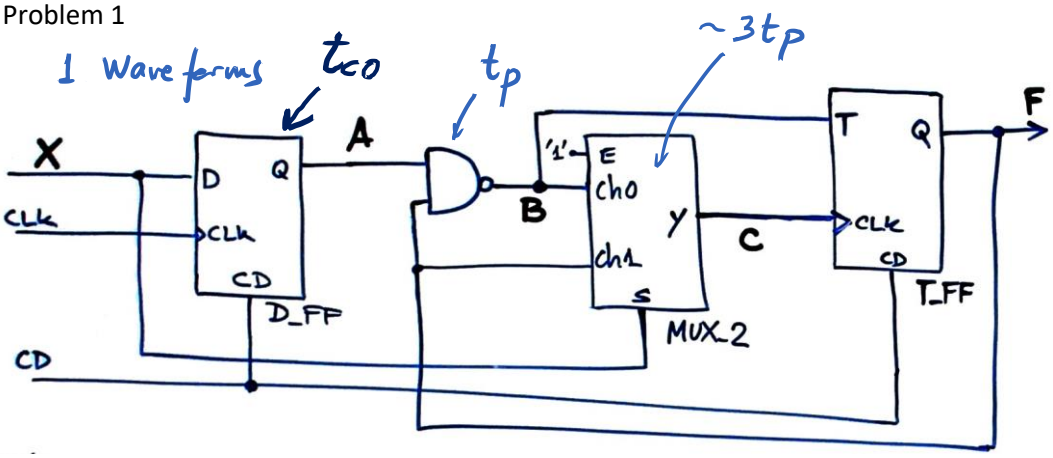
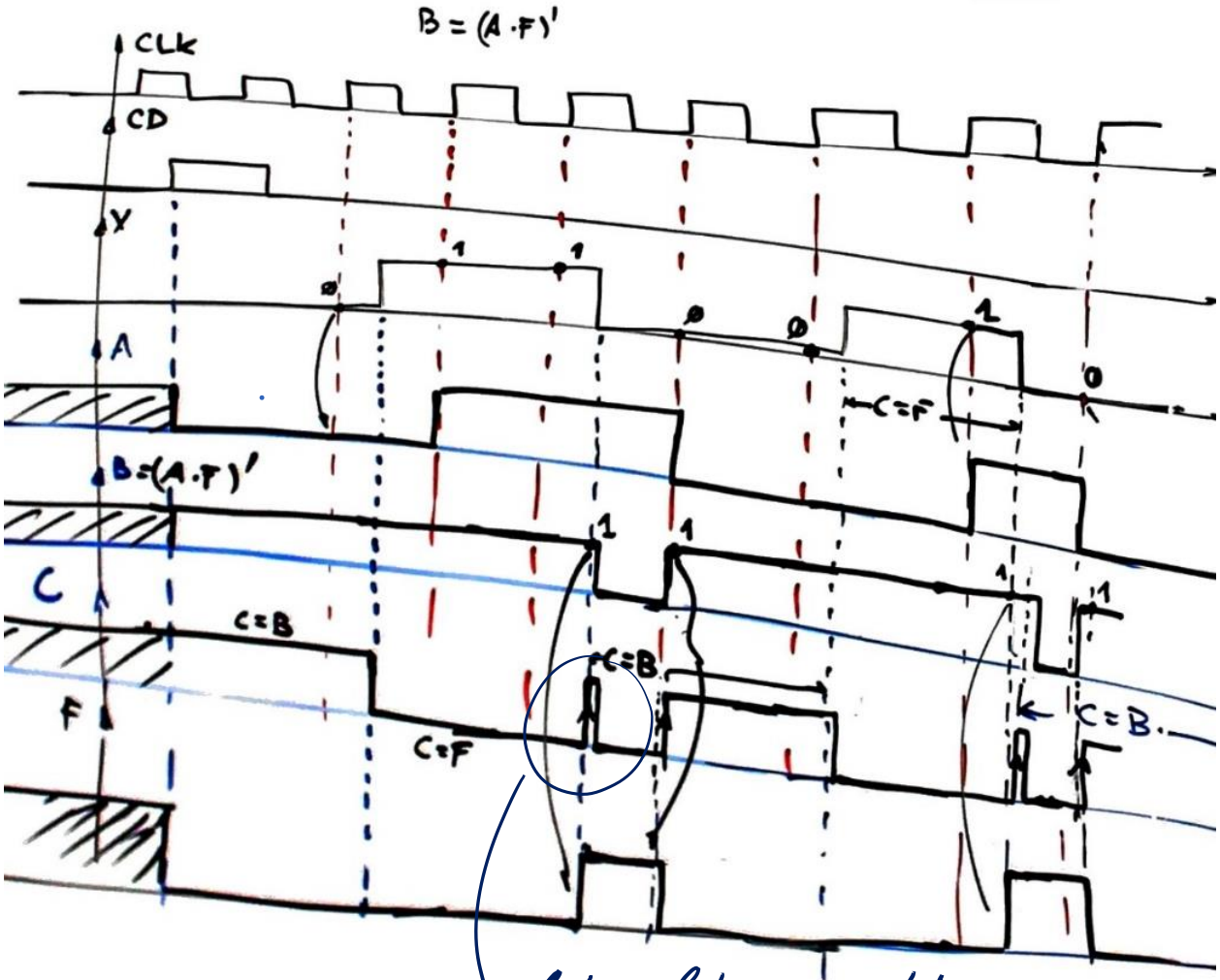
Problem 1



