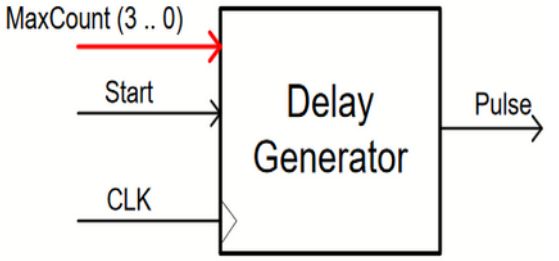
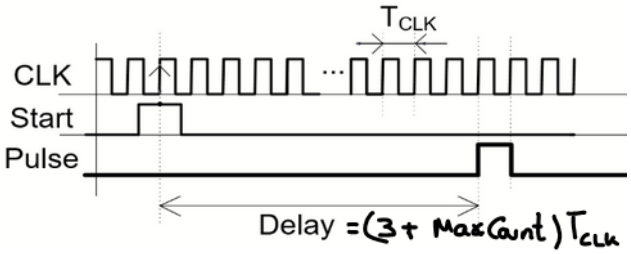
Example solution for the problem 1





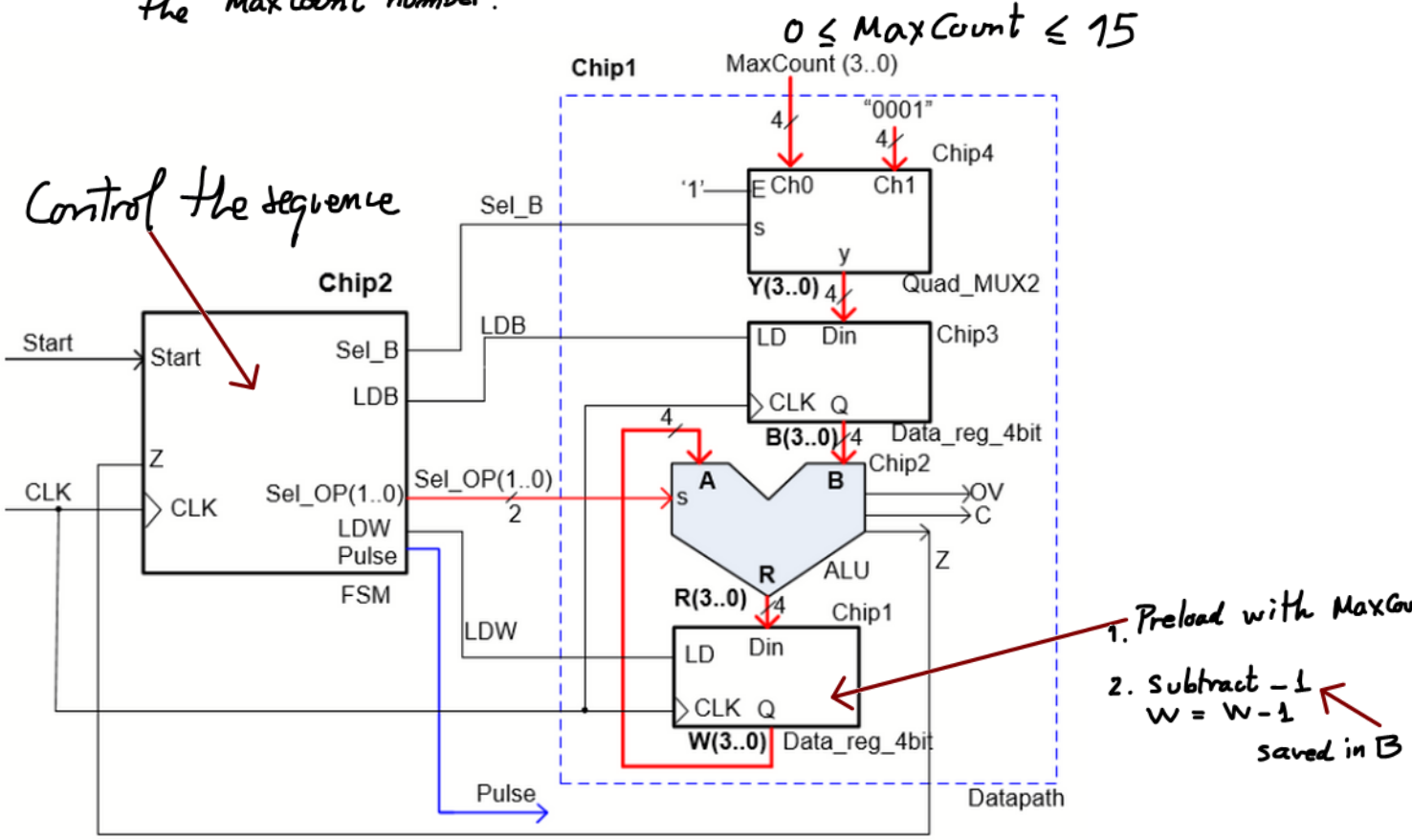
The idea is to generate a single pulse of T_{CLK} duration after clicking the start button and when the programmable delay has ended.

