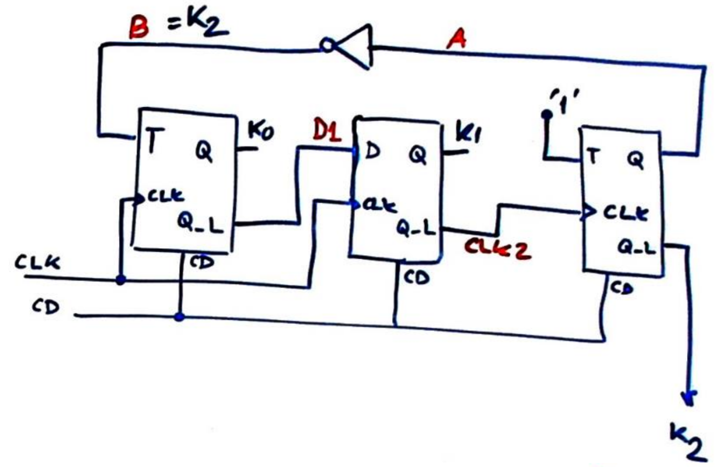
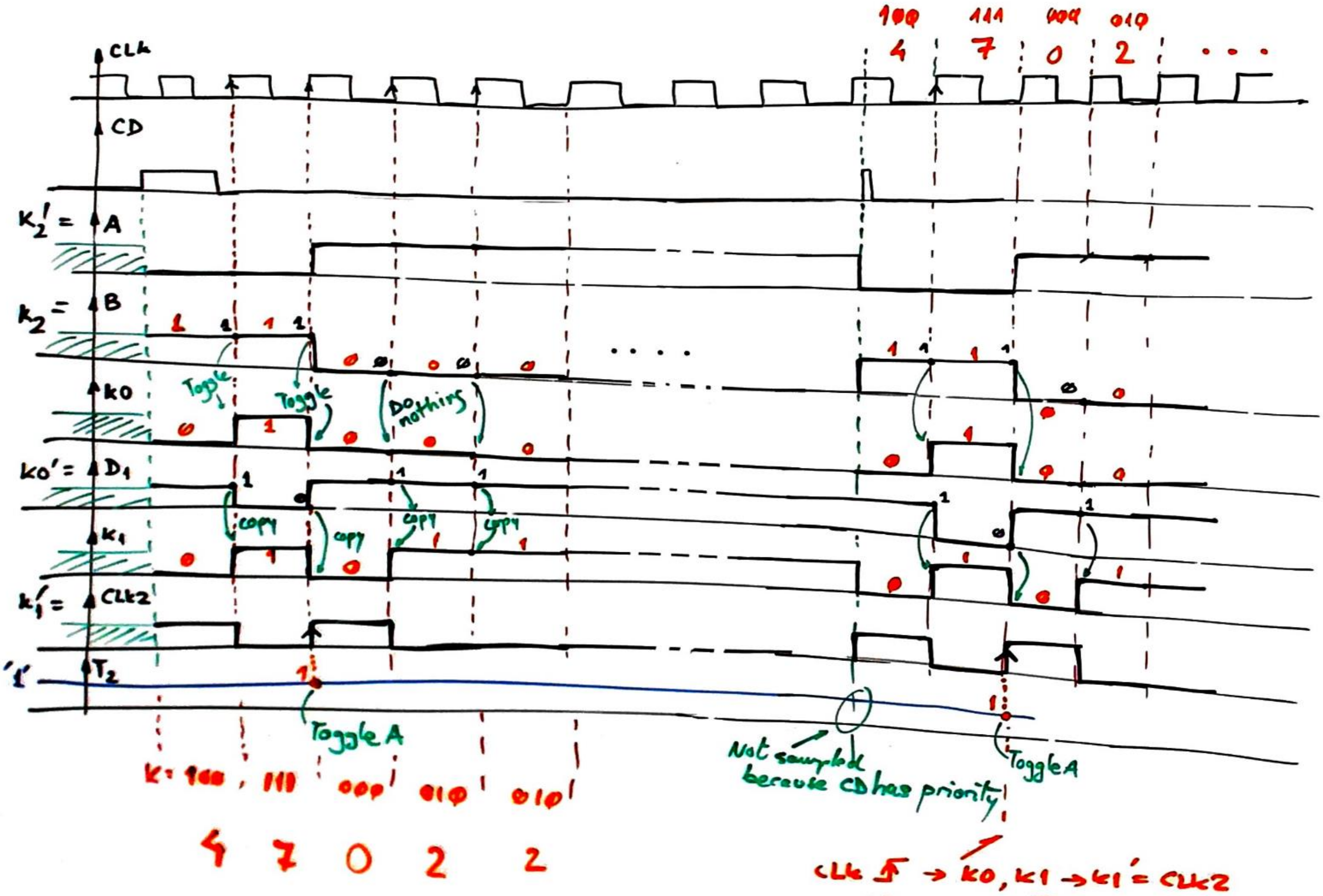
2223Q1 Exam 2 solution ideas