Let's suppose that signals propagate in some ns through the gates



looks like a glitch, but it is enough to generate a rising edge for the T_FF

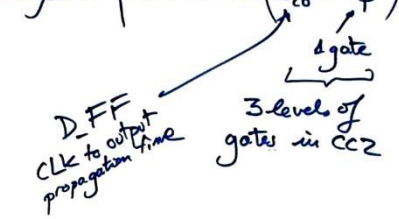
F pulse width depends on the delay between X falling edge and CLK rising edge

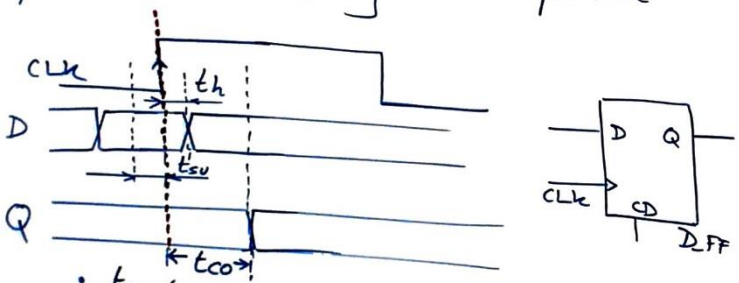
2.

Synchronous circuits are designed systematicaly as FSM and their propagation delays is predictable ($t_{co}+3.t_p$)

Asynchronous circuits are complicated for two main reasons:

- Several CLK signals do not allow systematic design
- CLK delays in ns range can generate unpredictable, outputs and FF times may not be respected





- t_{su} (setup time)
  D input must be stable t_{su} before CLK
- t_{co} (clock to output time)
- t_h (hold time)
  D input must be stable t_h after CLK

This circuit is not standard and its application is not clear. Simply for observing how complicated is to work with several CLK signals that depends on technologies and propagation times

Delay measurement between two signals can be implemented using canonical synchronous FSM where all FF are driven from the same CLK and the technology is not a relevant parameter if we do not have to work at the maximum frequency of operation.

3.

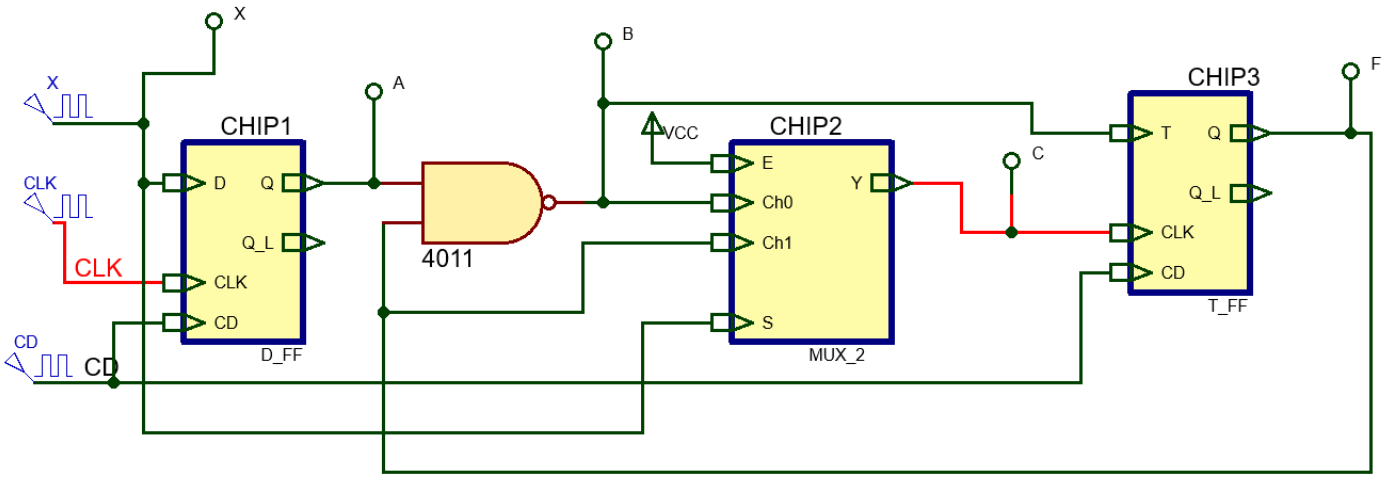
Plan C2 project in VHDL requires D_FF.vhd, T_FF.vhd, MUX_2.vhd and the top Circuit_P1.vhd files. For testing purposes, the testbench file Circuit_P1_tb.vhd is also required.

