Animation of a VHDL model in Modelsim using Tcl/Tk

David Sullins and Hardy J. Pottinger

Department of Electrical and Computer Engineering
University of Missouri - Rolla

Abstract

Visualization of the operation of a processor model is a difficult process that can be made more attractive by using Tcl/Tk procedures built into Modelsim. This paper describes the use of Modelsim and Tcl/Tk to display and animate a block diagram of a synthesizable model of a subset of the 8051 microcontroller (WIMP51). The model is being used to introduce Electrical and Computer Engineering undergraduates at the University of Missouri - Rolla to computer architecture and the 8051. The WIMP51 is binary compatible with a subset of the 8051's instructions, can be programmed with a standard 8051 assembler, and can be realized in a Xilinx 4005 FPGA.

1. Introduction

The University of Missouri-Rolla offers a course in hardware/software co-design and an associated lab. The students in this course study the 8051 family of microcontrollers for most of the course. However, an introduction to computer architecture is provided at the beginning of the course. Since the 8051 has a relatively complex architecture, a simple 4-bit processor called the Gnome[1] has been used in the past to provide a gentle introduction to computer architecture, before the 8051 is introduced.

The use of two very different processors in the class has caused confusion among students. Course evaluations reveal that many students feel their time was wasted learning the Gnome instruction set, only to be told to forget it and learn a new instruction set three weeks into the course. However, the Gnome processor has been useful for the purposes of the lab portion of the course, since it can easily be simulated in CAD tools and implemented in an FPGA. The 8051 microcontroller is complex enough that students might be unprepared to debug hardware problems immediately at the beginning of the lab course.

To resolve these problems, a replacement processor was created, the WIMP51. It is a simple subset the 8051, lacking internal memory, interrupts, and peripherals. This processor was implemented in VHDL using Modelsim and synthesized using Leonardo. A Tcl/Tk-based interface to Modelsim was created to allow students to interactively view the dataflow through the processor when simulating their own programs. This paper will describe the architecture of the WIMP51, the Tcl/Tk interface, and how all of this is used in the courses at the University of Missouri - Rolla.

2. Processor Architecture

2.1 Processor Overview

The WIMP51 processor shares several features with the Gnome processor it replaces:

- **Simple architecture**: The students need to be able to comprehend the entire datapath of the processor. They are required to complete exercises where they extend the processor with new instructions. Comprehension of the datapath is essential to completing these exercises.
- **Simple timing**: All instructions execute in the same amount of time. The WIMP51 processor uses three clock cycles per instruction (Fetch, Decode, and Execute in the style of the classic Von Neumann machine).
- **Small instruction set**: The number of instructions should be kept to a minimum in order to keep the design of the hardware very straightforward.
- **Synthesizable model**: In the lab portion of the course, students will simulate the processor interfaced with external memory. Then they will test that same design using an FPGA. A synthesizable simulation model is required for the lab.

Additionally, the new WIMP51 has certain features not shared with the Gnome:

- **Assembly language 8051 compatibility**: Every instruction that executes on the WIMP51 has an exact equivalent in the 8051 instruction set.
- **Machine code 8051 compatibility**: Any standard 8051 assembler can be used to produce the machine code for the WIMP51.

2.2 Instruction Set

All WIMP51 instructions are one or two bytes long. When choosing which 8051 instructions would be included in the instruction set, only immediate addressed and register addressed instructions were considered since the WIMP51 would include no internal memory or special function registers. To adhere to the three clock cycle Fetch/Decode/Execute timing, it would be logical to choose one byte instructions. However, there is no way to load an immediate value or perform a jump on an 8051 with only one byte instructions. Since both of these functions were considered necessary, two byte instructions were also examined. In the end it was decided to use two byte instructions only for two immediate addressed instructions and two jump instructions. All other instructions would be one byte long.

After examining the remaining instructions, a minimal subset of the interesting instructions was chosen. The AND, OR, XOR, and ADD functions were considered to be the most important and form the core of the instruction set. The SWAP A instruction was added later to facilitate the development of a laboratory exercise. The final instruction set can be seen in Table 1.