1 Example timing diagram.

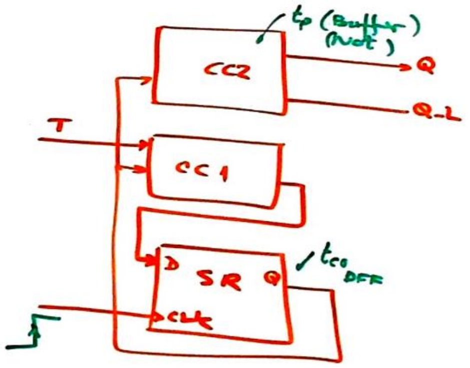




CLK ↑ → k0, k1 → k1' = CLK2

if CLK2 ↑ → Toggles A → B

$$t_{p_{CLK\ to\ B}} = t_{co_{1,2}} + t_{co_{3}} + t_p =$$

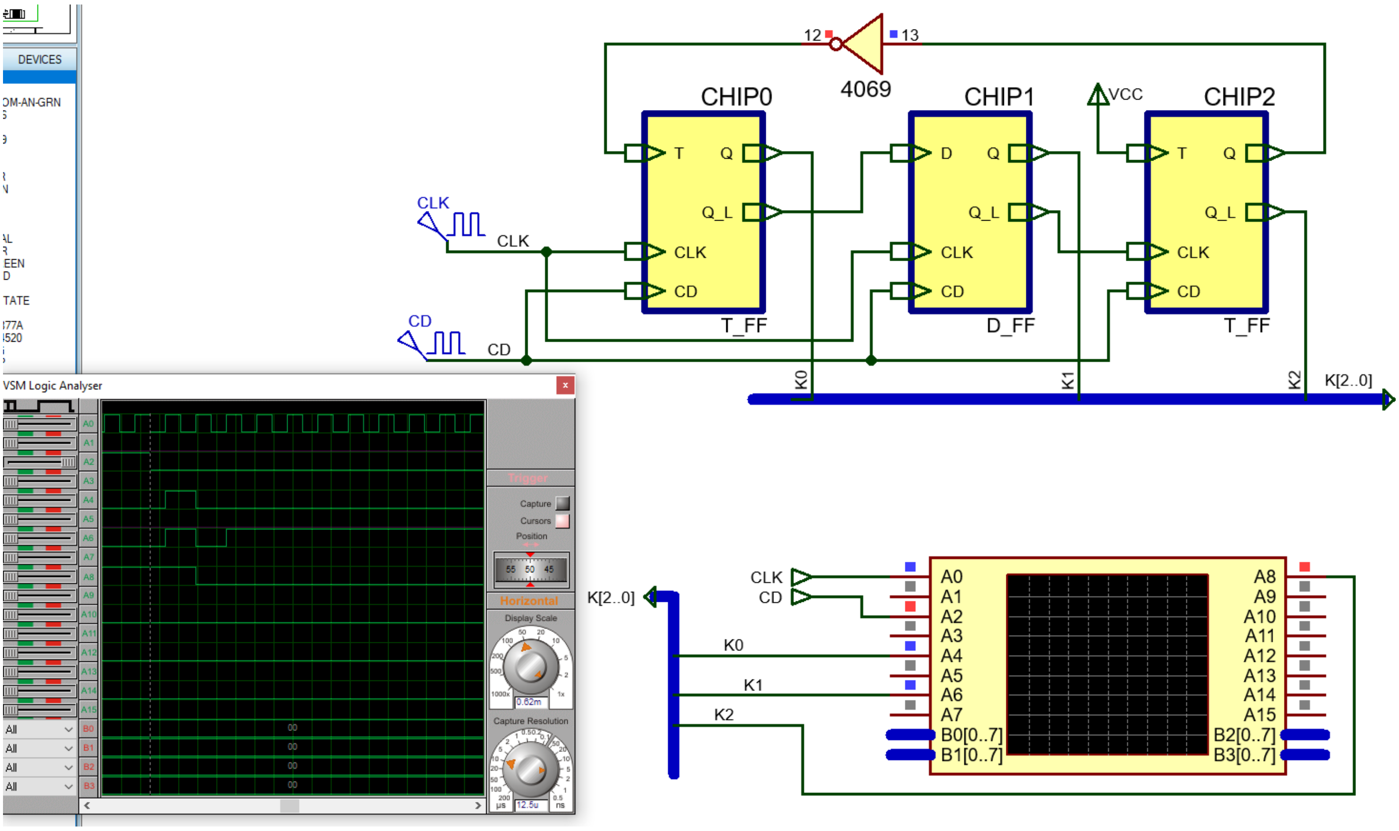


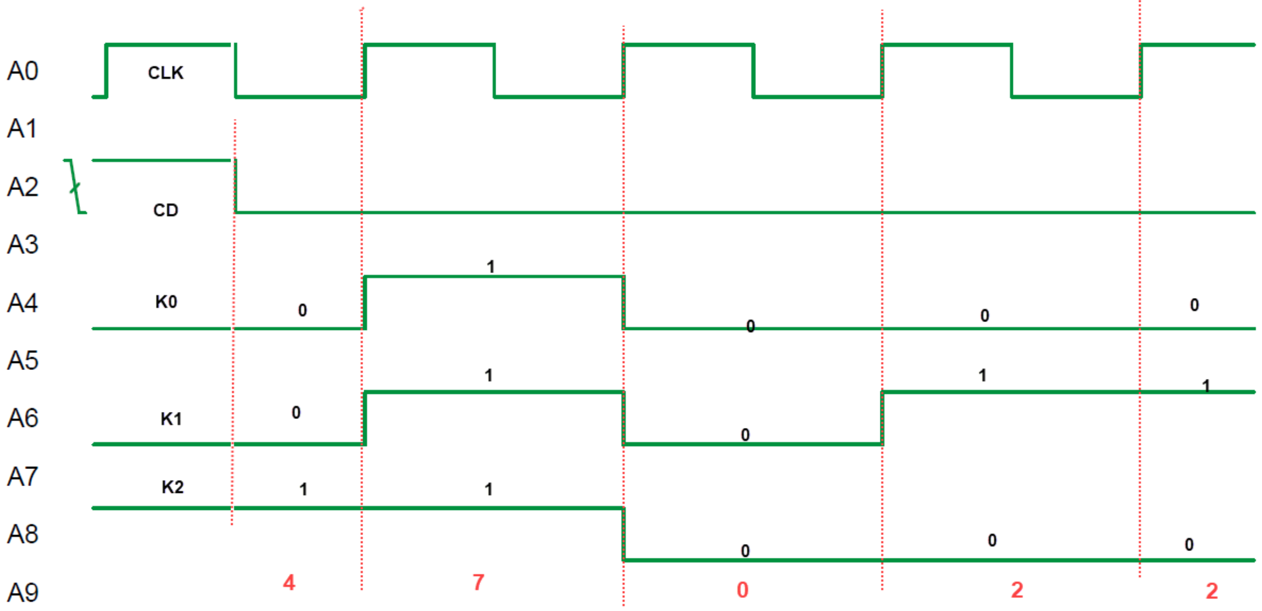
$$t_{co} = t_{co_{DFF}} + t_{p_{gate}}$$

$$t_{p_{CLK\ to\ B}} \simeq 3\cdot \underbrace{t_{co_{DFF}}}_{2.1nF} + 2\underbrace{t_{p_{gate}}}_{2.5nF} = 10{,}5 ns$$

$$f_{max} < 95 MHz$$

Capturing the circuit in Proteus we can develop this circuit and verify the waveforms





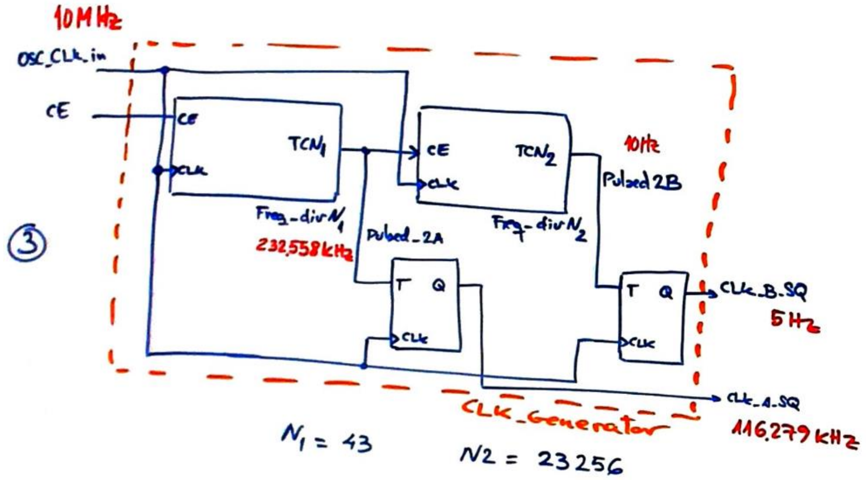
