cs281: Introduction to Computer Systems

Lab06 – Sequential Circuits, Part II

1 Overview and Objectives

In the last lab, through the prelab and the lab itself, we were introduced to the second major branch of digital logic design: *sequential circuits*. We learned the design process steps of

1. Solve the problem with a Finite State Machine, whose states, in general, are annotated with the bits that are used for **system output**, and whose transitions are taken based on the **system input**.
2. Use a collection of clocked 1 bit memory devices, called *Flip Flops* to remember what ”state” of the FSM the system is in. This requires, for each state of the FSM, to have a unique bit pattern that is the *state assignment* from the FSM state to the 0 and 1 values of the flip flops.
3. Using the techniques of *combinational circuit*s, design a **NextState** circuit, whose input comes from both the current saved values of the flip flops (the set of Q_i, for subscript i the indices of the set of flip flops), and from the system inputs that would be on the transitions of the FSM. The output of the NextState circuit are the control values used to transition the bits of the flip flops.
4. Design an *Output* circuit, which uses, as input, the state (the Q_is) to yield the bits of the system output desired, as given in the annotations of the FSM. We learned that this output circuit does *not* require any information/bits from the system input.
5. For each of the outputs of the NextState circuit and for each of the outputs of the Output circuit, we arrive at simplified Boolean expressions, typically through the use of K-maps.
6. The circuit is realized by simply putting together the above pieces with a clock and using feedback of the outputs of the flip flops routed to the NextState circuits and the Output circuits, and the outputs of the NextState circuits routed to the control inputs of the flip-flops.

Objectives: The objective of this Lab is to reinforce and practice the above design process, building on the last lab. In particular, we want:

- A FSM that requires more than four states, and so requires more than two flip flops.
- With n flip flops, a total of 2^n patterns representing states are possible. But if our FSM has $2^{n-1} < S < 2^n$ states, we end up with rows in our NextState circuit truth table filled with Don’t Care X entries. Our design needs to properly handle such Don’t Care entries.
- Learn about a simpler, but less capable flip flop – the D flip-flop. The advantage of the D flip flow is that it has only *one* control input (named D), which is the 0 or 1 value for the flip-flop to store in its one-bit memory. The disadvantage of the D flip flop is that we lose the control-ability of a ”command” to stay at the same value, or to toggle the value.
- Use logisim to build subcircuits. In this case, we want a subcircuit for the NextState circuit to help keep our top-level design (in main) clearer.

2 Problem

We want to build a "modulo-five" counter circuit with three bits of output C_2, C_1, C_0. This will be a clocked circuit that takes system input X in addition to the clock. On a transition (rising edge of the clock), and if X has value 1, the circuit transitions as follows:

$$000 \rightarrow 001 \rightarrow 010 \rightarrow 011 \rightarrow 100 \rightarrow 000 \rightarrow 001 \rightarrow \cdots$$

effectively doing a binary count from 0 up to 4, back to 0, up to 4, etc.

Any time the system input X is 0, the "counter" is paused, remaining at its last state.

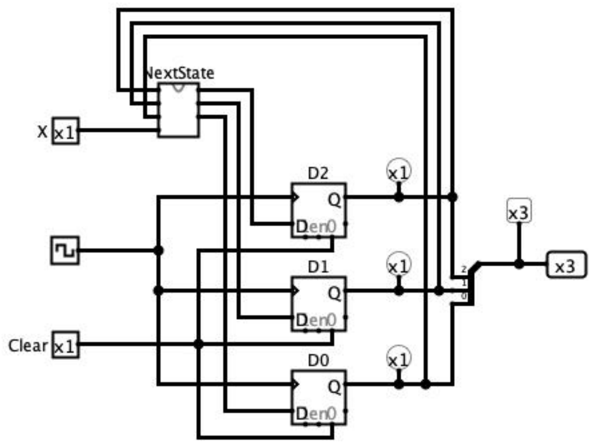
Recall that we **always** want to use subscripts in a consistent manner, so C_2 refers to the *most significant bit* of this counter, while C_0 refers to the least significant bit of this counter.

Note that, depending on how you design this circuit, and, in particular, the state assignment, you may have no Output circuit at all, and can simply yield the output directly from the state bits of the flip flops.

3 Implementation

We want you to use logisim to realize this circuit and, further, to use hardware decomposition in much the same way we use functional decomposition to help solve a problem. In particular, we want you to use a **subcircuit** for the NextState design, where this circuit has input pins that will come from the current state (Q_i) along with the system input (X in this case). By convention, we put the input pins on the left of the subcircuit's page, and our placement in the subcircuit determines where we see the input pins when we place the circuit at a higher level. The subcircuit has output pins for each of the controls used for the flip flops. These, by convention, are placed on the right side of the subcircuit's page.

Pictured below is my top level main circuit, and you can see the use of D flip flops and the use of a NextState subcircuit.



Note also, on the top level, that I have a pin that allows me to clear the state back to 0 asynchronously from the clock, and that I use probes (available under 'Wiring") to help see what values are on my C_i, as well as a wire splitter/joiner to get a three bit aggregate from the three individual wire bits.

4 Lab Report

This lab report will be slightly different. It will be less about a big project and more about the design of the individual parts. Please submit a report with the following sections:

1. PURPOSE: write a small paragraph that summarizes the purpose of this lab.
2. FSM: Describe the device you are building. Show the steps and all diagrams as you construct the FSM and eventually the circuit. All diagrams and table should be included in the lab report. Diagrams:
    - FSM circle diagram
    - Table translating FSM states to Flip Flop states
    - Truth table showing next-state calculations.
    - K-maps showing minimizations.
    - Boolean expressions from the K-maps.
    - Logisim diagrams, both for the top level, and the subcircuit.

You will submit **both the logisim file and a pdf of your report**.