Final Exam Review

CS 64: Computer Organization and Design Logic
Lecture #18

Ziad Matni
Dept. of Computer Science, UCSB
SHIIIIIIIIIT! I'M RUNNING OUT OF TIME!

GUESS I'LL JUST MARK ALL THE ANSWERS "C" AND HOPE FOR THE BEST...

CS64 Final

1. Draw the truth table for A XOR B and describe it in loving detail.

2. Why is MIPS so cool? Write a poem about the MIPS Calling Convention.

3. Where in the world is Carmen Sandiego?
• Labs 9 & 10 due on Thursday 3/16 (today)
  – If you need an extension, let me know TODAY

• FINAL EXAM is on **Thursday March 23\textsuperscript{rd} at 4:00 PM**
  – In this classroom
  – Material included is: ...everything!...

• Prof. Matni has **extended office hours** tomorrow (Fri. 3/17)
  – From 11:30 – 2:30 in SSMS 4409
• Starts at 4:00 PM **SHARP**
  Please start arriving 5-10 minutes before class

• I will chose where you sit!
  Wait on my instruction...

• Duration: 3 hours long
  Use your time wisely (and a little less stressed)

• **Closed book:** no calculators, no phones, no computers

• Only 1 sheet (double-sided ok) of written notes
  Must be no bigger than 8.5” x 11”
  You have to turn it in with the exam

• DON’T FORGET YOUR MIPS R.C. !!!!!!!!!!!!!!!!!!!!!!!!!!!!!

• You will write your answers on the exam sheet itself.
Practice Questions

• Convert \textbf{1001 0010 1100 0011} to 4-digit hexadecimal

• Convert the signed binary value \textbf{1001 1111} to decimal

• Add the 2 following 8-bit numbers: \textbf{0110 0010} and \textbf{0011 0100} and indicate the status of the carry and overflow bits at the end of the addition. \textit{Interpret your findings.}
Practice Questions

Translate this MIPS assembly code into C/C++ code.

data
talk: .ascii "blabla"
cs: .word 3
.text
main:
    li $t0, 5
    la $t1, cs
    lw $t2, 0($t1)
    blt $t0, $t2, gothere
    li $v0, 4
    la $a0, talk
    syscall
    j end

gohere:
    li $v0, 4
    la $a0, talk
    syscall
    syscall

end:
    li $v0, 10
    syscall
Practice Questions

• Write the following MIPS instructions in machine-language hexadecimals (show all work):
  \texttt{sub \$r2, \$s4, \$t5}

• What will the final value in the MIPS register \texttt{\$s0} in this code be (in hexadecimal)? Show your work.

\begin{verbatim}
li \$s0, 20
sll \$s0, \$s0, 2
add \$s0, \$s0, \$s0
sra \$s0, \$s0, 4
\end{verbatim}
Practice Questions

• Consider the C code below:

```c
// arr is a globally accessible array of ints
// s0 already holds a value of type unsigned int
unsigned int s1 = arr[s0];
unsigned int s2 = arr[s0 - 1];
unsigned int s3 = arr[s0 + 1];
```

Using **no more than six instructions**, implement the above C code snippet in MIPS. You don’t have to follow the MIPS Calling Convention.
Consider the C code below.

```
int sum( int arr[], int size ) {
    if ( size == 0 )
        return 0 ;
    else
        return sum( arr, size - 1 ) + arr[size – 1];
}
```

Using the MIPS Calling Convention, implement this in MIPS assembly, taking care to preserve the relevant values.
Practice Questions

Simplify this expression using Boolean algebra:

\[ F = \text{NOT } ((A \text{ NOR } B) \cdot (C + A \cdot B)) \]

Also, draw the resulting circuit.
Using one or more D-Latch, one or more 2-to-1 mux, and any number of basic logic blocks (i.e. AND, OR, NOT, etc.), construct a circuit that takes 2 bits as inputs (B0, B1) and writes them to 2 registers called REG0 and REG1, respectively, but only when another input, E = 1. The registers retain their values if E = 0. The output of this circuit is the value stored in one the registers as chosen by inputs S1 and S0.

If S1=1 and S0=1, we should see the value in REG1 appear on the output (O) and if S1=0 and S0=0, we should see the value in REG0 appear at O. Any other combination of S1 and S0 should leave the output unchanged, that is the previous selection should remain the same.
Consider a device that consists of 2 buttons labeled “UP” and “RESET”, along with a light. The device internally counts the number of times “UP” is pressed, and when it is pressed two times, the device causes the light to illuminate. Additional presses of “UP” do nothing. Pressing “RESET” at any point will reset the internal counter back to zero, and will cause the light to go out. (Note that the light may have already been off, as when the user presses “UP” once followed by “RESET”.)

For this question, you will implement this device as a finite state machine. The machine has the following two external inputs:

- R: set to 1 whenever “RESET” is pressed
- U: set to 1 whenever “UP” is pressed

The machine also has one external output: L: set to 1 whenever the light should be illuminated

If both “RESET” and “UP” are pressed at the same time, then the behavior should be as if only “RESET” was pressed.

Draw the finite state machine diagram corresponding to this task. All transitions should be drawn as products of R and U. For example, if a particular transition should be taken only if R = 1 and U = 0, then this should be drawn as R.U̅ or R.!U.

If we use the “one-hot method”, how many D-latches are necessary to implement this state machine? Write all the logic formulas that describe all the states, as well as the output L.
Answer to the last question

Using the one hot method, we’d need 3 DFFs. The formulas for each state and the output would be:

\[ S0^* = R + S0 \cdot \bar{U} \]
\[ S1^* = S0 \cdot U \cdot \bar{R} + S1 \cdot \bar{U} \]
\[ S2^* = S1 \cdot U + S2 \cdot \bar{R} \]
\[ L = S2 \cdot \bar{R} \text{ (or just } S2) \]