2. Chip3 has a different CLK, and this connection complicates the analysis and design. Besides, practically all common applications, like this one, a burst generator of 3-bit digital pattern consisting of codes “4 – 7 – 0 – 2” can be designed using the standard synchronous FSM. Instead of CD, a trigger button can generate such sequence.

D_FF.vhd, T_FF.vhd, Top_circuit.vhd are required to develop and test this device using FPGA EDA tools. Other examples in P5.

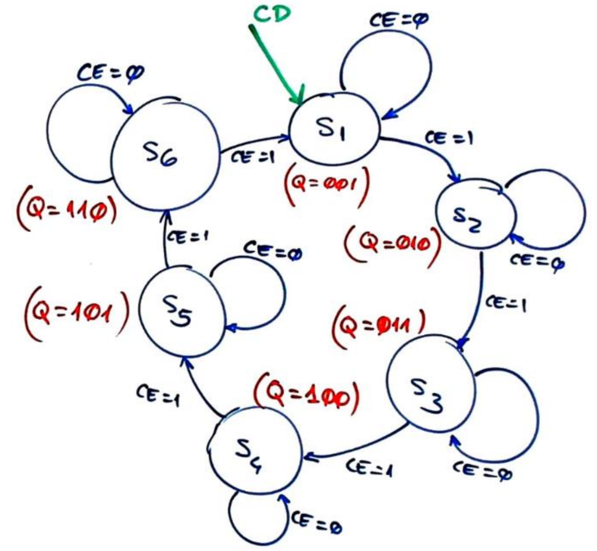
When CLK rising edge, K0 and K1 changes, and so, K1’ = CLK2, and from this new rising edge, Chip3 switches A, and finally B.

