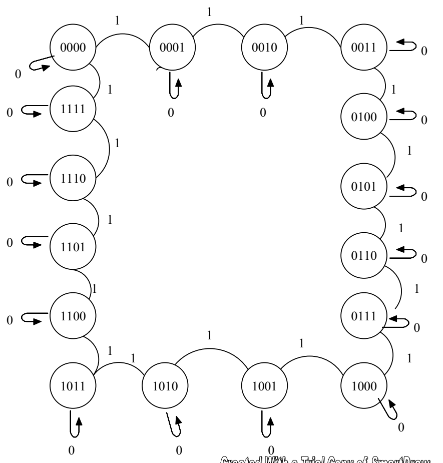
Designing a 4-bit binary synchronous counter with D flip-flops

By Darren Wiessner
Extra Credit assignment for exam 2

The first thing we need to do in designing a 4-bit synchronous counter with D flip-flops is to understand what a synchronous counter is. Synchronous counters differ from ripple counters in that a clock pulse is applied to all flip-flops simultaneously. With this information, we can make our first assumption about our design. We will simply hard wire the clocks all to the same clock pulse. The next step we need to take is understanding where each state goes. We do this by constructing a finite state machine (see figure 1). Doing this allows us to easily create a state table (see table 1).

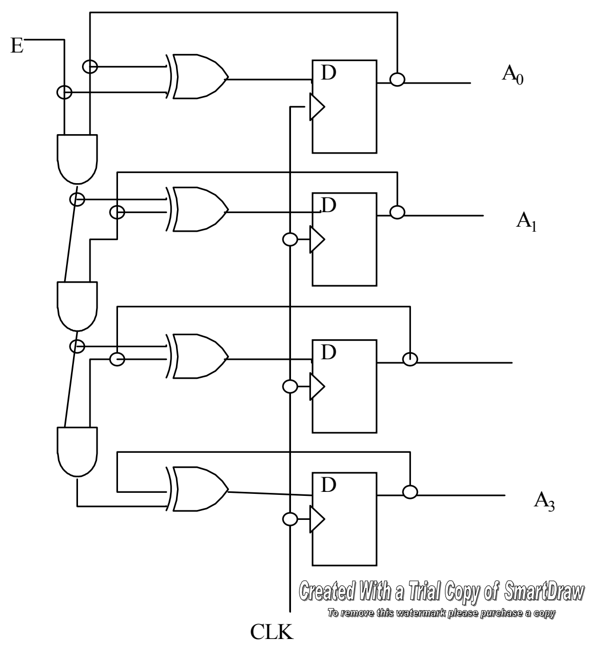
State Table for counter (table 1)

Present State	Input	Next State	Flip-flop Inputs			
			D_3	D_2	D_1	D_0
0000	0	0000	0	0	0	0
0000	1	0001	0	0	0	1
0001	0	0001	0	0	0	1
0001	1	0010	0	0	1	0
0010	0	0010	0	0	1	0
0010	1	0011	0	0	1	1
0011	0	0011	0	0	1	1
0011	1	0100	0	1	0	0
0100	0	0100	0	1	0	0
0100	1	0101	0	1	0	1
0101	0	0101	0	1	0	1
0101	1	0110	0	1	1	0
0110	0	0110	0	1	1	0
0110	1	0111	0	1	1	1
0111	0	0111	0	1	1	1
0111	1	1000	1	0	0	0
1000	0	1000	1	0	0	0
1000	1	1001	1	0	0	1
1001	0	1001	1	0	0	1
1001	1	1010	1	0	1	0
1010	0	1010	1	0	1	0
1010	1	1011	1	0	1	1
1011	0	1011	1	0	1	1
1011	1	1100	1	1	0	0
1100	0	1100	1	1	0	0
1100	1	1101	1	1	0	1
1101	0	1101	1	1	0	1
1101	1	1110	1	1	1	0
1110	0	1110	1	1	1	0
1110	1	1111	1	1	1	1
1111	0	1111	1	1	1	1
1111	1	0000	0	0	0	0



Finite State Machine for a 4-bit synchronous counter
Figure 1

Once all this is done, the easiest method is to look at the state table. You can easily see a pattern developing. When D_0 and input are equal, the next state of D_0 will equal 0. When they are not equal, the next state equals 1. This is an xor gate. So, $D_0 = A_0$ xor E. Now the D_1 is a little harder to see, but look again at the pattern. D_1 equals 1 when $A_0E = 1$ and $A_1 = 0$, or vice versa. Again, this is an xor gate. So we develop the formula, $D_1 = A_1$ xor (A_0E). Now looking at the pattern being developed by the formulas will give us a hint as to what D_2 should equal. We can see that D_2 equals 1 when $A_1A_0E = 1$ and $A_2 = 0$, or vice versa. So for formula 3, we get $D_2 = A_1A_0E$ xor A_2. For our final formula, we can see from the pattern what our D_3 will probably equal. Looking at the state table, we see that it is correct. $D_3 = A_2A_1A_0E$ xor A_3. Now we can design our circuit (see figure 2).


E
D
A_0
A_1
A_3
CLK