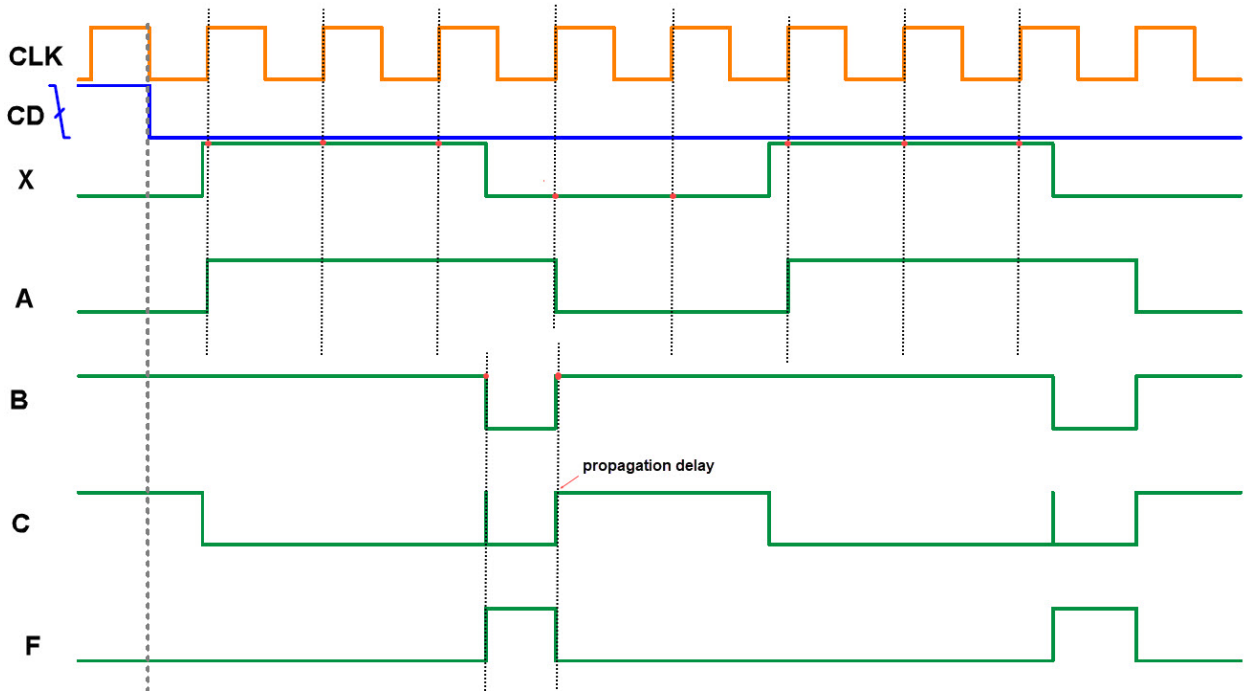
The circuit can be solved, as usual, using three methods: handwritten, Proteus and VHDL. This solution includes all the required files. The circuit works in the range of ns, where an apparent “glitch” is enough for driving the *T_FF*.

Functional simulation is not giving a clear sight of what is happening in the circuit. Thus, using gate-level simulations and trying different target chips (MAX II CPLD, Cyclone IV FPGA, MAX10, FPGA, etc) may give you a better comprehension on how this circuit works.

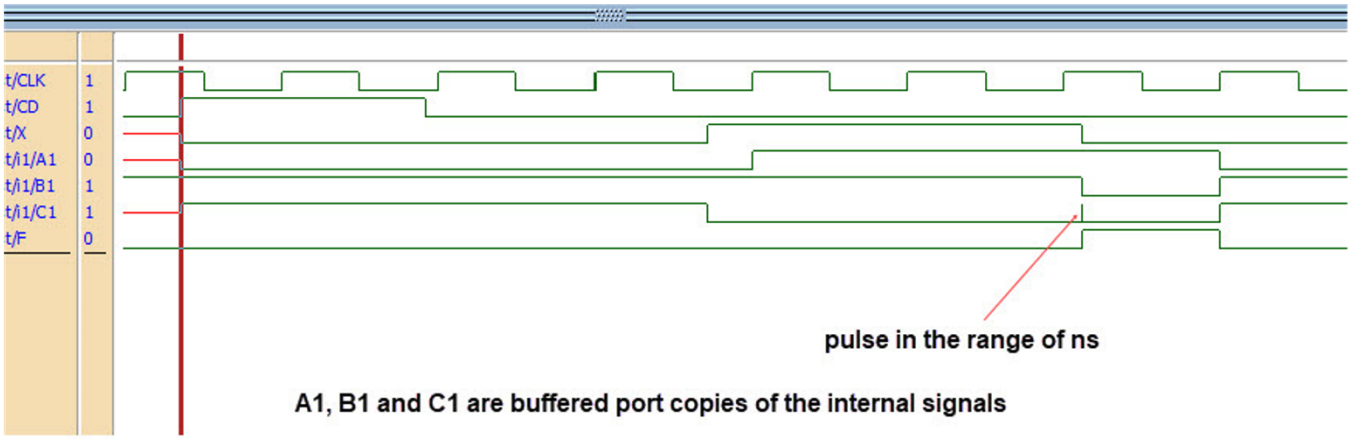
Preparing a laboratory prototype on a protoboard, for instance using CMOS classic technology chips like in Proteus, is also interesting in this case.