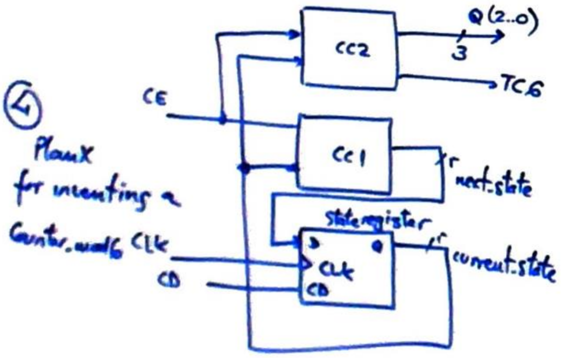
Problem 2

3. CLK_Generator



4. Counter_mod6



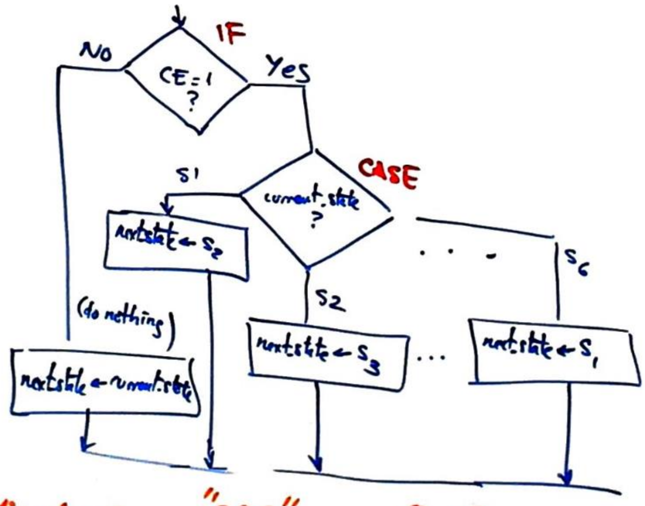


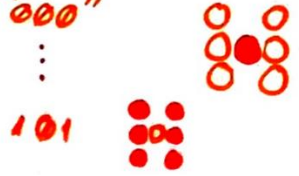
CE	current_state	next_state
0	current_state	current_state
1	S1	S2
1	S2	S3
	⋮	
	S6	S1

current_state	Q
S1	000
S2	001
⋮	
S6	101

→ dice decoder Chip 4

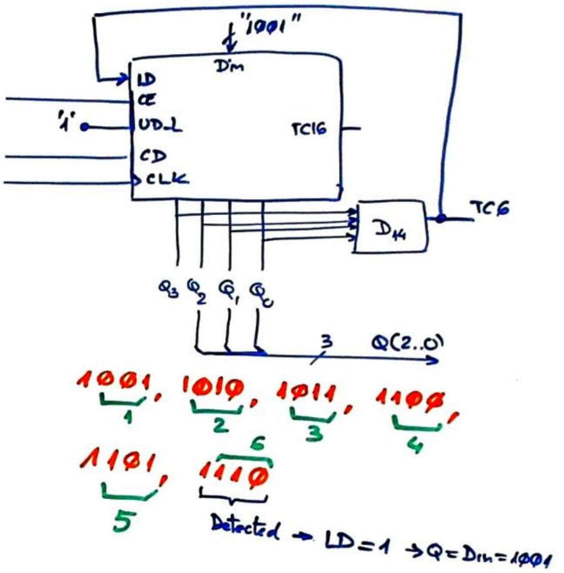
TC6 = '1' when S6 and CE = 1





5. Counter_mod6

(5) For example a truncation from 9 to 14 → Counter_mod6



7. VHDL files, D_FF ?

(7) * Clk_Generator (Chip2)

- Freq_div N1
- Freq_div N2
- D_FF (x2)

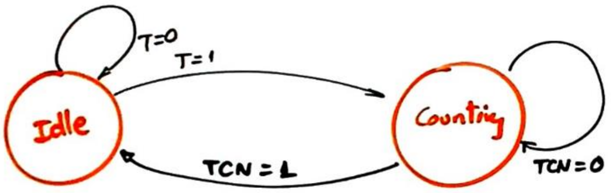
} → 6 +15 2 = 23 D_FF

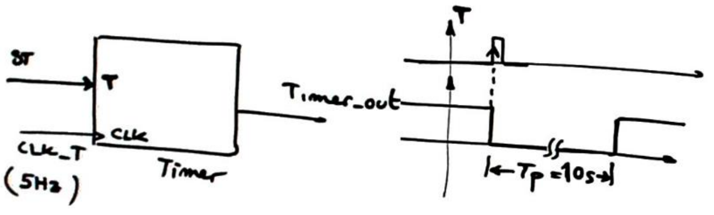
* Chip1 (version A)
  encoding in binary → 3 D_FF
* Chip 3
  - FSM → D_FF (2 states)
  - Freq_div50 → 6 D_FF