The minimum delay will be $3 \cdot T_{CLK}$ when MaxCount = 0
The maximum delay will be $(3 + 15) \cdot T_{CLK} = 18\, T_{CLK}$

and so, to get longer delays it is necessary simply to get a larger MaxCount vector, which, at the same time implies resising all the datapath blocks but keep the same FSM control unit.

Inspecting the architecture and the state diagram we see that the additional 3 CLK periods are necessary to start downcounting (subtracting) from the Chip1 W(3_0) data register once it has been preloaded with the MaxCount number.

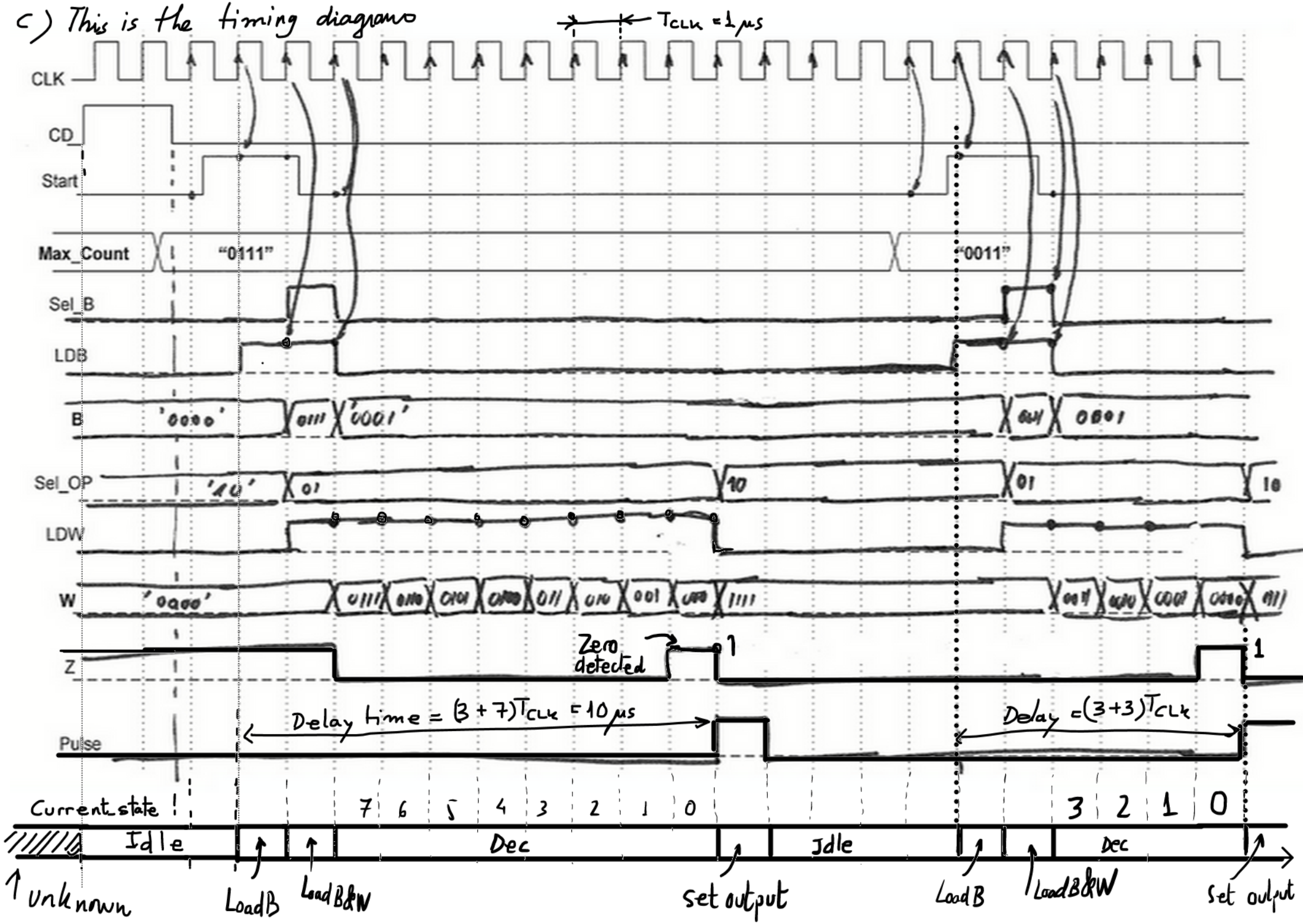
$0 \leq MaxCount \leq 15$



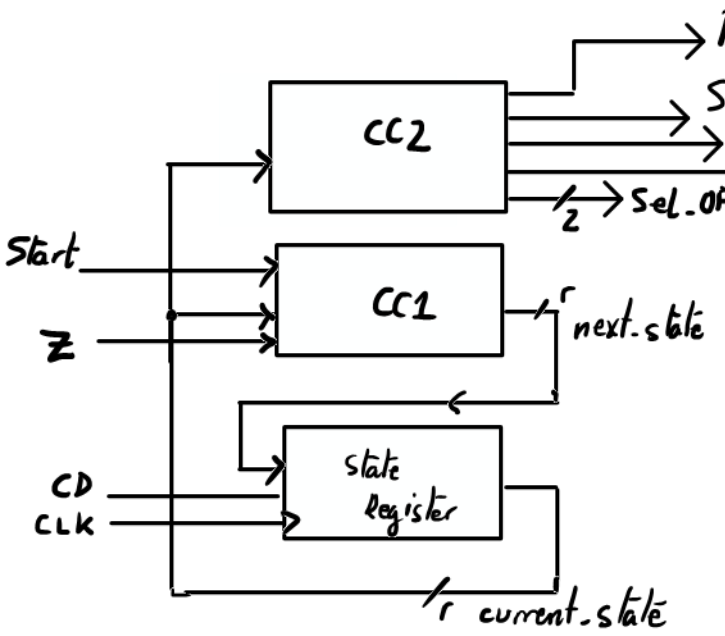
a) Top circuit Delay.vhd; FSM.vhd; Datapath.vhd, Data_Reg_4bit.vhd, ALU.vhd, Quad_MUX2.vhd (assuming that all the components are solved in a single file) → 6 VHDL files

b) FSM → one hot encoding → 5 D_FF ; Data_reg_4bit → 4 DFF ⇒ 13 D_FF

c) This is the timing diagram



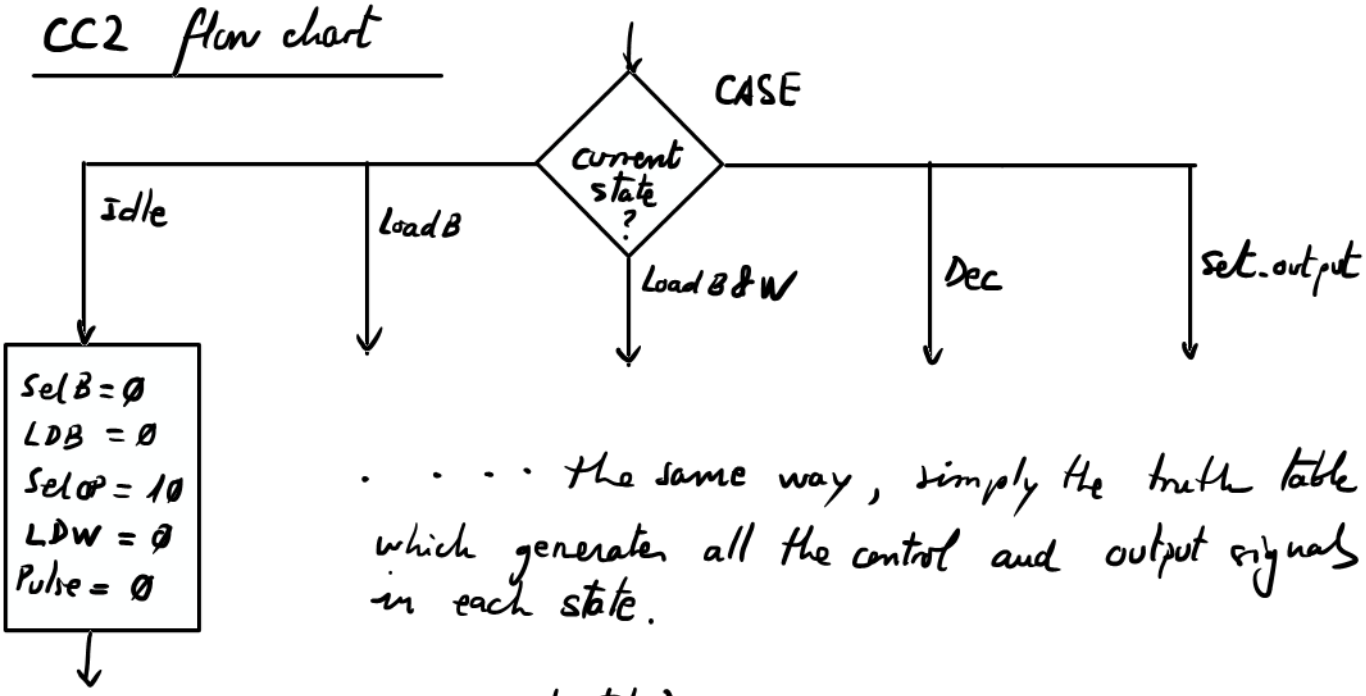
d) The FSM is organised in 3 blocks : SR, CC1, CC2



