# EE457 MT (~20%)

Closed-book Closed-notes Exam; No cheat sheets; Calculators are allowed. Verilog Guides are allowed but not needed.

## Spring 2015

**Instructor:** Gandhi Puvvada

Friday, 3/27/2015

09:00 AM - 12:00 Noon (3 Hour 00 min. = 180 min)

Location: THH301

---

**Student’s Last Name:** ____________________________

**Student’s First Name:** ___________________________

**Student’s DEN Bb username:** ______________________@usc.edu

---

<table>
<thead>
<tr>
<th>Ques#</th>
<th>Topic</th>
<th>Page#</th>
<th>Time</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2nd edition multi-cycle CPU</td>
<td>2-4</td>
<td>40 min.</td>
<td>48</td>
</tr>
<tr>
<td>2</td>
<td>Lab 7 Part 1 3-elem. adder pipeline</td>
<td>5-7</td>
<td>50 min.</td>
<td>69</td>
</tr>
<tr>
<td>3</td>
<td>5-stage Early branch and Branch Delay slot</td>
<td>8-9</td>
<td>50 min.</td>
<td>59</td>
</tr>
<tr>
<td>4</td>
<td>convert 5-stage early branch to 3-stage</td>
<td>10-11</td>
<td>20 min.</td>
<td>35</td>
</tr>
<tr>
<td>5</td>
<td>Cache (reverse engineering)</td>
<td>12</td>
<td>15 min.</td>
<td>36</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td></td>
<td><strong>Cover +11</strong></td>
<td><strong>175 min.</strong></td>
</tr>
</tbody>
</table>

**Perfect Score**

---

Viterbi School of Engineering

**University of Southern California**
Improve the 2nd edition multi-cycle CPU:

Reproduced on the next two pages are the 2nd Edition CU (Control Unit) state diagram and the DPU (Data Path Unit). Miss Trojan suggested that, instead of returning to state 0 and then fetching the next instruction, you can prefetch the next instruction in the last clock of the current instruction and return to state 1, thereby saving a clock. She wanted to do this in the case of a lw (load word) instruction, a R-Type instruction and a jump instruction. She was about to copy all the signals of the state 0 into the three states, 4, 7, and 9, but realized that she has to do something special in the case of a J (a JUMP instruction) (state 9). She added a mux next to the PC in the DPU and named the select line PFCJ (PFCJ = Prefetch Control for Jump) and made it a "1" in the state 9 as shown.

She left the design for you to complete in the next two pages.
(a) add "PFCJ = 1" or "PFCJ = 0" in a subset of the remaining 9 states where it matters.
(b) add new state transition arrows as needed to return to state 1 and delete some earlier state transition arrows returning to state 0.
(c) In the datapath, complete the connections to the new mux and break or make any other connections as needed.
(d) adjust any other signals such as PCSource.

1.1 State briefly, what is the difference between prefetching the next instructions from states 4 and 5 as compared to prefetching the next instructions from state 9?

____________________________________________________________________________
____________________________________________________________________________
____________________________________________________________________________
____________________________________________________________________________

1.2 Why didn’t Miss Trojan try to prefetch the next instruction from states 5 (last clock of SW) and state 8 (last clock of beq)?

____________________________________________________________________________
____________________________________________________________________________
____________________________________________________________________________
____________________________________________________________________________

1.3 Why similar prefetching is not possible in the first edition design of the multi-cycle CPU?

____________________________________________________________________________
____________________________________________________________________________
____________________________________________________________________________
____________________________________________________________________________
Adjust control signals for state 9. Add or delete state transition arrows. Write "PFCJ = 1" or "PFCJ = 0" where needed.
Complete connections to the new mux. Make other needed adjustments.
Our VLSI engineer, Miss Bruin, was doing VLSI layout for our 5-stage pipelined 3-element adder (Lab 7 Part 1) and ended up adding a dummy stage between the original EX1 and EX2. So now we have a 6-stage pipeline with EX1, EX2 (the dummy stage) and EX3 (the original EX2). A Z\_Mux was added to the dummy stage on the next page. Complete the design on the next two pages.

Before you stall an instruction in ID stage, you make sure that the instruction you are stalling is not a NOP. T / F

Is this the same in Lab 6 5-stage pipeline discussed in the book/class? Y / N

Reason for similarity or difference: ______________________________________________________
___________________________________________________________________________________
___________________________________________________________________________________

Stalling: Spurious stalls ____________________________ (lower performance / produce wrong results).

Forwarding: You care to check if the senior providing forwarding help is not a NOP. T / F
You care to check if the junior receiving forwarding help is not a NOP. T / F

Is one of the above very important? Explain _______________________________________________
___________________________________________________________________________________
___________________________________________________________________________________

Can you stall the dependent instruction in EX1 stage instead of in the ID stage either in the original 5-stage pipeline or in this 6-stage pipeline?

Dependency for the Z register on a senior _________ (did / didn’t) lead to a stall in the original 5-stage pipeline. Dependency for the Z register on a senior _________ (did / didn’t) lead to a stall in this 6-stage pipeline.