⇨ 33 D_FF

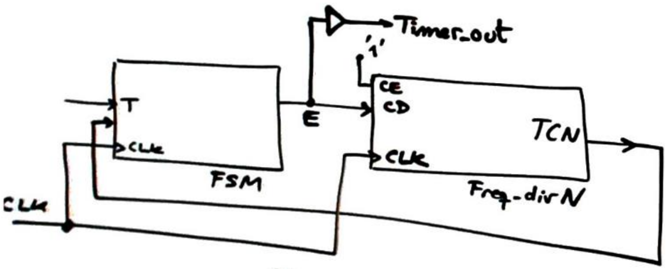
→10 VHDL files (Chip2 version A)
(+ Chip4)

6. Timer

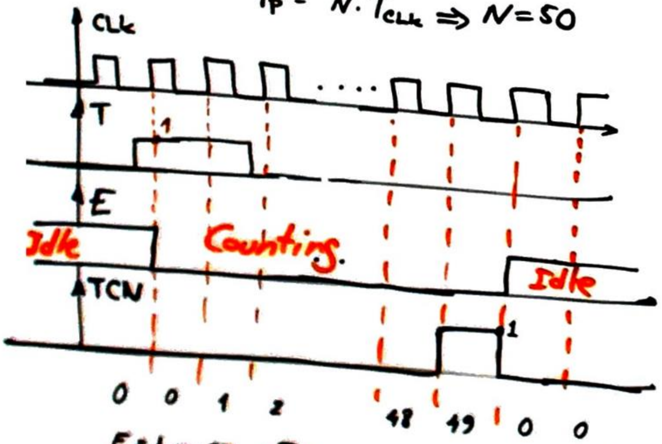




How to replace the *Freq_divN* or the *Counter_modN*, by a shift register?



$T_p = N \cdot T_{CLK} \Rightarrow N=50$

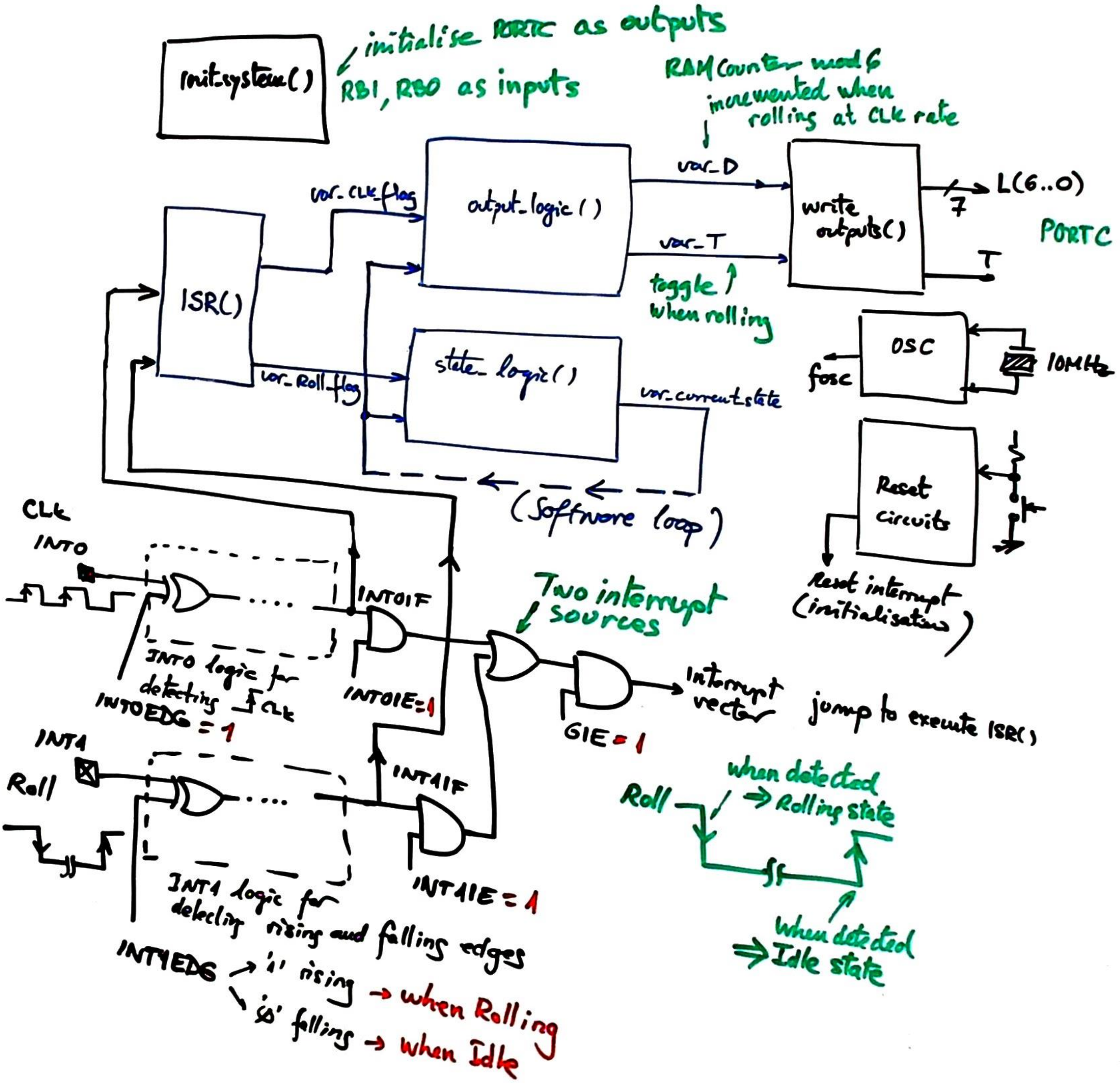


E=1 = CD → Frequency divider (counter) is kept reset (blocked)