Table 1: WIMP51 Instruction Set

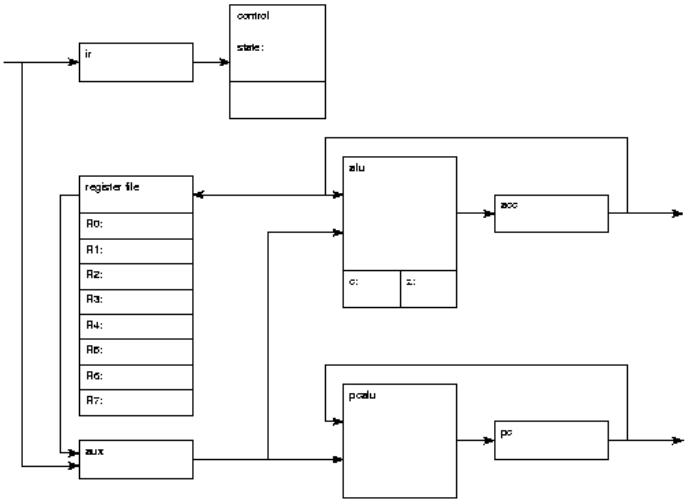
Instruction	Operation
MOV A, #data	A <= data
MOV A, Rn	A <= (Rn)
MOV Rn, A	Rn <= (A)
ADDC A, #data	C,A <= (A) + data + (C)
ADDC A, Rn	C,A <= (A) + (Rn) + (C)

XRL A, Rn	A <= (A) ^ (Rn)
ANL A, Rn	A <= (A) & (Rn)
ORL A, Rn	A <= (A) | (Rn)
SWAP A	A <= $(A)_{3-0}$ $(A)_{7-4}$
SETB C	C <= 1
CLR C	C <= 0
SJMP relative	PC <= (PC) + relative
JZ relative	PC <= (PC) + relative if A=0

2.3 Hardware Design

The WIMP51 has eight eight-bit general purpose registers, R0-R7. It also has an instruction register (IR), an auxiliary operand register (AUX), an accumulator (ACC), and a program counter (PC), all eight bit registers.

Figure 1: WIMP51 Block Diagram



In addition to the registers, the WIMP51 has three logic units. The control unit (CU) decodes each instruction and controls the timing for all the control signals in the processor. The eight-bit PC adder/incrementer (PCALU) takes the PC and the AUX as inputs and writes the result into the PC. The eight-bit arithmetic logic unit (ALU) takes input from the ACC and the AUX and writes its result into the ACC. The ALU also contains a 1-bit carry register (C) that stores the carry from the previous ADDC operation.

A block diagram describing this architecture is shown in Figure 1.

In the Fetch clock cycle, a byte of code is fetched from external memory into the IR. The PC is also incremented in this clock cycle in preparation for the next instruction fetch.

In the Decode clock cycle, the AUX is possibly loaded with the second operand of the instruction. If the instruction is a jump or immediate-addressed instruction, then a second byte is loaded from program memory into the AUX, and the PC is incremented a second time. If the instruction is a register source instruction, the contents of the appropriate register are loaded into the AUX. Otherwise, nothing occurs in the Decode cycle.

In the Execute clock cycle, the destination register is loaded with the appropriate value. For accumulator destination instructions, the ALU result is loaded into the accumulator. For the MOV Rn, A instruction, the appropriate register is loaded with the value of the accumulator. For the two jump instructions, the PC is loaded with the result from adding the contents of the PC to the contents of AUX. See Figure 2 for an example timing diagram generated from Modelsim.

The WIMP51 has an eight-bit data input bus, an eight-bit address output bus, an eight-bit accumulator output bus, and an active-low program memory read control line called PSEN. The address bus is always driven with the value in the PC. The accumulator bus shows the value in the accumulator to verify correct execution of a program.

