Computer Science 605.611

Problem Set 5

1. Shown below is the FSM (finite state machine) that models the behavior of the multicycle data path. Each state in the diagram is labeled with the time required by the hardware to complete the activity required in that state. Each state corresponds to a single CPU clock cycle.

Diagram

Description automatically generated

a) (5) Is this FSM an example of a Moore machine or is it an example of a Mealy machine?

Moore machines and Mealy machines were defined in module 5. Explain your answer.

This is a Mealy Machine. This is because the output depends on the state of the machine as well as the inputs, Once you pass start you can see that bubble 4 as well mas many other bubbles also go into the instruction fetch meaning it needs that information to continue.

b) (5) Based on the FSM diagram, what is the maximum CPU clock rate that can be used for the system if each state must consume one clock cycle? Express your answer in MHz.

Critical Path = 0-1-2-3-4 which is 210ns (minimum cycle time)

Max clock rate = 1/210ns = **21 MHz**

c) (5) Based on the FSM diagram, how many nano-seconds are required to execute a MIPS sw instruction if each state must consume one clock cycle?

**190ns**

2. Suppose we replace the lw and sw instructions with new versions. Instead of using a base register plus a sign-extended displacement to compute the memory data operand address, the new replacement sw and lw instructions compute the memory data operand address as the sum of the base register (rs) contents plus the contents of a second register called the index register (rt). A third register (rd) is loaded with the value read from memory by the new lw instruction. The rd register contains the value to be written to memory by the new sw instruction.

For example: lw $2, $4, $6 reads the contents of the memory word at address ($4)+($6) and places the word into register $2. The instruction sw $2, $4, $6 writes the contents of $2 to memory at the address ($4)+($6).

1. (5) Show the machine code format that should be used for these new lw and sw instructions.

The Machine code format that would allow for this would be

Opcode: 6 bits

RD: 5 bits

RS: 5 bits

RT: 5 bits

For both Load word and save word

1. (5) List the minimum modifications required to the multi-cycle FSM to implement this change.

Some of the modifiactions would include.

Modifying the instruction decoding so it understands that it is no longer looking for a 16 bit offset but instead a register.

Update the ALU to calculate the memory address of as the sum of RS and RT for these instructions

Modify the control signals to accept the new operation

Adjust timing as needed due to how the instructions are handled

1. (3) Explain why implementing these new instructions on the micro-programmed controlled unit is easier than implementing them on the hardwired control unit.

There are a few reasons for this. Micro-programs are much more flexible and easily modifiable where a hardwired CU is much more difficult to change. Micro-programmed control unit is also easier to debug and test for a similar reason of, to change it means to actually change the hardware of the hardwired CU.

3. Answer the following questions about the micro-programmed implementation of the multi-cycle data path control unit described in module 5.

a) (5) Running at the same clock rate, why does it take longer to execute the MIPS load word instruction lw $2,8($3) on a micro-programmed version of the control unit than on a hardwired implementation?

The reason for this is in our micro-programmed version of the CU we do not have to spend any time decoding. When we do it through a hardwired implementation we have to encode and decode which slows down speeds.

b) (5) How many bits are in each micro-instruction for the system described in module 5?

Each micro-instruction is 18 bits wide.

c) (5) What is the purpose of the micro-PC and how many bits does it contain?

The Micro-PC is what used to be the state register, however unlike the state register it is pointing to items within the control memory that’s on the same chip as the CPU. It contains 4 bits.

d) (5) Why does the CPU use a fixed clock cycle time rather than a variable clock cycle time for the multi-cycle data path?

It uses a fixed clock cycle because it needs to determine the next state of the multi-cycle data path. In the implementation of the multi-cycle data path from module 5 is a Mealy machine meaning the state helps to determine the output. If the clock cycles were variable than we could run into issues where the machine has to slow down or crash due to not knowing what state it is in yet.

4. The following diagram shows a slightly modified version of our MIPS multi-cycle datapath. For each of the instructions listed below (j, bne, and jr), explain why this datapath supports the instruction or does not support the instruction.

Diagram

Description automatically generated

a) (3) j (jump)

This datapath does support the jump instruction. The main feature that is required for the jump instruction is the jump bit, which this datapath contains. It also has a multiplex which is needed and passes the target address to the PC.

b) (3) bne (branch if not equal)

This datapath does support the bne instruction. For one it handles the branch opcode. It also sends the branch code to a gate that checks with the Zero instruction to determine if the branch is met or not. This is then sent to the PC to update the target address or increment.

c) (3) jr (jump register)

This datapath does not support the jump register instruction. This is because, while it does handle the jump instruction it is not setup to handle the register part of this instruction. Unlike j jr gets the address from a register, the jump path of the above datapath doesn’t touch registers at all meaning that it could not support the jr instruction.

5. a) (5) Explain how the finite state machine, described in module 5 and shown again below, can be extended or changed to include support for the addi instruction. Specify any new states required and indicate the values of the control bits used in each new state. Do not modify or duplicate any of the states already present in the finite state machine.

.

Diagram

Description automatically generated

To implement the addi instruction into our fsm we need to do a few things. We need to add a new state that supports the addi instruction, in this state we would need to add the control bits for register read, the ALUop, and register write. This new state would be added where it is sent to the addi state after decoding, then sent back to initial state/next instruction state.

b) (5) Assume that the micro-programmed system, described in module 5, is extended or changed to support the addi instruction. Use five hex digits to show each of the micro-instructions in the new micro-program that implements the addi instruction.

0x00018 (go to state 1)

0x0800B (go to new state 10 for addi)

0x09408 (return to state 0)

c) (5) Use hexadecimal to show each of the micro-instructions in a new micro-program that could implement the MIPS nop instruction.

0x09408 (send it back to state 0)

6. (9) If each micro-instruction takes 4ns to complete, what is the total number of nano-seconds required to execute each of the instructions listed in the table below on the micro-programmed implementation of the multi-cycle datapath.

|  |  |
| --- | --- |
| Instruction | Required execution time (ns) |
| add | 40ns |
| sub | 40ns |
| slt | 40ns |
| and | 40ns |
| or | 40ns |
| lw | 80ns |
| sw | 80ns |
| beq | 60ns |
| j | 40ns |

7. (5) Explain the purpose of dispatch ROM1 and dispatch ROM2 used in the micro-programmed implementation of our MIPS control unit.

In our implementation of the MIPS control unit micro-program ROM1 and ROM2 are integral to how the control unit manages the flow of execution. Each of the ROMs help determine the next states to transition to based on the instruction being executed.

8. Recall that our MIPS core instruction subset consists of the instructions: add, sub, slt, and, or, lw, sw, beq and j.

a) (3) What ALU operation is performed for the slt instruction on the multi-cycle data path?

For the slt instruction on the multi-cycle data path the “less than” operation is performed.

b) (3) What ALU operation is performed for the beq instruction on the multi-cycle data path?

The ALU operation performed for the beq instruction is the subtraction. The ALU subtracts the value from one register in another register if its 0 branch is taken if not it is not taken.

9. (5) If the third micro-instruction in a certain micro-program is 0x00113, show in hex what the next micro-instruction should be.

The instruction is in step 6 (0000 0000 0001 0001 0011), so we need it to be in step 7 for the next instruction. The binary for 7 is 00 0000 0000 0000 1100 = **0x0000C**

10. (3) What is the minimum size in bits for the control memory containing only the micro-instructions for our MIPS core instruction subset?

The minimum is 18 bits.