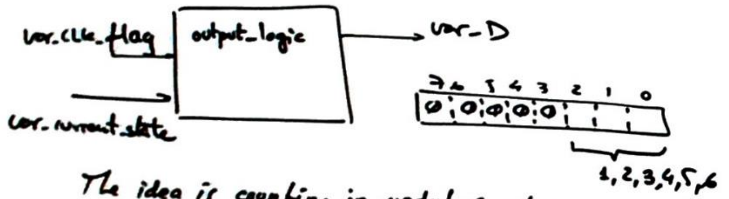
Problem 3

There are several ways to plan and conceive this application. This is only an example considering the state diagram in Fig. 5.

8. Hardware-software diagram

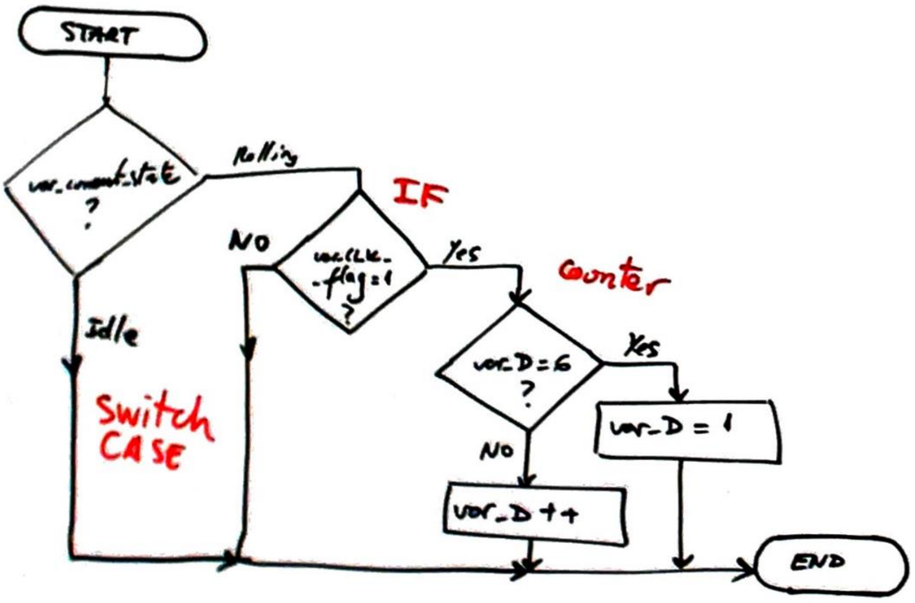


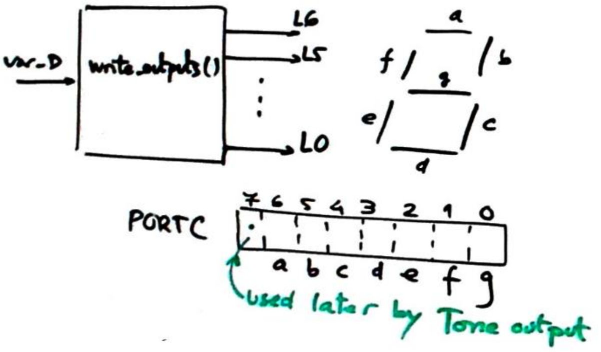
9. *output_logic(), write_outputs()*

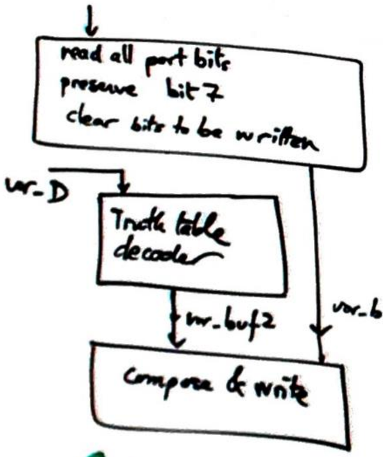


The idea is counting in modulo 6 when rolling

var_CLK_flag	var_current_state	var_D
X	Idle	Do nothing
0	Rolling	var_D ++ (modulo 6)
1	"	Do nothing







var_Buf = PORTC & 10000000;

