Assignment 2: Sequential Circuits

Due by: Friday 02/23/2018 10:00 PM

Collaboration: None

Grading: Correctness 100%

Overview

The second assignment is mostly about sequential circuits, i.e. circuits that combine combinational circuits with latches, flip-flops, and registers. It's possible for you to submit the assignment without drawing a single circuit, but it's **not recommended** that you do that; most of the diagrams you'll need are rather "blocky" and you can do them just fine as ASCII art. If you want to include actual circuit diagrams, please either "draw" them as ASCII art or include a PDF file with clearly labeled diagrams.

Problem 1: RS Latches using NAND Gates (20%)

In lecture we discussed RS latches built from NOR gates in some detail. Here you'll repeat some of that process for latches built from NAND gates instead:

1. For the basic NAND RS latch, carefully derive the circuit's behavior depending on the values of R and S. Your main goal is to identify the values of R and S that will result in the

HOLD, SET, RESET, and ILLEGAL states for the NAND RS latch, but you should discuss the NAND RS latch more generally.

2. Explain the additional circuitry necessary to turn NAND RS latch into an edge-triggered D-type flip-flop. Make sure you discuss in what sense the resulting flip-flop *differs* from the one built out of NOR RS latches. Do they have the same number of gates? Do they have the same "propagation delay" as it were? Anything else?

After completing this problem, you may have to "erase" the NAND stuff from your brain again because the remaining problems assume that you're using latches and flip-flops based on NOR gates, just as we did in lecture.

Problem 2: Shift Register (40%)

Design a 4-bit shift register out of edge-triggered D-type NAND flip-flops. The shift register has 4 inputs and 4 outputs for the data stored in the register (4 parallel lines to read/write the bit pattern in the register). The outputs should be available continuously and should represent the current value stored in the register. The inputs should only be sampled (and transferred into the register) when an external load signal *LD* is high.

If the *LD* signal is low, then the shift register will perform either a left shift or a right shift on each clock pulse, depending on an external *DIR* signal. If *DIR* is low, the

register shifts to the left (toward the higher bit value), if *DIR* is high, the register shifts to the right (toward the lower bit value). A bit that would "shift out" is "shifted back in" at the opposite end of the register. An example (assuming LD and DIR are low!) may make this clearer (each arrow represents a clock pulse):

```
... --> 0001 --> 0010 --> 0100 --> 1000 --> 0001 --> ...
```

Hint 1: All these options regarding where the inputs for your flip-flops come from (actual input lines, right or left neighbors) probably requires some kind of multiplexer. You can use an abstract multiplexer (black box), you don't have to include the gate-level multiplexer circuitry.

Hint 2: It's a **really** good idea to design one "cell" of your shift register first; then you can use that "cell" as a building block for the register as a whole. Your presentation of the circuit will be **much** improved by this.

Problem 3: Arithmetic Logic Unit (ALU) (20%)

Design a 4-bit ALU that supports the operations AND, OR, NOT, and ADD on two 4-bit inputs; there is a single 4-bit result. Explain the **structure** of the 4-bit AND, OR, NOT, and ADD circuitry, but don't worry about drawing it in detail. Explain how you combine everything into one ALU using either multiplexers/demultiplexers or encoders/decoders and discrete gate-level circuitry. You may have to figure out (and

should explain in detail!) how you can use multiplexers *in parallel* to route 4-bit quantities around your ALU as needed.

Problem 4: Reliability of Memory (20%)

You are manufacturing random-access memory (RAM) modules for computer systems. You build these modules out of edge-triggered D-type master-slave flip-flops which in turn rely on D-type latches as we discussed in lecture. (Note that this is a rather fantastic fiction, but this is a homework problem after all, nothing more.)

Assume that you can manufacture D-type latches with 99.999999% reliability: whenever you manufacture a D-type latch, the chance for it operating correctly "forever" is 99.999999%.

Ignoring all other possible problems, what is the chance that a 1 MB (what I call one Megabyte, also known as mebibyte these days) memory module will operate correctly?

If we want to manufacture a 4 GB (what I call four Gigabytes, also known as gibibytes these days) memory module that has a 99% chance of operating correctly,

what kind of reliability do we have to expect of our D-type latch manufacturing processes?

Deliverables

Please turn in a `pdf` of your assignment to Gradescope.

Grading

For reference, here is a short explanation of the grading criteria; some of the criteria don't apply to all problems, and not all of the criteria are used on all assignments. **Packaging** refers to the proper organization of the stuff you hand in, following the guidelines for Deliverables above. **Style** refers to either to programming style if we are talking about a programming problem, or to the clarity and readability of your solution for a written problem.