Z is the status signal that comes from the datapath to indicate that a given operation is giving a zero result.

If the FSM is coded in 'one-hot' we need r = 5 D_FF working in parallel in the state register.

e) CC2 flow chart



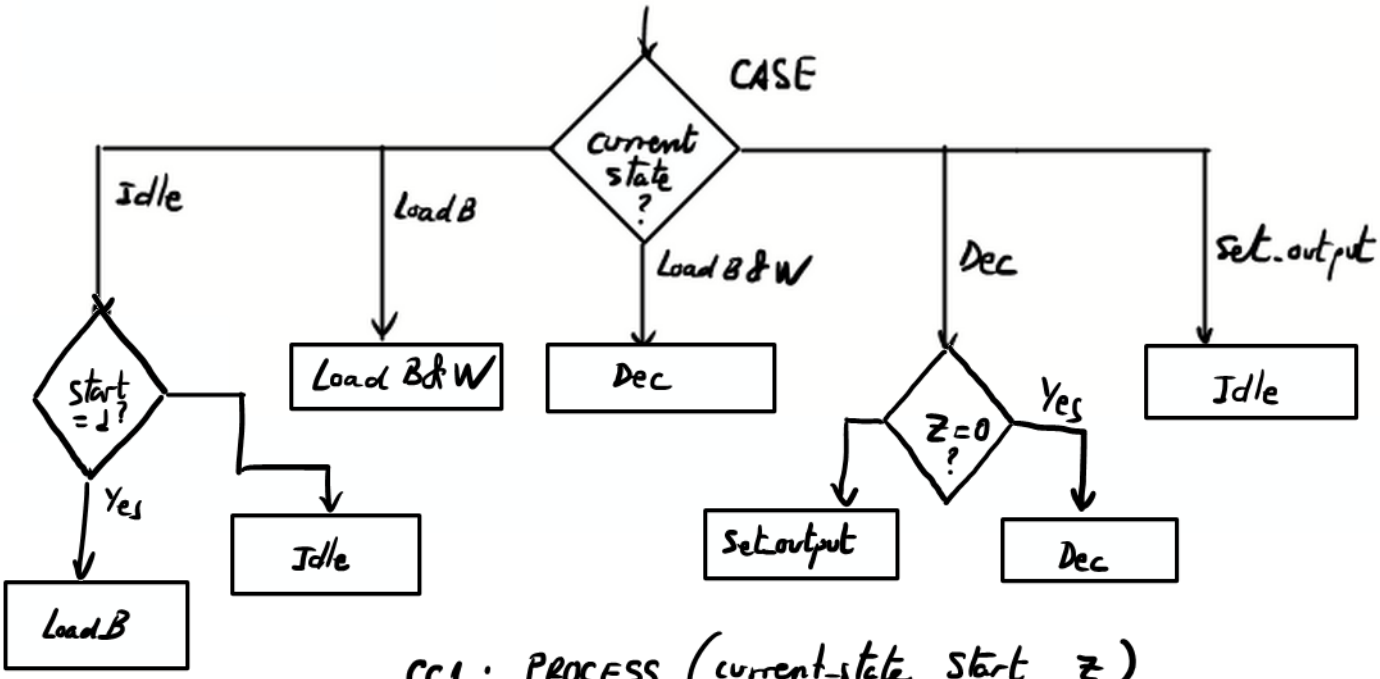
. . . . the same way, simply the truth table which generates all the control and output signals in each state.

```
CC2 : PROCESS (current_state)
BEGIN
   CASE current_state IS
When LoadB =>
      SelB <= '0';
      LDB <= '1';
      LoadB&W <= "10";
      LDW <= '0';
      Pulse <= '0';
!
```

