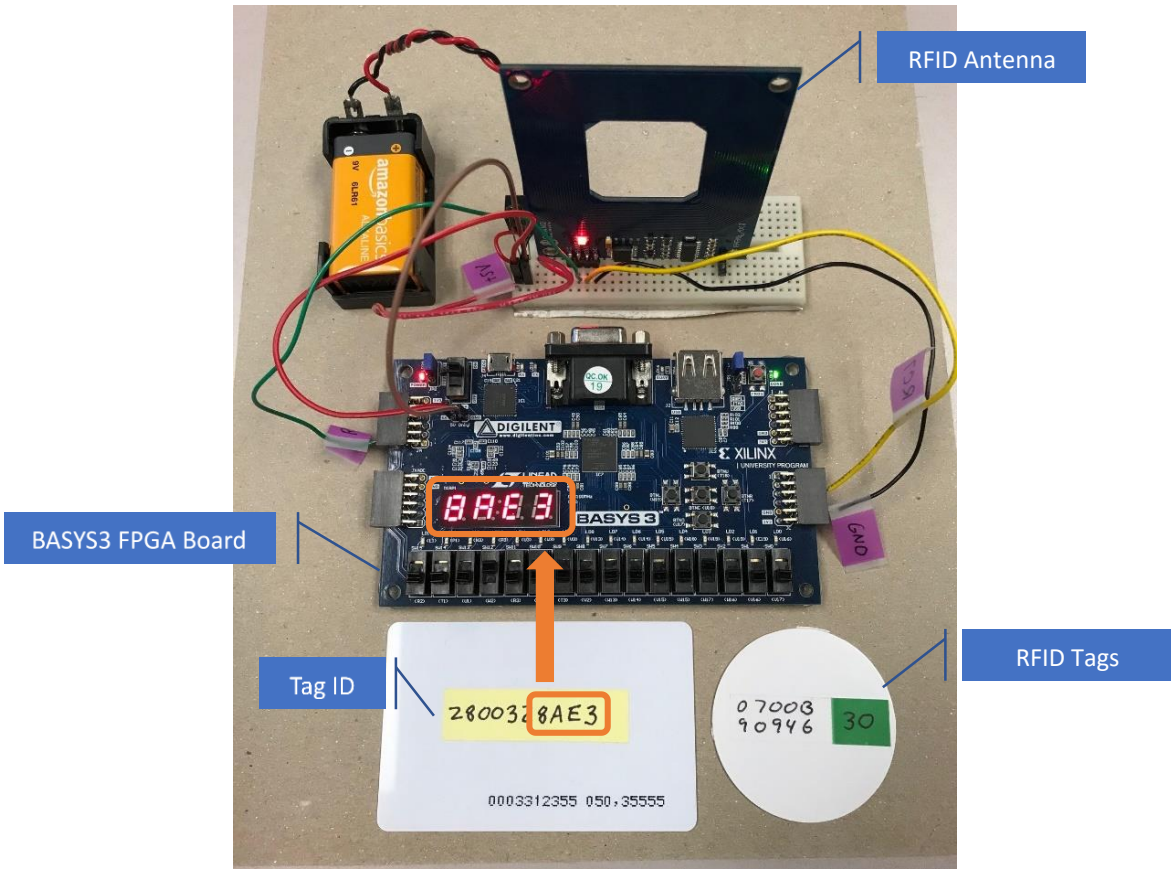


Figure 7. RFID Card Reader and BASYS 3 FPGA Board.

Results and Conclusions

The initial offering of this new Computer Architecture course produced great results. Even though the students had very mixed backgrounds and VHDL experience, all were able to successfully complete the homework assignments and design project. Realization of only a subset of the processor instructions provides sufficient capabilities to demonstrate fundamental programming concepts. Additional instructions, implemented as homework assignments, allow direct application of the design techniques taught in class.

Integration of the UART, timer, and interrupt systems provide the students with an in-depth look at the design and interfacing of internal and external components. The RFID microcontroller application demonstrates the multi-tasking capabilities of the expanded system. Hardware prototyping using an FPGA provides hands-on experience that can't be obtained by simulation alone.