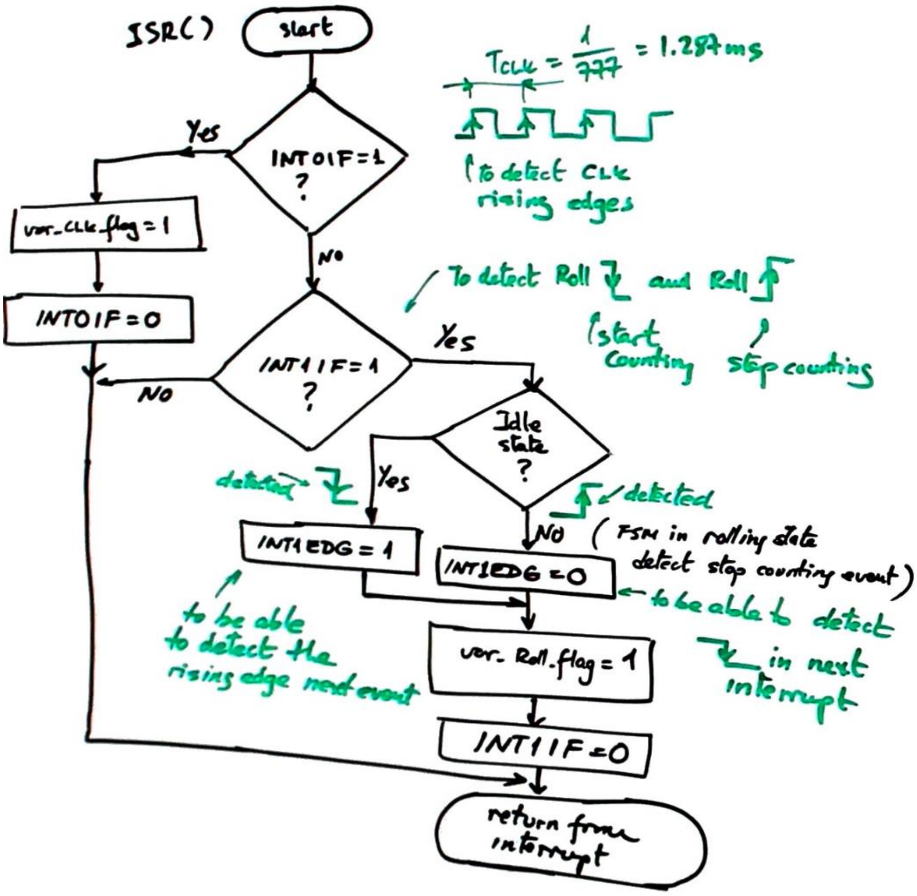
var_D	var_buf2
1	00110000
2	01101101
⋮	⋮
6	01011111

PORTC = var_Buf | var_Buf2 ;

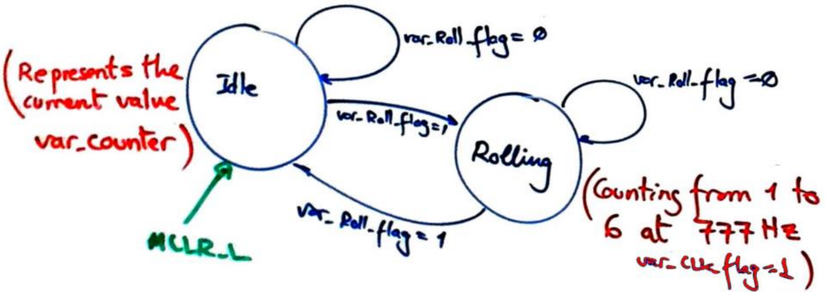
This function may be executed only when there is a new var_D value

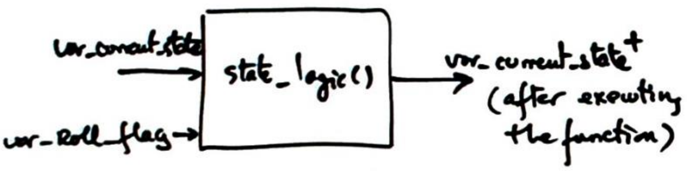
if (var_D ≠ var_D_old) ⇒ execute
else skip (do nothing)

10. Interrupts, *ISR()*

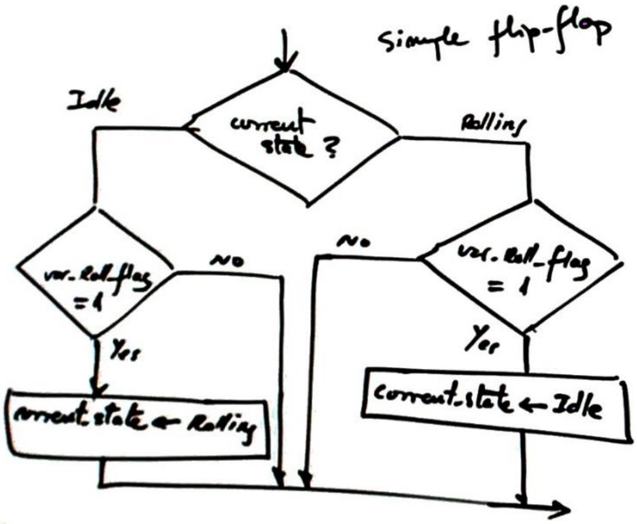

ISR()
start
INT0IF=1 ?
Yes
var_CLK_flag = 1
INT0IF=0
NO
INT1IF=1 ?
Yes
NO
Idle state ?
Yes
NO
INT1EDG = 1
INT1EDG = 0
var_Roll_flag = 1
INT1IF=0
return from interrupt
$T_{CLK} = \frac{1}{777} = 1.287$ ms
to detect CLK rising edges
To detect Roll ↓ and Roll ↑
start counting
stop counting
detecting ↓
↑ detected
(FSM in rolling state detect stop counting event)
to be able to detect the rising edge next event
to be able to detect ↓ in next interrupt

