CS 151
Final

Name : __________________, __________________
      (Last Name)                  (First Name)
Student ID : __________________
Signature : __________________

Instructions:

1. Please verify that your paper contains 15 pages including this cover.
2. Write down your Student-Id on the top of each page of this final.
3. This exam is closed book. No notes or other materials are permitted.
4. Total credits of this final are 100 + 10 EXTRA CREDIT.
5. To receive credit you must show your work clearly.
6. No re-grades will be entertained if you use a pencil.
7. Calculators are NOT allowed.

<table>
<thead>
<tr>
<th></th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Credit</td>
<td>25</td>
<td>15</td>
<td>30</td>
<td>15</td>
<td>25</td>
<td>110</td>
</tr>
<tr>
<td>Score</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Q1: Design of a Moore Machine [25 points]

We want to design a non-resetting sequence detector using a finite state machine with one input X and one output Y. The FSM asserts its output Y when it recognizes the following input bit sequence: "1101". The machine will keep checking for the proper bit sequence and does not reset to the initial state after it has recognized the string.

[Note: As an example the input string X= "..1101101.." will cause the output to go high twice: Y = "..0001001.."

(a) Capture the Moore FSM. (10points)
(b) Create the architecture (3 points)

```
Combinational Logic Circuit

State Register
```

\[ X \rightarrow \text{Combinational Logic Circuit} \rightarrow Y \]

\[ \text{s2, s1, s0} \rightarrow \text{n0, n1, n2} \]
(c) Encode the states (use a simple binary encoding)  (2 points)

\begin{align*}
S0 &= 000 \\
S1 &= 001 \\
S2 &= 010 \\
S3 &= 011 \\
S4 &= 100
\end{align*}

(d) Create the state table  (10 points)

<table>
<thead>
<tr>
<th>Current State</th>
<th>Input</th>
<th>Next State</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S2 )</td>
<td>( S1 )</td>
<td>( S0 )</td>
<td>( X )</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Q2: [RTL Design] [15 points]

We want to design a timer with 2 modes: **1Min**, **30Sec**. There is a **START** button and a **STOP** button for this timer. The timer is initially in **standby** mode. Then, we select a mode by pressing **1Min** or **30Sec**. After pressing the button **START**, the timer starts counting until it reaches the time (if it is in **1Min** mode it counts for 1 minute and if it is in **30Sec** mode, it counts for 30 seconds). Then it beeps and goes to **standby** mode. If during the time that the timer is working we press the **STOP** button the timer beeps and it goes to **standby** mode. If the timer is working in **30Sec** mode and the user presses **1Min** button, then after completion of 30 seconds the timer works for another 30 seconds. This can happen just once and then the timer beeps and goes to **standby** mode.

Assuming that we have necessary components to design the circuit for this timer, design the high-level state machine for this timer.
Q3: [Data Path Design] [30 points]

The following high-level state machine is designed to find an input X in a register file of size 1024. X is the input that we are searching for in the register file, START is a one bit input and RETURN and DONE are the outputs. $Mem[t]$ is the value stored in the register file at address $t$. 

![State Diagram]

- **Start**
  - $cnt = 0$, $return = -1$

- **(cnt < 1024)**
  - $cnt < 1024$
  - $(mem[cnt] = x)'$

- **cnt++**
  - $mem[cnt] = x$
  - $return = cnt$

- **DONE = 1**
1. Design the data path for this high-level state machine. [15 points]
Design the interface of the controller and the data path as well as the interface of the system. [5 points]
Design the controller’s FSM. [10 points]
Q4: [FSM Timing] [15 points]

The following FSM has one bit input I and one bit output O. (I/O is presented on each edge of the FSM).

1. Determine if it is a Mealy machine or a Moore machine. [2 points]

Mealy
2. In the following timing diagram show how the state and the output change with input. Assume that we initially start from state S0. [13 points]

NOTE: Assume that delay of the gates and the register is negligible.
Q5: [Component Design] [25 points]

1. We want to design a frequency divider circuit that divides a given frequency by 3 or 5. If the input MODE=0 it divides the frequency by 3 and if MODE= 1 it divides the frequency by 5. [15 points]

Use as many of the following as needed in your design:
   a- 3-bit counter with clear
   b- 2-to-1 multiplexer
   c- Logic gates (AND, OR, NOT)
\[ x \oplus 0 = x \]
\[ x \oplus 1 = \overline{x} \]

Second Method

Second Method

3-bit Counter

\[ f_{\text{div}} 3 \text{ or } 5 \]
2. Design a 16-to-1 multiplexer using ONLY 4-to-1 multiplexers. [10 points]