f) Flow chart for the CC1 truth table → generate the next state to go

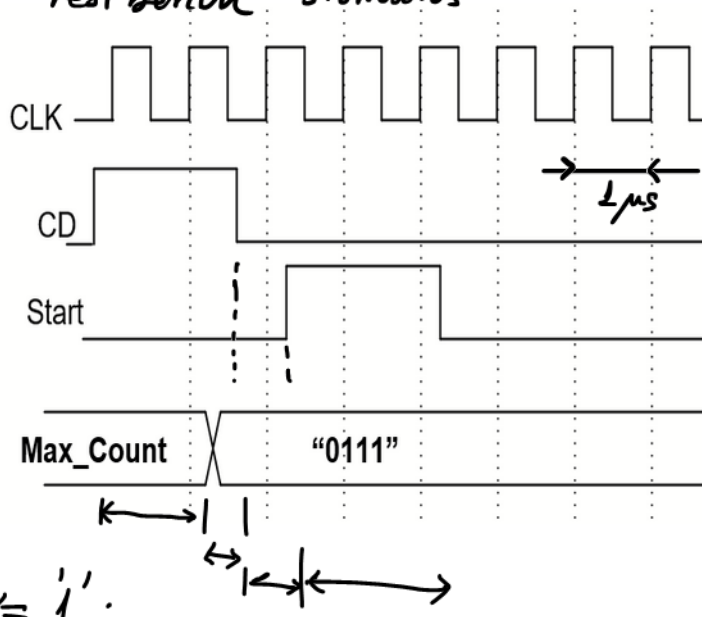
start	z	current_state	next_state
0	X	Idle	Idle ↻ loop
1	X	Idle	LoadB → transition with a condition
X	X	LoadB	LoadB&W →
X	X	LoadB&W	Dec →
X	0	Dec	Dec ↻
X	1	Dec	Set_output →
X	X	Set_output	Idle →

This table implements all the arrows in the state diagram




```
CC1 : PROCESS (current_state, start, z)
BEGIN
    CASE current_state IS
        When Idle =>
            IF (start = '1') then
                next_state <= LoadB;
            ELSE next_state <= Idle;
            END IF;
        :
    END CASE
END PROCESS CC1;
```

