Timing (10 Points)

1. Use the timing table below to find the propagation and contamination delays of
   the circuit shown below. Draw the short path and critical path on the logic circuit.
   All of the delays are in picoseconds.

<table>
<thead>
<tr>
<th>Gate</th>
<th>T_{pd}</th>
<th>T_{cd}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not</td>
<td>10</td>
<td>15</td>
</tr>
<tr>
<td>2-input AND</td>
<td>25</td>
<td>30</td>
</tr>
<tr>
<td>3-input AND</td>
<td>30</td>
<td>40</td>
</tr>
<tr>
<td>2-input OR</td>
<td>30</td>
<td>40</td>
</tr>
<tr>
<td>3-input OR</td>
<td>45</td>
<td>55</td>
</tr>
<tr>
<td>2-input NAND</td>
<td>15</td>
<td>20</td>
</tr>
<tr>
<td>3-input NAND</td>
<td>25</td>
<td>30</td>
</tr>
<tr>
<td>2-input NOR</td>
<td>25</td>
<td>30</td>
</tr>
<tr>
<td>3-input NOR</td>
<td>35</td>
<td>45</td>
</tr>
<tr>
<td>2-input XOR</td>
<td>40</td>
<td>60</td>
</tr>
</tbody>
</table>

2. Sequential Logic Timing (15 Points)

   2. Use the following timing specifications to find the minimum clock period of this circuit.

   Flip-Flops
   - T_{ccq} = 25 ps
   - T_{pcq} = 100 ps
   - T_{setup} = 100 ps
   - T_{thold} = 50 ps

   Combo Logic
   - T_{cd} = 20 ps
   - T_{pd} = 50 ps

   \[ T_{min} = 250 \text{ ps} \]

3. Sketch all of the timings on the waveform below. Use a period of 300 ps. Do any
   violations occur?

   There is a held violation.
4. Draw the logic described by this Verilog module. Include all information possible. What does this mystery block do?

```verilog
module MysteryBlock(
    input [7:0] A, B,
    input C,
    output [8:0] Z);
    wire [7:0] Bout;
    assign Bout = C ? ~B : B;
    assign Z = A + Bout + C;
endmodule
```

5. Draw the logic described by this Verilog module. Include all information possible. What is the name of this mystery block, and why would it be useful?

```verilog
module MysteryBlock(
    input Clock,
    input Reset,
    output Clock_2,
    output Clock_4);

    MysteryBlockStage Divider_1 (  
        .Clock(Clock),  
        .Reset(Reset),  
        .HalvedClock(Clock_2)  
    );

    MysteryBlockStage Divider_2 (  
        .Clock(Clock_2),  
        .Reset(Reset),  
        .HalvedClock(Clock_4)  
    );
endmodule

module MysteryBlockStage(
    input Clock,
    input Reset,
    output reg HalvedClock
);

    always @(posedge Clock, negedge Reset)
        if(!Reset) HalvedClock <= 0;
        else HalvedClock <= ~HalvedClock;
endmodule
```