3. Tcl/Tk interface

3.1 Tcl/Tk Overview

Tcl (Tool Command Language) is a scripting language created by Dr. John K. Ousterhout. Tk (Tool Kit) is a set of graphical extensions to Tcl that provide an easy way of creating and manipulating graphical user interfaces. Together they form Tcl/Tk, a scripting language which can run standalone or be embedded inside of applications such as Modelsim [2].

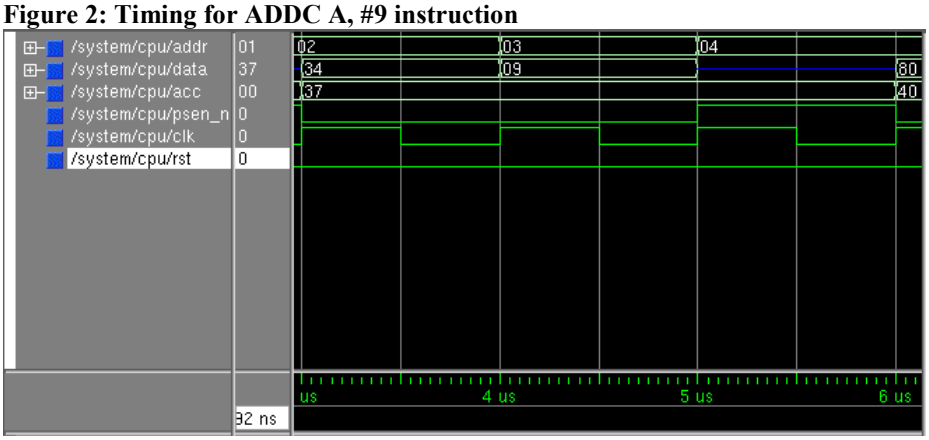
Many people use the Modelsim graphical interface without realizing that they are actually interacting with a Tcl/Tk interpreter. The Modelsim command window is a complete Tcl interpreter, capable of running any standard Tcl program. Additionally, all of the Modelsim windows are actually Tk windows and can be modified through standard Tk commands. This provides Modelsim with a great flexibility, as the user can modify the interface to suit his or her needs.

3.2 Visual WIMP51 Implementation

In the case of the WIMP51, the Tcl/Tk extensions were used to create a window that will interactively display the dataflow through the processor and highlight the changing register values as the user steps one clock cycle at a time or one instruction at a time through a program. Options to view the contents of memory, toggle the display of control signals, and print the display are included. An example screenshot is shown in Figure 3.

The program used to generate the interactive display is highly event-driven. Tcl provides a command called “trace” which allows a procedure to be called every time a variable changes. A Modelsim-specific command called “when” is used to copy the values of VHDL signals into Tcl variables. A trace is placed on each of these signals to update the display every time their values change. When a register’s value has changed since the previous clock cycle, its value is highlighted in red. The dataflow through the processor is highlighted in blue. For

Figure 2: Timing for ADDC A, #9 instruction



example, for the execute cycle of the instruction "ADDC A, #9" shown in Figure 3 the datapath from the AUX to the ALU and from the ACC to the ALU is highlighted. The display on the ALU is also updated and highlighted in red to show that the current ALU operation is "ADD."

