This is a closed-book, closed-note exam. No calculators. You have 75 minutes to finish the paper.

1. Given the following FSM circuit diagram:

(a) [7 points] In the table above, fill in the value of the outputs $f_0$ and $f_1$.

(b) [3 points] Draw the wave form for $f_0$ and $f_1$ below: (draw the clock cycle as a wave of alternating highs and lows)
(c) [2 points] Draw the state transition diagram for the above FSM:

(d) [Optional - extra credit, 5 points] convert the above implementation into a "one-hot-encoded" implementation. For this part, add a reset input signal (RST). Don’t forget to generate the outputs!

2. [4 points]
For the circuit shown, assume that the register setup time requirement is 1 ns and the clk to Q delay for the register is 1ns. What is the maximum delay possible through the adder and still guarantee correct operation at 100MHZ?

3. [6 points] Derive the truth table for the combinational logic circuit shown below:

\[
\begin{array}{ccc|c}
abc & f \\
000 & 0 \\
001 & 0 \\
010 & 0 \\
011 & 0 \\
100 & 0 \\
101 & 0 \\
110 & 0 \\
111 & 0 \\
\end{array}
\]

4. [6 points] For the boolean logic operation \((a + b + c)(d + e + f)(g + h + i)\),

(a) Draw a circuit schematic that implements the equation using only 3-input gates:

(b) Draw a circuit schematic for the same equation using only 2-input gates:
5. Considering the design of a simple 8-bit wide computer with:

- a single data operand register, ACC,
- a 64 X 8-bit ROM
- an 8-bit instruction register (IR)
- a 2-input 8-bit ALU with 2 bits of control (S) defined as follows:

![ALU Diagram]

Instructions are in the form of: \( \text{OP} \quad \text{ADDR} \) with 2 bits for the \( \text{OP} \) field and 6 bits for the \( \text{ADDR} \) field.

(a) [12 point] Three instructions; ADD, SUB, and NOR, correspond to the instruction opcodes 00, 01, 10 respectively and work as follows:

\[
\text{ACC} \leftarrow \text{ACC} \quad \text{OP} \quad \text{ROM}[\text{ADDR}]
\]

Draw a simple datapath for executing these instructions. Label all necessary control signals. Do not design the controller or the internal details of the ROM, ALU, and registers. Also, you do not need to show details of the instruction fetch or PC logic.

How many cycles do these instructions take to execute?

(b) [10 points] Draw another datapath with all the capabilities of the above one, plus with the ability to execute a new instruction with the following function:

\[
\text{ACC} \leftarrow \text{ROM[ROM[ADDR]]}
\]
How many cycles do these instructions take to execute?

6. [25 points] Consider the design of a 1-bit wide 4-element deep stack (FILO buffer), defined below:

On each cycle:
   if push = 1
       then the value presented on d is pushed on the stack
   else if push = 0
       then a value is popped from the stack and appears on d

Example:

<table>
<thead>
<tr>
<th>cycle</th>
<th>push</th>
<th>d</th>
<th>stack values</th>
<th>(at end of cycle)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>A</td>
<td>AXXX</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>B</td>
<td>BAXX</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>C</td>
<td>CBAX</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>D</td>
<td>DCBA</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>E</td>
<td>EDCB</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>E</td>
<td>DCBB</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>D</td>
<td>CBBB</td>
<td></td>
</tr>
</tbody>
</table>

Where X = "unknown value"

Using only the following set of primitives:
draw the schematic for a circuit that implements the stack.

Circuit Diagram: