**SOLUTION**

**EE254L Final (~20%)**
Open-book Open-notes Exam; No loose sheets
Calculators and Verilog Guides are allowed.

**Spring 2015**
Instructor: Gandhi Puvvada
Saturday, 5/9/2015
07:00 PM - 10:00 PM (3 Hour 00 min.)
Location: SGM101

Student’s Last Name: _______________________________________
Student’s First Name: _______________________________________  
Student’s USC username: ______________________________@usc.edu

<table>
<thead>
<tr>
<th>Ques#</th>
<th>Topic</th>
<th>Page#</th>
<th>Time</th>
<th>Points</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>State machine, OFL, waveforms</td>
<td>2-7</td>
<td>60 min.</td>
<td>82</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Mux, Shannon’s expansion theorem</td>
<td>8</td>
<td>15 min.</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>FIFO</td>
<td>9</td>
<td>35 min.</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td><strong>Miscellaneous</strong> (Handshake, I2C, UART, Rotating Prioritizer, Address Map, Address Decoding, Counter/State machine analysis)</td>
<td>10-11</td>
<td>60 min.</td>
<td>61</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>11</td>
<td>170 min</td>
<td>201</td>
<td></td>
</tr>
</tbody>
</table>

**Perfect Score**

Viterbi School of Engineering
University of Southern California
We need to inspect a 16 bit item B[0:15] starting from B[0] to find (a) the total number of 1’s (b) total number of chunks 1’s  and (c) number of 1’s in the largest chunk.

Complete the two separate state diagrams below. The two significant state are Z and NZ.

**Z** Zeros, go through zeros until a one is seen

**NZ** Non-Zeros, go through ones until a zero is seen

"I" is the index into B(I). "J" is the number of ones counter and "K" is the number of chunks of ones counter.

"N" is the counter to keep track of the number of ones in the current chunk of ones and "N_max" is a register holding the length of the longest chunk of ones found so far.

1.1 This state diagram is to find the number of ones "J" and the number of chunks of ones "K" in B. Increment K when you see the start of a new chunk.
1.1.1 For the first example, I have completed waveforms below. Please complete waveforms for the third example.
1.2 The goal of this state diagram is to find the length of the longest chunk of ones $N_{\text{max}}$ in $B$. Make $N \leftarrow 1$ on seeing the first one of a new chunk of ones in the $Z$ state. On seeing the first zero after the end of the current chunk of ones in the $\text{NZ}$ state, see if the length of the current chunk gathered in $N$ is greater than the length of the longest chunk found so far as saved in $N_{\text{max}}$ and update $N_{\text{max}}$ if needed.

Important note:

<table>
<thead>
<tr>
<th>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example 2: $N_{\text{max}} = 4$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example 4: $N_{\text{max}} = 1$</td>
</tr>
</tbody>
</table>

The chunk of ones could end at the very end of $B$ like in the examples 2 and 4 reproduced above. So you may not see a zero at (after) the end of the very last chunk. And you are in the process of counting the last 1 (that you are seeing) into $N$ but the $N$ counter has not updated with this value yet. If so, do you want to compare $N$ or $N+1$ with the current $N_{\text{max}}$? And, if needed, do you want to update $N_{\text{max}}$ to $N$ or $N+1$?

Please see what is provided for $N$ and $N_{\text{max}}$ in the DPU (datapath unit) on page 7 and figure out what you need to do in the $\text{NZ}$ state if you were about to move to the $D$ state from the $\text{NZ}$ state. Similarly see what you need to do to $N_{\text{max}}$ if you are moving from $Z$ to $D$ state in example #4 where you just found the single 1 and $\overline{N}$ is being made to become a 1.

Mr. Trojan says that this initialization of $N$ to 0 is actually not needed as he thinks that you will never increment $N$ from 0 to 1. Rather you always initialize $N$ with a 1 on detecting the first 1 in $\overline{B}(i)$.
1.2.1 For the first of the example lines, I have completed waveforms below. Please complete waveforms for the second and fourth examples.
1.2.2 Draw OFL for the second state diagram (for finding the length of the longest chunk of ones) based on the datapath on the next page. Assume one-hot implementation for the 4 states and use the one-hot flip-flop outputs ($Q_I$, $Q_Z$, $Q_{NZ}$, $Q_D$) as needed.
Supports both designs (both state diagrams on pages 2 & 4)
2 (6 + 9 + 4 = 19 points) 15 min.

2.1 Mux:
The first of the four muxes is a 2-to-1 non-inverting mux. Using minimum number of inverters, convert the rest of the three to a non-inverting mux (functionally equivalent to the first mux) and label the inputs and output using the same labels as the first mux (namely I0, I1, S, and Y).