Student feedback was collected from the Student Evaluation of Learning survey, using a five-point Likert scale: 1. Strongly Disagree (SD), 2. Disagree (D), 3. Undecided (U), 4. Agree (A), 5. Strongly Agree (SA), Number of Responses (N). The results are very positive and shown in Figure 8.

Student Evaluation of Learning	SD	D	U	A	SA	N	Mean
My ability to identify, formulate, and solve complex engineering problems by applying principles of engineering, science, and mathematics has improved as a result of this course.	0	0	0	0	4	**4**	**5.0**
My ability to develop and conduct appropriate experimentation, analyze and interpret data, and use engineering judgement to draw conclusions has improved as a result of this course.	0	0	0	1	3	**4**	**4.8**
My ability to acquire and apply new knowledge as needed, using appropriate learning strategies has improved as a result of this course.	0	0	0	0	4	**4**	**5.0**

Figure 8. Student Evaluation of Learning Survey Results.

Sample student responses to survey questions are also presented in Figure 9.

What did you like most about this course?

"The use of multiple tools for learning. Dr. Hayne made the class very interesting. The lecture slides and homework assignements were very helpful tools to help us learn the material."

"Gaining an understanding of computer architecture design gives a clear picture in the inner workings of processors and microcontrollers. I've worked on computers since I was 14 but the innerworkings of components on boards were always a mystery to me. This course brings together Discrete mathematics, digital systems, and design into a single class with a focus on design. I feel much more comfortable in my understanding of computer hardware design and searching for jobs in the related field now."

"I really enjoyed the focus on understanding concepts so that our design practices can be applied to a wide range of problems."

Figure 9. Sample Student Responses to Survey Questions.

In conclusion, this new course has successfully used the expanded Instructional Processor to teach Computer Architecture using VHDL modelling and simulation, combined with FPGA implementation and testing. The design project serves as a good assessment of the students' understanding of key design concepts via their use of industry-standard tools. Feedback has been very positive that the course illustrates fundamental design concepts reinforced by actual functioning hardware.

References

[1] R. Hayne and J. Moore, Jr., "Evolution of the Instructional Processor," *Computers in Education Journal*, ASEE, Vol. 6 No. 4, October - December 2015.

[2] R. Hayne, "Design of an Instructional Processor," Supplement to: C. Roth and L. John, *Digital Systems Design Using VHDL*, Third Edition, Boston, MA: Cengage Learning, 2018**.** [Online]. Available: http://academic.cengage.com/resource_uploads/downloads/1305635140_559956.pdf.

[3] RISC-V International. [Online]. Available: https://riscv.org/.

[4] S. Harris, D. Chaver, L. Pinuel, O. Kindgren, and R. Owen, "RVfpga: Computer Architecture Course and MOOC using a RISC-V SoC Targeted to an FPGA and Simulation," *Proceedings ASEE Annual Conference and Exposition*, Baltimore, MD, June 2023.

[5] Grenoble Institute of Technology, "LeaRnV: RISC-V based SoC Platform for Research Development and Education." 2020. [Online]. Available: https://tima-amfors.gricad-pages.univ-grenoble-alpes.fr/learnv/.

[6] Arm Introduction to Computer Architecture. [Online]. Available: https://www.arm.com/resources/education/education-kits/computer-architecture.

[7] Vivado Design Suite User Guide, UG892, v2019.1, Xilinx, Inc., 2019.

[8] D. Patterson and J. Hennessy, *Computer Organization and Design MIPS Edition*, Sixth Edition. Boston, MA: Morgan Kaufman, 2021.

[9] R. Hayne, "Implementing Serial Communication for the Instructional Processor," *Proceedings ASEE Virtual Conference*, June 2020.

[10] R. Hayne, "Synthesis vs. Simulation: Developing a Hardware Interrupt System for the Instructional Processor," *Proceedings ASEE Southeastern Section Conference*, Charleston, SC, March 2022.

[11] BASYS 3 FPGA Board Reference Manual, Digilent, Inc., April 2016.

[12] RFID Card Reader, Serial (#28140), v2.2, Parallax, Inc., March 2010.