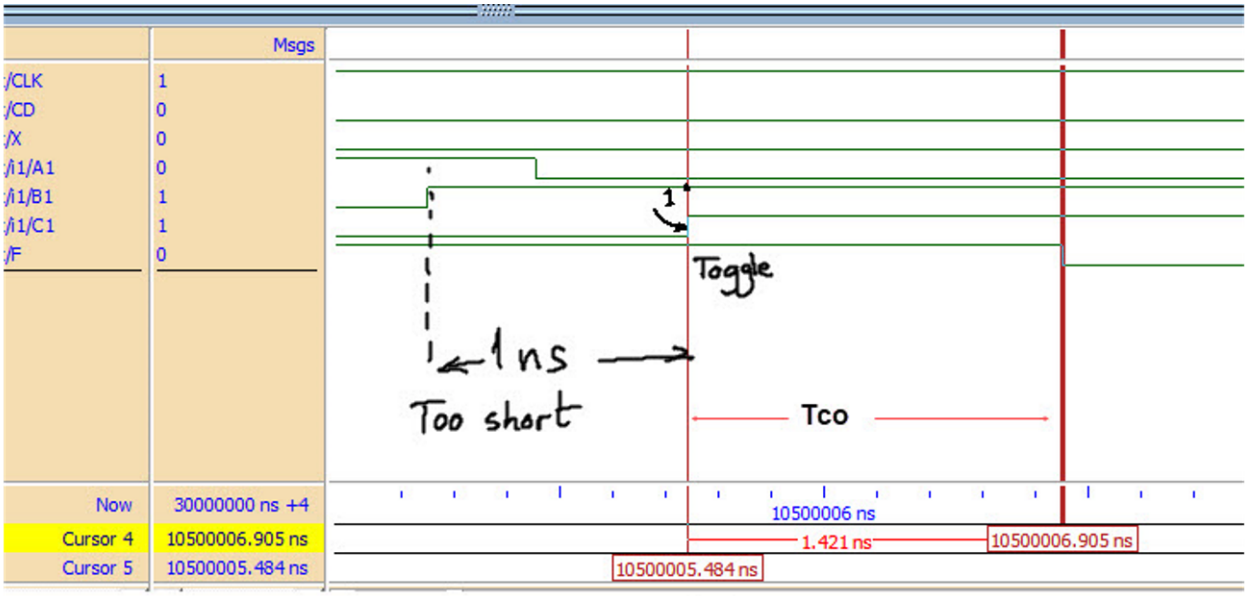
This is the circuit and the timing diagram from Proteus:





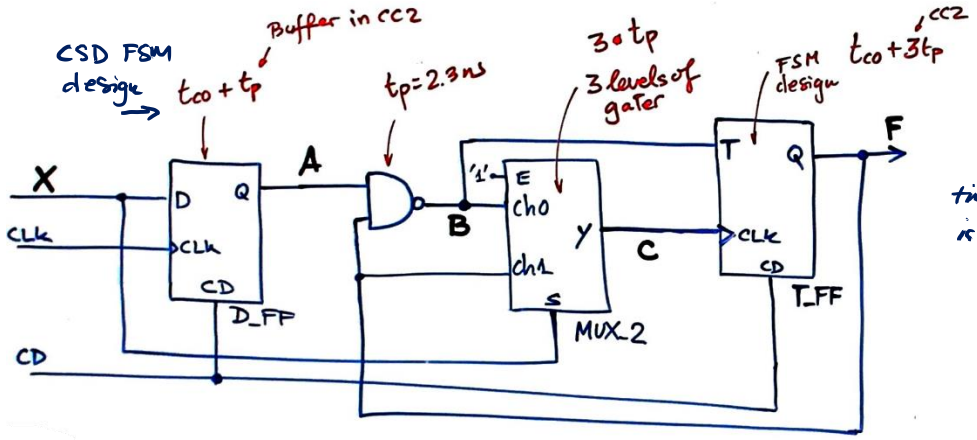
This is the timing diagram from Quartus Prime project and ModelSim targeting a MAXII CPLD device (project files available in DIGSYS).