2.2 Given below are Shannon’s Expansion Theorems:

\[
F(X_1, X_2, \ldots, X_n) = X_1 \cdot F(1, X_2, \ldots, X_n) + X_1' \cdot F(0, X_2, \ldots, X_n)
\]

\[
F(X_1, X_2, \ldots, X_n) = [X_1 + F(0, X_2, \ldots, X_n)] \cdot [X_1' + F(1, X_2, \ldots, X_n)]
\]

Now consider the Carry-Out from a full-adder. Cout is a function of (X, Y, Cin)

\[
Cout = F(Cin, X, Y) = X \cdot Cin + Y \cdot Cin + X \cdot Y = Cin \cdot (X + Y) + (X \cdot Y)
\]

If Cin = 0, then Cout = (X . Y) and if Cin = 1, then Cout = (X + Y) Note (X+Y) is a superset of (X.Y).

Now using an OR gate to produce (X+Y) and an AND gate to produce (X.Y), and using Cin as a late-arriving select line, produce Y using each of the 4 mux designs (with minimum number of additional inverters).

If Cin is not late arriving, would you still use (a) the above designs or (b) use (X . Cin + Y . Cin + X . Y = Cin)?

\[\text{We use } X \cdot Cin + Y \cdot Cin + X \cdot Y \quad \text{General guide line: 2-level logic is faster than a 3-level logic as long as we did not exceed the fan-in limit of any gate.}\]
3) (6 + 6 + 12 + 8 + 8 = 40 points) 35 min. FIFO

3.1 We can use \((n + 1)\)-bit\((n\text{-bit} / (n+1)\text{-bit})\) pointers either in the single-clock FIFO or in the
2-clock FIFO where as we can use \(n\text{-bit}\)\((n\text{-bit} / (n+1)\text{-bit})\) pointers only in the

\textbf{single-clok FIFO}\(\text{single-clock FIFO} / 2\text{-clock FIFO})

3.2 For a 32-location deep FIFO, when do you do mod-32 as shown in \([(\text{WP-RP})\text{mod32}]\) and when
do you do mod-64 as shown in \([(\text{WP-RP})\text{mod64}]\)? If the pointers are \(5\text{-bit} (n\text{-bit method})\) then
we use \((\text{WP-RP})\text{mod32}\). However, we know that \(n\text{-bit method works only in single clock FIFO and}
for \(2\text{-clock FIFO it needs a\(\epsilon\) FF to record and\(\epsilon\) FF recently it was running almost full or almost empty\). If we use 6-bit
pointer\((n+1)\text{-bit method})\). Then we use \((\text{WP-RP})\text{mod64}\) to arrive at depth. This method can be used for both single clock
and 2-clock FIFOs.

3.3 A new EE254L TA, Mr. Bruin, simulated a
FIFO design submitted by the student, Mr.
Trojan, and captured the 6-bit RPSS \((\text{RP}
double synchronized to Wclk}) activity for a 32-
location FIFO in a modelsim waveform along
with the 6-bit WP and the Wclk, and displayed the values of WP and RPSS in decimal as shown
above. Since the RPSS was jumping as shown below, he concluded that Mr. Trojan’s design is
bad. Please educate Mr. Bruin.

\textbf{Waveform (above)}

\begin{center}
\begin{tabular}{c|c|c|c}
Wclk & RPSS & WP \\
\hline
59 & 22 & \text{\textcolor{red}{23}} \\
\end{tabular}
\end{center}

\textbf{From the above double synchronization, Wclk, RPSS, it is clear that we are}
dealing with a 2-clock FIFO. And since RPSS is jumping from 59 to 5 (a jump by 16 steps) and from 5 to 19
(a jump by 14 steps), it implies that \(\text{RClk is much faster than Wclk}\). In one clock of \(\text{Wclk}\), the
consumer \(\text{working at faster Rclk can consume many data items and RP can increment several times.}
Obviously, when we sample \((\text{RClk slow}) \text{and Wclk fast})\), we can capture every value of \(\text{the fast changing RP}\).

