Digital Systems (Sheet #2)

Marius Gavrilescu

4.1
1. As in the lecture notes, xor the output of the D-type flip flop with the input and feed the result into the D-type flip-flop.
2. Start from the circuit described in the next part of this question. The binary incrementer is made of many half adders, which contain *xor* gates (to produce the output bit) and *and* gates (to produce the carry bit). We can combine a *xor* gate followed by a D-type flip-flop to form a T-type flip-flop. This yields a circuit made of many T-type flip-flops and *and* gates.
3. Connect the output of the binary incrementer to a row of D-type flip-flops, and their outputs to the inputs of the binary incrementer.
4. As seen in part 2, the counter using T-type flip-flops is built by taking the counter using D-type flip-flops and combining D-type flip-flops with *xor* gates to form T-type flip flops.

4.3 Use a full adder. Connect the circuit's inputs to the full adder, and the full adder's output bit to the circuit's output. Then connect the full adder's carry out to a D-type flip-flop, whose output is then fed into the full adder's carry in.

This is a Mealy machine.

The easiest way to make it a Moore machine is to buffer the inputs with two D-type flip-flops. Hence the output will only depend on the internal state (values in the three flip-flops) and not on the input. One disadvantage is that we waste a cycle this way, another is that the resulting circuit is more complicated.

To add a succession of pairs of numerals, just feed two zeroes after every pair and skip the resulting output bit.

4.6 Have two states: active (waiting for someone to enter/leave), cooldown (waiting for sensors to reset). Represent these as one variable u, which is 1 for active and 0 for cooldown.

Use two D-type flip-flops to remember the previous values a' and b' of a and b. Whenever the circuit is active and a and b are both true we look at the previous values to see which signal to send and set the circuit into cooldown. Whenever both sensors return false set the circuit to active.

Thus we have $up = u \wedge (a \wedge b) \wedge a'$, $down = u \wedge (a \wedge b) \wedge b'$ and $u' = (\neg a \wedge \neg b) \vee (u \wedge \neg up \wedge \neg down)$. Here u' is the new value of u, while a' and b' are old values of a and b.

If both sensors are triggered at the same time, a and b will both be 1 while a' and b' will be 0, so the circuit will send no signal.

4.8 The states progress in the order 000..0, 100..0, 110..0, ..., 111..11, 011..11, 001..11, ..., 000..01, 000..00. 2n states in total.

We can make a circuit with an average of one and gate and one not gate per state, as follows:

For the all zeros state, do $\neg a_1 \wedge \neg a_n$. For the all ones state, do $a_1 \wedge a_n$. For the first states do $a_i \wedge \neg a_{i+1}$ for all i between 1 and $n-1$. For the last states, do $\neg a_i \wedge a_{i+1}$ for all i between 1 and $n-1$.