Note: Do not use a calculator or computer to complete the following exercises. You must show all your work and put a box around your final answer to receive credit. Messy or unreadable submissions will receive no credit.

Homework will only be accepted at the beginning of class and all pages must be stapled together.

Total Points: 121

1. (0 points) How long did it take you to complete the homework? This will not affect your grade (unless omitted) but it helps gauge the workload for this and future semesters. If you do not answer this question you will get -5 points.

2. (30 points) Timing Analysis

Consider the circuit below that computes a registered four-input XOR function. Each two-input XOR gate has a propagation delay of 100 ps and a contamination delay of 55 ps. Each flip-flop has a setup time of 60 ps, a hold time of 20 ps, a clock-to-Q maximum delay of 70 ps, and a clock-to-Q minimum delay of 50 ps.

(a) (6 points) What is the minimum cycle time of the system?

(b) (6 points) What is the maximum operating frequency of the system?

(c) (6 points) Is the hold time constraint violated? Justify your answer.

(d) (6 points) What is the latency of the circuit (i.e. the time to make a single 4-input XOR calculation)? The answer should be in ps.

(e) (6 points) What is the throughput of the circuit (i.e. how many 4-input XOR calculations are completed per second)? The answer should be in # calculations/second.

Solution

(a) \( T_c \geq t_{pcq} + t_{pd} + t_{setup} = 70 + 3 \times 100 + 60 = 430 \) ps

(b) \( f = 1/T_c = 1/430p = 2.33 \) GHz

(c) For hold time constraint, \( t_{\text{hold}} < t_{ccq} + t_{cd} \Rightarrow 20 < (50 + 55) \). Since this is met, the hold time constraint is NOT violated.

(d) Latency = \( T_c = 430 \) ps

(e) Throughput = \( 1/430p = 2.33 \) giga calculations/second.
3. (36 points) Temporal Parallelism (Pipelining)

You implement the four-input XOR function using temporal parallelism (pipelining) by adding a register between the input and output of each XOR gate as shown in the circuit below. Each two-input XOR gate has a propagation delay of 100 ps and a contamination delay of 55 ps. Each flip-flop has a setup time of 60 ps, a hold time of 20 ps, a clock-to-Q maximum delay of 70 ps, and a clock-to-Q minimum delay of 50 ps.

![Circuit Diagram]

(a) (6 points) What is the minimum cycle time of the system?

(b) (6 points) What is the maximum operating frequency of the system?

(c) (6 points) Is the hold time constraint violated? Justify your answer.

(d) (6 points) What is the latency of the circuit (i.e. the time to make a single 4-input XOR calculation)? The answer should be in ps.

(e) (6 points) What is the throughput of the circuit (i.e. how many 4-input XOR calculations are completed per second)? The answer should be in # calculations/second.

(f) (6 points) Compare this circuit to the one from Question 2. Discuss advantages and disadvantages.

Solution

(a) This time, there is only a single XOR in the longest path between registers. \( T_c \geq t_{pcq} + t_{pd} + t_{setup} = 70 + 100 + 60 = 230 \) ps

(b) \( f = 1/T_c = 1/230p = 4.35 \) GHz

(c) For hold time constraint, \( t_{hold} < t_{ccq} + t_{cd} \Rightarrow 20 < (50 + 0) \). Since this is met, the hold time constraint is NOT violated.

(d) In order to compute \( Y \), the inputs have to travel through 3 registers (stages). Latency = \( 3 \times T_c = 3 \times 230 = 690 \) ps

(e) When the pipeline is full, an output can be calculated each clock cycle. Throughput = \( 1/230p = 4.35 \) giga calculations/second.

(f) The pipeline implementation has higher throughput so can calculate the output \( Y \) faster but this comes at a cost of longer latency than the original circuit. The added latency comes from pipelining sequencing overhead. In addition, the pipeline implementation has more hardware (extra registers). In general, the pipeline version is preferred since it can compute faster (almost twice as fast). However, this is only applicable when you need to compute \( Y \) many times otherwise it doesn’t make much sense to have a pipeline.

