Department of Electrical and Computer Engineering
The University of Texas at Austin

EE 382N, Spring 2004

Homework 4

Due: 5 March 2004, 5:00 pm in Room 541 ENS
Yale Patt, Instructor
Hyesoon Kim, Moinuddin Qureshi, TAs

Part 1. The next step in your computer design is to add the control. This is the last assignment that you will work on by yourself. Starting with HW 5, you will be part of a group, and we will expect to see a single solution from the entire group.

In this assignment, you are asked to specify the mechanism for generating the control signals that control the data path. This means all the control signals that will be needed every cycle to process every opcode, data type, and addressing mode that has been specified thus far. If you are planning to use microprogrammed control, these control signals will be generated centrally. Your job in this assignment is to specify the microinstruction, its fields for controlling the data path and the microsequencer, and the micro-orders that make up each field. If you are planning to pipeline your design (highly recommended), then these control signals will be distributed over the various stages of the pipeline for processing each ISA instruction through the entire pipeline. Where each control signal for each stage is generated is your decision. Also, for a pipelined design, it is important to provide sufficient interlocks to ensure safe exectuion of each instruction through the pipeline.

Your specification at this point does not have to specify the actual logic required to specify each control signal. However, we do want to see input/output behavior; that is, we want to see what input signals are required to specify each control signal. These can be shown as block diagrams on your drawings.

Hand in hard copies of the schematics of the control structure you create.

Part 2. Specify all the control logic required to completely emulate one of the three sets of x86 instructions specified below. Complete emulation means from the fetch of the first instruction to the destination write of the last instruction. Use Verilog to simulate and verify that your design works for the set of x86 instructions you have chosen. Use Virsim to monitor important data and control signals in the cpu. Hand in annotated hard copies of your Virsim graphs. Please make sure that all signals on the printouts are readable. Use several pages if necessary.

Part 3. Hand in a block diagram of your datapath with explanations to clarify all signals shown.

Assume that testcase starts at memory location x3000

```
SET1 : Dependency test
          MOV EAX,0x12340000   //  B8 00 00 34 12
          MOV ECX,0x00001234   //  B9 34 12 00 00
          ADD EAX,ECX          //  01 C1
          ADD EAX,#-1          //  83 C0 FF
          XCHG EAX, ECX        //  91
          HLT                  //  F4


SET2 : Control flow test
          MOV EAX, 0x12341234  // B8 00 00 34 12
          JMP Down             // E9 05 00 00 00
          MOV EAX, 0xFADEDACE  // B8 CE DA DE FA
  Down    ADD EAX, #1          // 05 01 00 00 00
          HALT                 // F4



SET3 : Register file test
          MOV EAX, 0xCAFEFFFFF  // B8 FF FF FE CA
MOV CL,0X01           // B1 01
MOV DL,0X02           // B2 02
ADD AX, 01            // 66 05 01 00
XCHG CL, CH           // 86 CD
HLT                   // F4
```