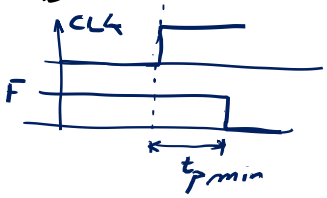
(*) Try this gate level simulation using MAX II and Cyclone IV devices

4. Simply an approximate estimation considering the architecture of the circuit.



Let's imagine that the propagation time from CLK to output for a simple D-FF is similar to tp of one gate

At the limit, we can consider that when there is a CLK rising edge, signals A → B → C switch in a chain, and C ↓ also toggles F value

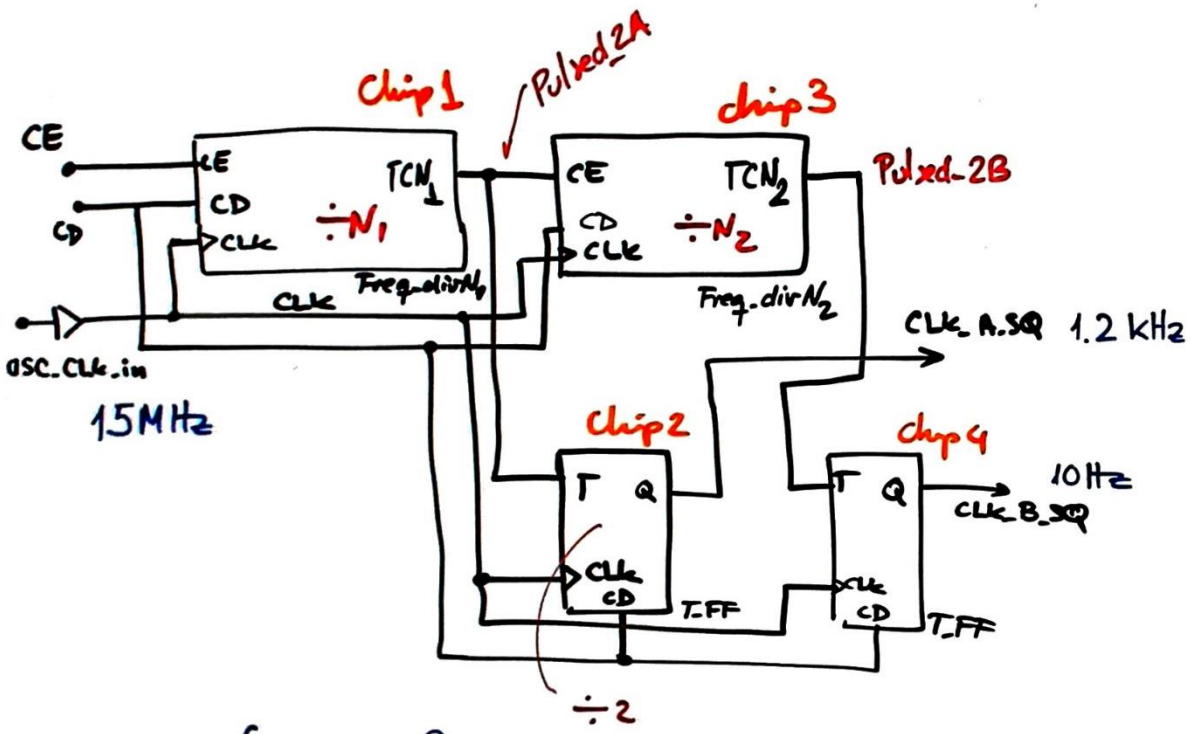


$t_{p\,min} \simeq t_{co} + t_p + t_p + 3t_p + t_{co} + 3t_p = 23\,ns$

$f_{max} \leq 43.4\,MHz$

Problem 2

1.

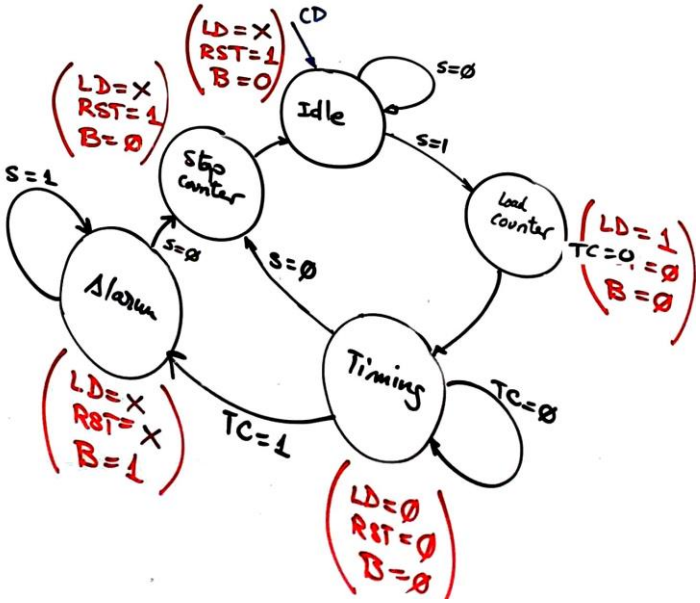


$$f_{CLK\_A\_SQ} = \frac{f_{Pulsed\_2A}}{2} = \frac{f_{OSC\_CLK\_in}}{2 \cdot N_1} \qquad N_1 = \frac{15MHz}{2 \cdot 1.2kHz} = 6250$$