4. Class Usage of the Tcl/Tk Interface

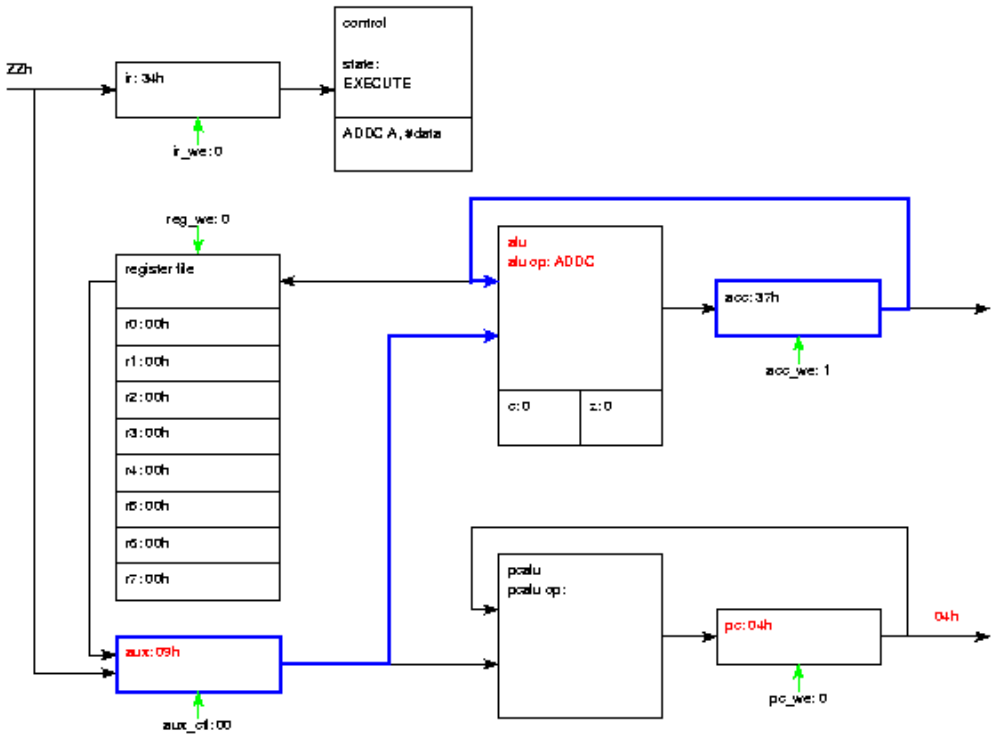
4.1 Laboratory Setup

The students taking the course where the WIMP51 is used have already had experience using Design Architect and Quicksim II in previous coursework. For the WIMP51 experiments, they use QSPro for hardware simulation. QSPro combines Quicksim II, used for schematic simulation, and Modelsim, used for VHDL simulation. The students mostly interact with the familiar Quicksim interface, but use Modelsim to launch the graphical WIMP51 interface. They also have an opportunity to test the hardware itself in a Xilinx XC4005XL FPGA on the XS40 board from the Xess Corporation [3].

4.2 Laboratory Exercises

The WIMP51 processor is currently being used in a lecture course and the associated laboratory course at the University of Missouri - Rolla. Two laboratory exercises include the use of the WIMP51.

Figure 3: Screenshot of Visual WIMP51



The first laboratory exercise requires the students to write a simple WIMP51 program and debug it using QSPro. Students set up QSPro to trace the external signals of the processor, give the processor the appropriate inputs, and then use the "Visual WIMP51" feature to see the internal processor signals and debug their program. When they have finished debugging their program, they print out the traces from QSPro, and write in the disassembled instructions.

The second laboratory exercise requires the students to test their program from the previous exercise in hardware. But we challenge them by giving them modified hardware. We change the operation of one instruction so that their program will not work. They then have to use a digital oscilloscope to analyze what the processor is doing and determine which instruction was implemented incorrectly by comparing to the traces from the previous exercise. The students must modify their program to work around the hardware bug. Discovering exactly where the hardware bug occurred would be difficult if not impossible without having first verified correct operation of the program in simulation. Having a good simulation environment assures the students that the problem is caused by hardware, not their software.

5. Conclusion

The WIMP51 meets our goal of providing an 8051 subset that can be used to teach students the basic concepts of processor architecture. The Tcl/Tk animation of the dataflow through the WIMP51 gives the instructor a powerful visual aid when describing the operation of the processor. It also gives the students a way of easily verifying correct operation of their programs while learning more about what is happening inside the processor. While we used Tcl/Tk as a way to make concepts clearer to students, it is also easy to see how a similar program could be used to facilitate debugging a complex VHDL model. The embedded Tcl/Tk interpreter makes Modelsim an extremely flexible tool.

References

[1] *The Practical Xilinx Designers Lab Book*, D. Van den Bout, Prentice Hall, 1999
[2] Tcl Developer Xchange: tcl.activestate.com
[3] Xess Corporation website: www.xess.com