4. (55 points) FSM: 1965 Ford Thunderbird Tail Lights Computer Assignment

Full instructions on the website

http://www.ee.unlv.edu/~b1morris/cpe100/docs/comp02.pdf
(a) (0 points) How many hours did you spend on the lab?
(b) (10 points) Full FSM design, including a completed state transition diagram for your FSM and schematic.
(c) (20 points) A printout of the Quartus II schematic.
(d) (20 points) A print out of the ModelSim simulation of the circuit. Make sure you test the circuit completely. Show the result for a reset, left signal, and right signal. Your signals should be printed in the following order: \(clk, reset, left, right, LC, LB, LA, RA, RB, RC\).
(e) (5 points) A brief paragraph (2-4 sentences) highlighting what you learned on this lab assignment. List any difficulties you encountered.

Solution

(a) XX hours (hopefully only 2-3)
(b) The FSM design comes from Homework 8.

![State Transition Diagram]

<table>
<thead>
<tr>
<th>State</th>
<th>Encoding</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>(S_0)</td>
<td>0 0 0 0 0</td>
<td>(LC = 0) (LB = 0) (LA = 0) (RA = 0) (RB = 0)</td>
</tr>
<tr>
<td>(S_1)</td>
<td>0 0 1 0 0</td>
<td>(LC = 0) (LB = 0) (LA = 0) (RA = 0) (RB = 0)</td>
</tr>
<tr>
<td>(S_2)</td>
<td>0 1 1 0 0</td>
<td>(LC = 0) (LB = 0) (LA = 0) (RA = 0) (RB = 0)</td>
</tr>
<tr>
<td>(S_3)</td>
<td>1 1 1 0 0</td>
<td>(LC = 1) (LB = 1) (LA = 0) (RA = 0) (RB = 0)</td>
</tr>
<tr>
<td>(S_4)</td>
<td>0 0 0 1 0</td>
<td>(LC = 0) (LB = 0) (LA = 1) (RA = 0) (RB = 0)</td>
</tr>
<tr>
<td>(S_5)</td>
<td>0 0 0 1 1</td>
<td>(LC = 0) (LB = 0) (LA = 1) (RA = 0) (RB = 1)</td>
</tr>
<tr>
<td>(S_6)</td>
<td>0 0 0 1 1</td>
<td>(LC = 0) (LB = 0) (LA = 1) (RA = 0) (RB = 1)</td>
</tr>
</tbody>
</table>

Next state equations and output:

\[
S'_5 = \bar{S}_5 S_4 \\
S'_4 = \bar{S}_5 S_3 \\
S'_3 = \bar{S}_5 S_3 + \bar{S}_3 \bar{S}_2 L \\
S'_2 = S_2 \bar{S}_0 + \bar{S}_3 \bar{S}_2 R \\
S'_1 = S_2 \bar{S}_0 \\
S'_0 = S_1 \bar{S}_0 \\
L_A = S_3 \\
L_B = S_4 \\
L_C = S_5 \\
R_A = S_2 \\
R_B = S_1 \\
R_C = S_0 \]
(c) Note: There is a bubble on the CLRN input to the DFF block meaning it is active low. The schematic uses an inverter on the Reset signal to have normal active high logic. For more information search the Intel Quartus information pages [link].

(d) Below is the timing waveform diagram for the simulated circuit. This was generated using a .do script [link] file which can make generating the clock and other signals much easier. Note that for ease, the state of the system is shown in a single bus (which also corresponds directly to the output) for more easy visualization.

(e) Each person will have a different answer. Hopefully you were able to realize the design and simulate it. Notice that our design, L and R should not have been able to be active at the same time. In simulation you can see (at the end of the waveform) that the behavior is unexpected when this situation does occur.
5. (0 points) Teaching Evaluation

Reminder, you should have received an email to your RebelMail account asking to give an evaluation of the course. Please do take the ~ 3 minutes required to enter in your assessment. This provides feedback which is helpful for improving further iterations of the course.