$$f_{CLK\_B\_SQ} = \frac{f_{Pulsed\_2B}}{2} = \frac{f_{Pulsed\_2A}}{2 \cdot N_2} = \frac{f_{OSC\_CLK\_in}}{2 \cdot N_1 \cdot N_2} \qquad N_2 = \frac{15MHz}{2 \cdot 6250 \cdot 10} = 120$$

$$\text{Number of D\_FF} \rightarrow \left(\frac{1}{\lg 2}\right) \cdot \lg_{10}(6250) + 7 + 2 = 13 + 9 = 22$$

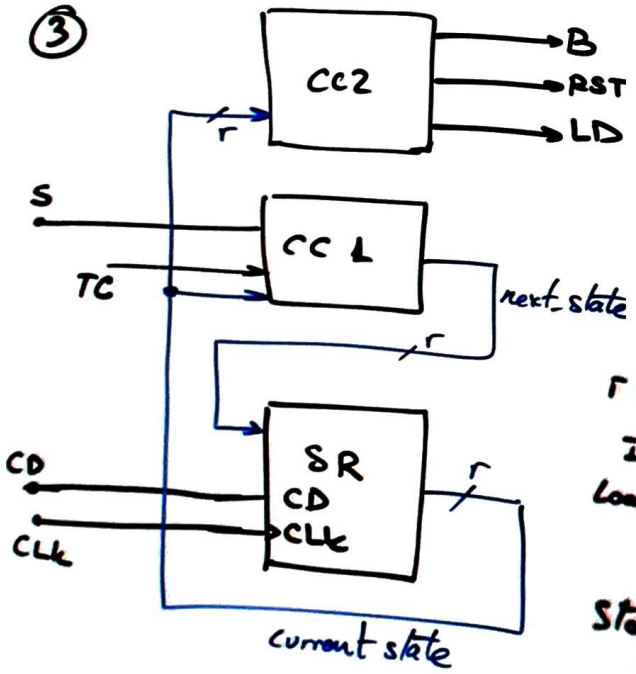
2.



The datapath counter is not used when Idle or when generating the alarm tone, so we don't care about RST, LD signals in some states