11. *State_logic()*


Idle
Rolling
var_Roll_flag = 0
var_Roll_flag = 1
var_Roll_flag = 0
var_Roll_flag = 1
MCLR_L
(Represents the current value var_counter)
(Counting from 1 to 6 at 777 Hz var_CLK_flag=1)



var_Roll_flag	var_current_state	var_current_state+
0	Idle	Idle
1	Idle	Rolling
0	Rolling	Rolling
1	Rolling	Idle



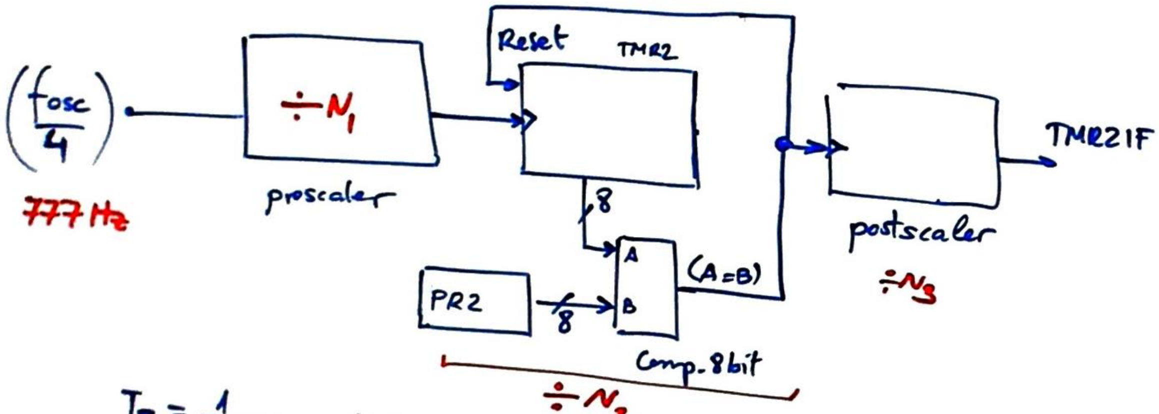
(var_CLK_flag is used by output_logic( ) to run the counter modulo6 (as a datapath)

12. LCD --- Materials for learning LCD is in P11

The LCD is connected in PORTD, and we can use the library of LCD functions and headers to compile applications and generate *.HEX and *.COF files.

13. TMR2 for generatin a timing period T_P = T_{CLK} = 1/ 777 Hz = 1287 µs

Generate a $T_p = T_{CLK}$ to replace external INT0 using TMR2



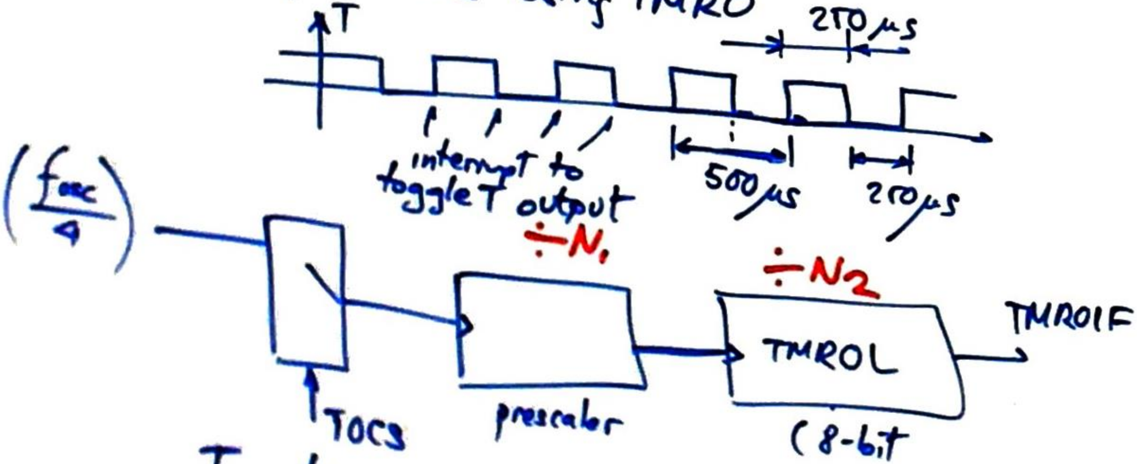
$$T_p = \frac{1}{777\,Hz} = 1287\,\mu s = \left(\frac{4}{10\,MHz}\right)\cdot N_1 \cdot N_2 \cdot N_3 = 1286.4\,\mu s$$

$$0.4\,\mu s \cdot 16 \cdot 201 \cdot 1$$

also → $0.4\,\mu s \cdot 1 \cdot 99 \cdot 13 = 1287\,\mu s$

14. TMR0 for generating a tone of 2 kHz, toggle output every T_P = 1 ms

Generate a 2kHz square wave using TMR0



$$T_p = \left(\frac{4}{f_{osc}}\right)\cdot N_1 \cdot N_2 = 250\,\mu s = \frac{4}{(10\,MHz)}\cdot N_1 \cdot N_2$$

$$249.6\,\mu s = 0.4\,\mu s \cdot 16 \cdot 39$$