3.4 We need to add Data 1 \((D1)\) produced by Producer \(P1\) and Data 2 \((D2)\) produced by Producer \(P2\)
and convey the Sum 3 \((S3)\) to Consumer \(C3\). The two producers and the single consumer work
at their own individual clocks \(P1\text{clk, P2\text{clk, and C3\text{clk}}}\). Also they are all busy with various
other things. So we need 3 FIFOs, one to collect a series of \(D1\) data, and another to collect a series
of \(D2\) data. We produce a series of sums \(S3\) and leave in a third FIFO for the consumer \(C3\)
to consume from. Complete the design below by producing \(\text{REN1, REN2, and WEN3}\) taking into
consideration the \(\text{Full and Empty signals as appropriate from the three FIFOs.}\)

\begin{itemize}
  \item If \(D1\text{out}\) is ready as indicated by \((\text{Empty1} = 1)\) and \(D2\text{out}\) is ready as indicated by
\(\text{(Empty2} = 1)\), and further \(S3\text{in}\) can be successfully deposited in the 3rd FIFO as indicated
by \((\text{Full3} = 1)\), then we activate all three control signals \(\text{REN1, REN2, and WEN3}\).
\end{itemize}

\textbf{Draw the logic below.}

\textbf{The adder circuit is a timing path in the \(C3\text{clk}\) (P1\text{clk/P2\text{clk/C3\text{clk}}}\) clock domain.)}
4 (10 + 8 + 6 + 6 + 8 + 6 + 2 + 6 + 9 = 61 points) 60 min. miscellaneous

4.1 Level-sensitive signals are used in _________ (2-way/4-way) handshake. 4-state state diagrams are used in _________ (2-way/4-way) handshake and when you go through the 4 states, you had transferred _______ (1/2/3) data items.

Full hand-shake is implemented in _________ (2-way/4-way) handshake and when you go through the 4 states, you had transferred _______ (1/2/3) data items.

A 2-clock FIFO is ___________ (superior/inferior) to _________ (2-way/4-way) handshake.

4.2 I2C: The Serial Data SDA and the Serial Clock SCL are ___________ (2-state / 3-state / open-collector) lines. SDA is driven by the slave during a ________ (read / write) transactions. To get more time (this is called handshaking in I2C), _________ (the Slave/the Master/either/neither) ______  (pull down / let go) the _______ (SCL/SDA) line. Two or more _______ (slaves/masters) arbitrate by watching the _______ (SCL/SDA) line to see if it went _____ (up/down) as desired.

4.3 UART: UART stands for Universal Asynchronous Receiver Transmitter.

The START bit is necessarily a _____ (0/1) where as a STOP bit is necessarily a _____ (0/1). Byte-by-Byte synchronization is done by "hunting" for the start of the _______ (START/STOP) bit using a ___________ (high-speed/low-speed) clock. This allows the transmitter clock and the receiver clock to differ as much as _______ (0.5% / 5%).

4.4 In a rotating prioritizer of 4 inputs and 4 outputs if the ID of the lowest priority requester is #1 then his request is pushed down to the lowest as shown on the side. Draw a similar line to show that the next requester (#2) is elevated to the top.

Muxes are present in _________ (IR/FPR/OR/multiple of these (list them))

Note: IR = Input Rotator, OR = Output Rotator, FPR = Fixed Priority Resolver.

4.5 The 20-bit address 57B21H (specified in hex notation) belongs to a natural 64K range 50000H - 5FFFFH. Specify the natural 16K and 1K ranges this address belongs. Note 64K = 2**16; 16K = 2**14; 1K = 2**10

Natural 16K range = ___________ (54000H to 57FFFH) Natural 1K range = ___________ (57800H to 57AFFH, 57B21H = 0101_0111_1011_0010_0001 Replace the lower 16 bits with 1s and 0s to get 2**16 = 64K range.

4.6 Find the size and address range of the shadowed area in the 2**20 address map below.

Size: ___________ KB = ___________ (14) Bytes

Note: Address pins A[13:0] change from 0's to 1's => Address pins A[13:0] change from 0's to 1's
4.7 Memory width expansion costs _______ (less than/more than/equal to) memory depth expansion.

4.8 Address decoding logic: A 2**20 range is represented by A[19:0] going from 20 zeros to 20 ones. Specify the ranges of addresses for the three low-active chip-select outputs and state if any of them are unnatural.

- **CS0**: A0000 H to A3FFF H Unnatural: Yes / No
- **CS1**: A4000 H to A7FFF H Unnatural: Yes / No
- **CS2**: A8000 H to AFFFF H Unnatural: Yes / No

4.9 A single 5-bit up/down counter is used in the state machine on the side to control the Jefferson/Vermont lights (Green and Yellow). It start with I == 0 in JG. The clock frequency is 1c/s. Find the number of seconds for each of the four light together with the range of values of the counter in that light.

- **JG**: 30 sec. I range = [00:29]
- **JY**: 5 sec. I range = [26:26]
- **VG**: 11 sec. I range = [10:0]
- **VY**: 6 sec. I range = [31:26]

---

Rough work area

---

The TAs, the Graders, and I have enjoyed teaching this course. Hope you also liked it. Hope to see many of you in EE457 again. Have a great Summer break!

-- Gandhi