- Load counter → LD = 1
- Timing → LD = Ø, RST = Ø
- Alarm → B = 1

③



r = 5 one-hot 5 states

Idle → 00001

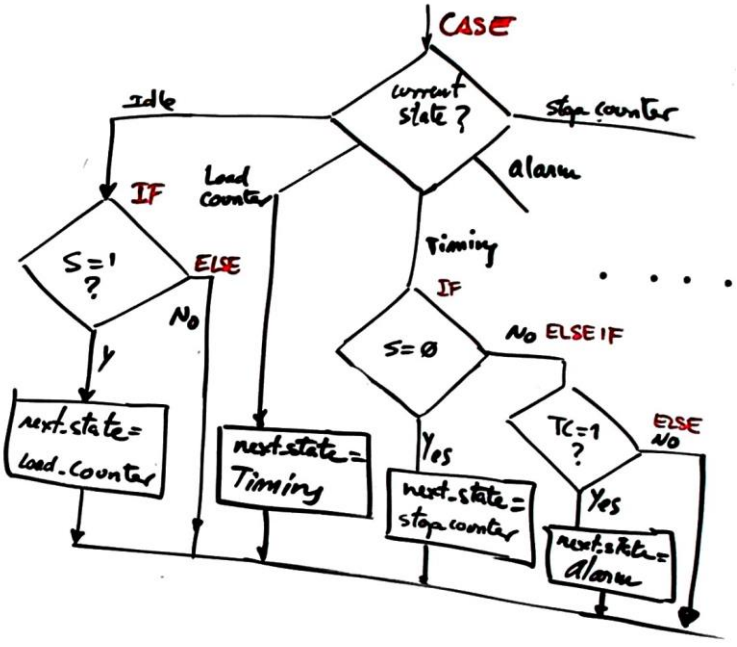
Load counter → 00010

⋮

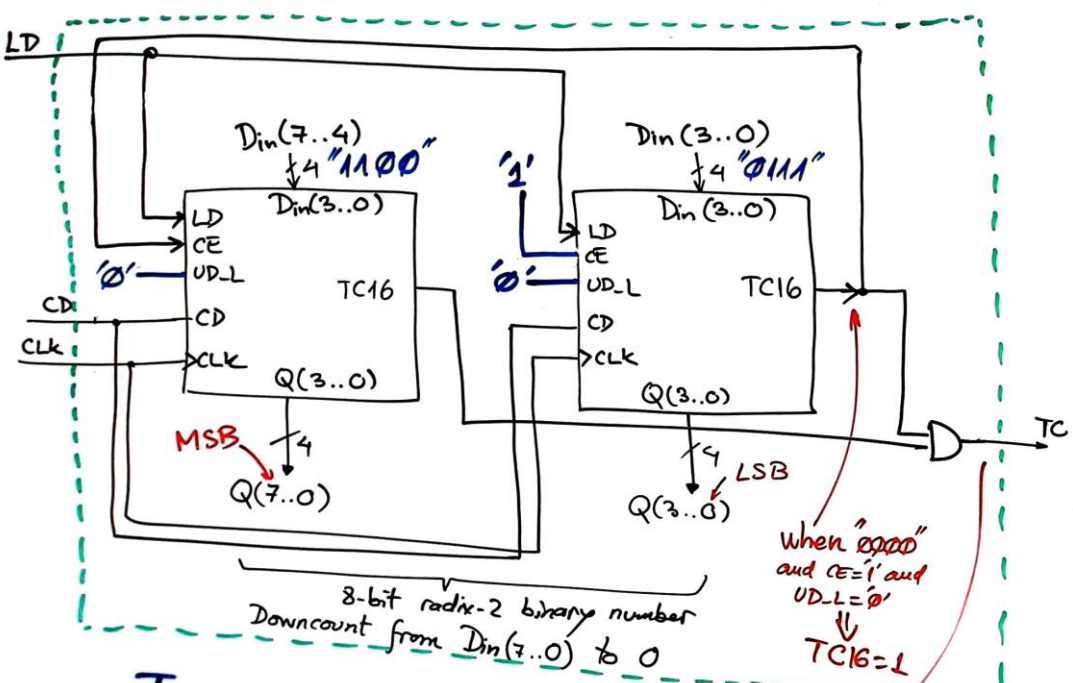
Stop counter → 10000

④

TC	S	current_state	next_state
x	0	Idle	Idle
x	1	Idle	Load counter
x	x	Load counter	Timing
x	0	Timing	Stop counter
1	1	Timing	Alarm
0	1	Timing	Timing
x	1	alarm	alarm
x	0	alarm	Stoptimer
x	x	Stoptimer	Idle



5



Timing period = $T_p = 20s = N_1 \cdot T_{CLK}$ (100ms)

$N_1 = 200 \Rightarrow Din = 199 = "11000111"$

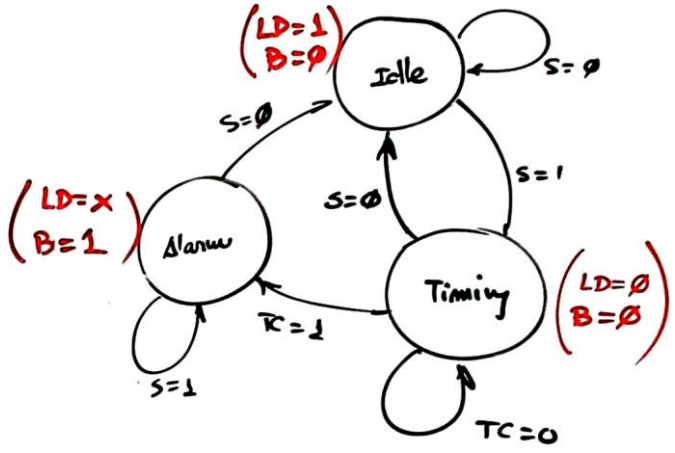
Datapath is a Counter_mod200 built using a chain of two Counter_mod16 and count truncation

TC is the status signal (flag) from the datapath to control the FSM

TC = 1 when "00000000"