g) Some testbench stimulus

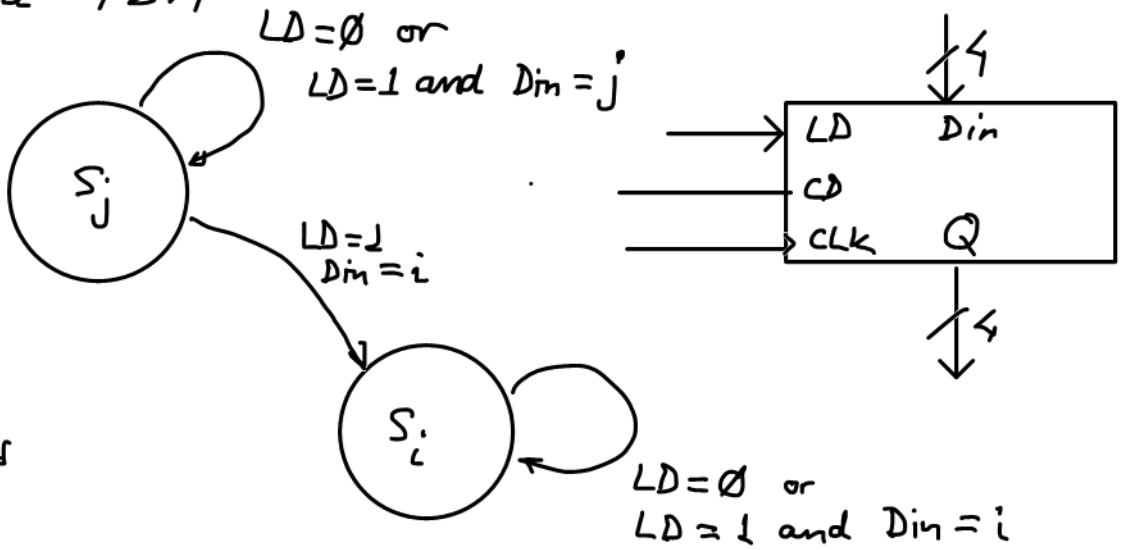


clk_period = 1 µs;

```
clk_process : process
  clk <= '0';
  wait for clk_period/2;
  clk <= '1';
  wait for clk_period/2;
end process;
```

```
CD <= '1';
Start <= '0';
MaxCount <= "0011";
Wait for 1.3 * clk_period;
MaxCount <= "0111";
wait for 0.5 * clk_period;
CD <= '0';
Wait for 0.75 * clk_period;
Start <= '1'
Wait for 2.55 * clk_period;
wait;
```

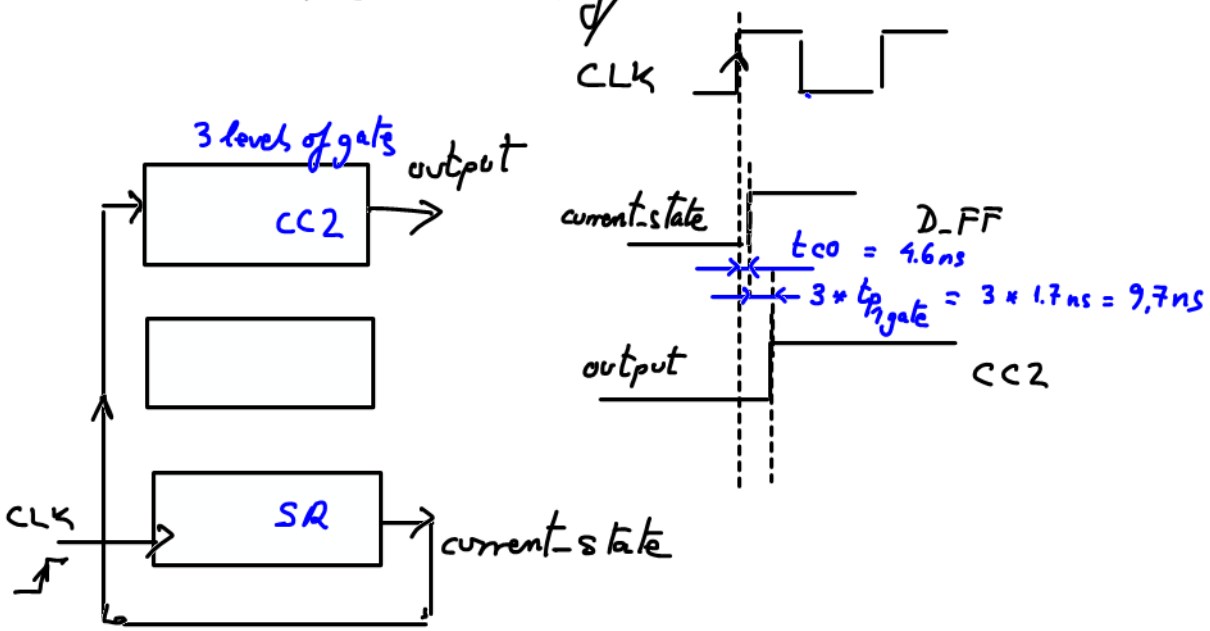
h) This section is about the design of the Data_reg_4bit as a FSM



16 states
4 DFF

⇒ See example solution in P7 digsys.upc.edu

i) The resolution of the system is related to the architecture and the technology



In this application the CLK frequency can be 103 MHz as maximum, and the minimum delay that can be programmed is $(3+0)\cdot T_{CLK} = 29,1\,ns$

↑ Max_Count = 0

So, the time resolution imagining that another complete CLK period is required to detect the start button is

$$4 * T_{CLK} = 38,8\,ns$$