For the sake of this exercise, is it possible to move one or two or three Z muxes from their current position to another stage? For each Z mux, state your answer using words such as possible or not possible and also desirable or not desirable. ______________________________________________________________
___________________________________________________________________________________
___________________________________________________________________________________
___________________________________________________________________________________

Similar to the "lw" delay slot, can you think of a delay slot in this design. If so how many delay slots? One or two or three? How does it affect your hardware design and cost? If the compiler designer is not quite smart and fills the delay slot using a NOP 90% of the time, is that worse than the hardware solution or still better? How? ____________________________
Complete the design (6 EN controls, RegFile write-port connections, forwarding paths, bubble-injection, etc.)

Generate on the next 2 pages
STALL, X_FORWARD1, Y_FORWARD1, Z_FORWARD1, Z_FORWARD2, Z_FORWARD3.

Pipelined 3-element Adder
Block Diagram
LAB 7 Part 1 with a dummy stage
3 (49 points) 50 min. Early branch and Branch Delay slot

The Block diagram for the Early Branch design from our lab 6 is given on the next page with support for JUMP instruction and one delay slot added as per the solution to question Q3.4.1 of Midterm Fall 2013.

3.1 Did we support delay slot for beq only or jump only or for both? _______________________
Explain: _______________________________________________________________________
_____________________________________________________________________________

3.2 IF.Flush is crossed off because ____________________________
______________________________________________________
_____________________________________________________________________________
Why VDD is connected to the wrist-band FF here? Can we connect GND (ground) instead? Can we remove the wrist-band FF all together? Does connecting the GND instead of VDD call for change in the Verilog code for the Wrist-Band FF? ________ (Yes/No). Any changes to the block diagram needed if we change VDD to GND?
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
If we remove the VDD connection and restore the crossed-off part, what happens to beq instruction and what happens to the j (jump) instruction? Would you suggest doing something more? Explain. ________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
If you need to add or delete or modify something, please show the same on next page.

3.3 Why are we clearing ID/EX, EX/MEM, MEM/WB stage registers on reset using ?
____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________

3.4 In a 7-stage pipeline (in one of the questions in the lab 6 part 4), as shown on the right, we made arrangements for instruction(s) in ______________ (each of IF1 and IF2 / only IF1 / only IF2) to wear a wrist-band, ______________ (and / but not ) carry it until it/they reach(es) the ID stage.
Modify the diagram on the right to support 2 (two) delay slots. Explain: __________________________________________
_________________________________________________
It is ___________ (OK / a SIN) to fill the delay slot of a branch with another branch instruction. Branch delay slots are filled by the compiler at compile time and not by hardware at run-time. T / F
(35 points) 20 min.  5-stage => 3-stage

We want to convert our 5-stage early branch pipeline to a 3-stage early branch pipeline by combining the IF and ID stages into one IF_ID stage and also combining the MEM and WB stages into one MEM WB stage. I have reproduced on the next page our 5-stage pipeline and just removed the IF/ID stage register and the MEM/WB stage register but did not do any consequential changes.

Timing: Let us assume that the original 5-stage pipeline was running at 100 MHz (10ns clock) and we will be running this 3-stage pipeline at 50 MHz (20ns clock). So we did not introduce any timing problems! Since the EX stage was not combined with any other stage, we are planning to add more complex arithmetic operations to ALU to make use of the 20ns clock. So we conclude that the MEM WB stage can not help the junior in EX stage but can help the junior in the ID stage through internally forwarding register file.

Briefly comment on the following areas (when space is provided) but you do not have to carryout any changes on the next page. If the unit being commented upon had comparators, state the number of comparators in it.

1. The wrist-band flip-flop and the two inverters associated with it should be removed. T / F
2. The AND gate after the equality checker shall be removed. T / F
3. Wrist-band FF is needed to wrist band the random instruction in the IF_ID stage when you switch on power to the 3-stage pipeline. T / F
4. Like before, clearing the stage registers IF_ID/EX and EX/MEM WB on reset, makes sure that on power-on, we have bubbles (i.e. no random instructions) in stages ____________________________.
5. The control unit continues to produce 9 control signals. Yes / No
6. Internally forwarding in the register file and the number of comparators in it: ____________

7. Successful branch, flushing, wrist-band FF, delay slot: ____________

8. HDU and STALL_LW, and the number of comparators in HDU:

9. HDU_Br and STALL_BEQ, and the number of comparators in HDU_Br:

10. FU_Br and the two forwarding muxes, and the number of comparators in HDU_Br:

11. FU and the 4 forwarding muxes, and the number of comparators in:

12. (Though / Since) number of stalls are less in the 3-stage pipeline, its overall performance is ____________ (higher / lower).
5 \( (18 + 10 + 8 = 36 \text{ points})\) 15 min. Cache and Main Memory Organization:

A 64-bit data (D63-D0) 32-bit (logical) address byte-addressable processor (address pins: A31-A3, /BE7-/BE0) has its cache and MM organized as shown below. Fill-in the 9 boxes.

5.1 Block size (based on degree of lower-order interleaving of the MM) = _______ Words = _______ Bytes
The design uses __________________ (set-associative / direct) mapping.
If set-associative, degree of set associativity = _______ blocks/set
The processor address space = _______ GBytes. Cache size = _______ KBytes

5.2 Please divide the address below into appropriate fields and name the fields.

---

It is not difficult to get an A in EE457. You need to work for it and seek help from the 457 teaching team on whatever you do not understand. We are eager to help you. The final topics, virtual memory, exceptions, branch prediction, out-of-order execution, chip multi-threading, chip multiprocessing, cache coherency, locks and mutual exclusion are interesting and challenging too. They are the focus of 50% of the final exam. Best wishes!