Datapath

6. Counter is loading in parallel continuously when Idle and downcounting when timing



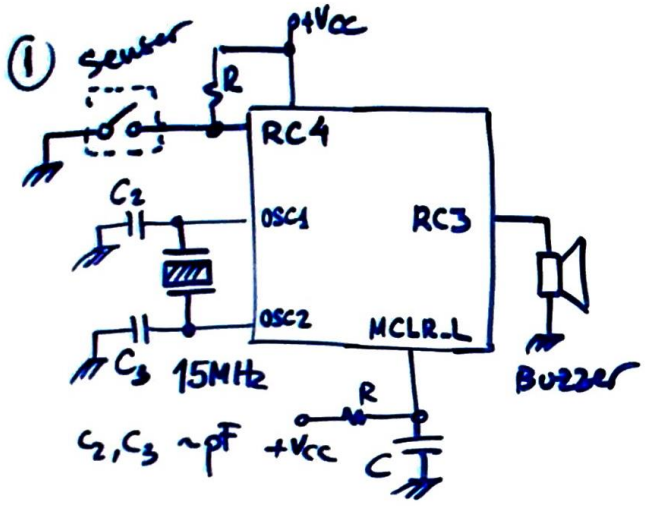
⑦ CLK_generator → 22 D_FF

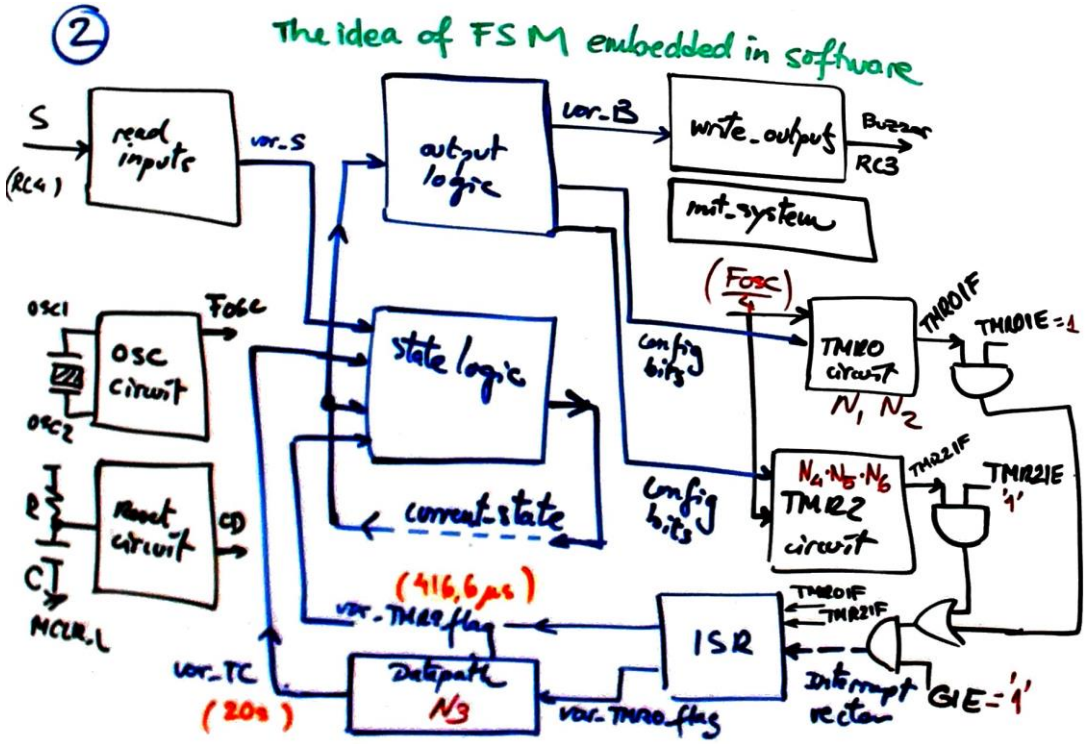
FSM → 5 D_FF

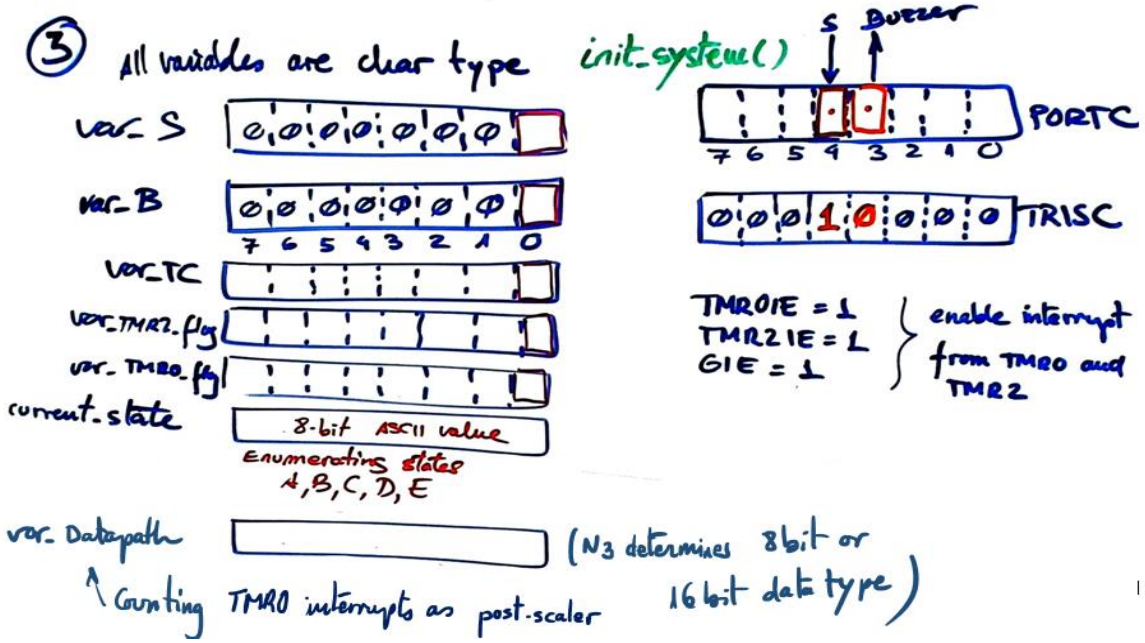
Datapath → 8 D_FF

35 registers (1-bit D_FF memory cells)

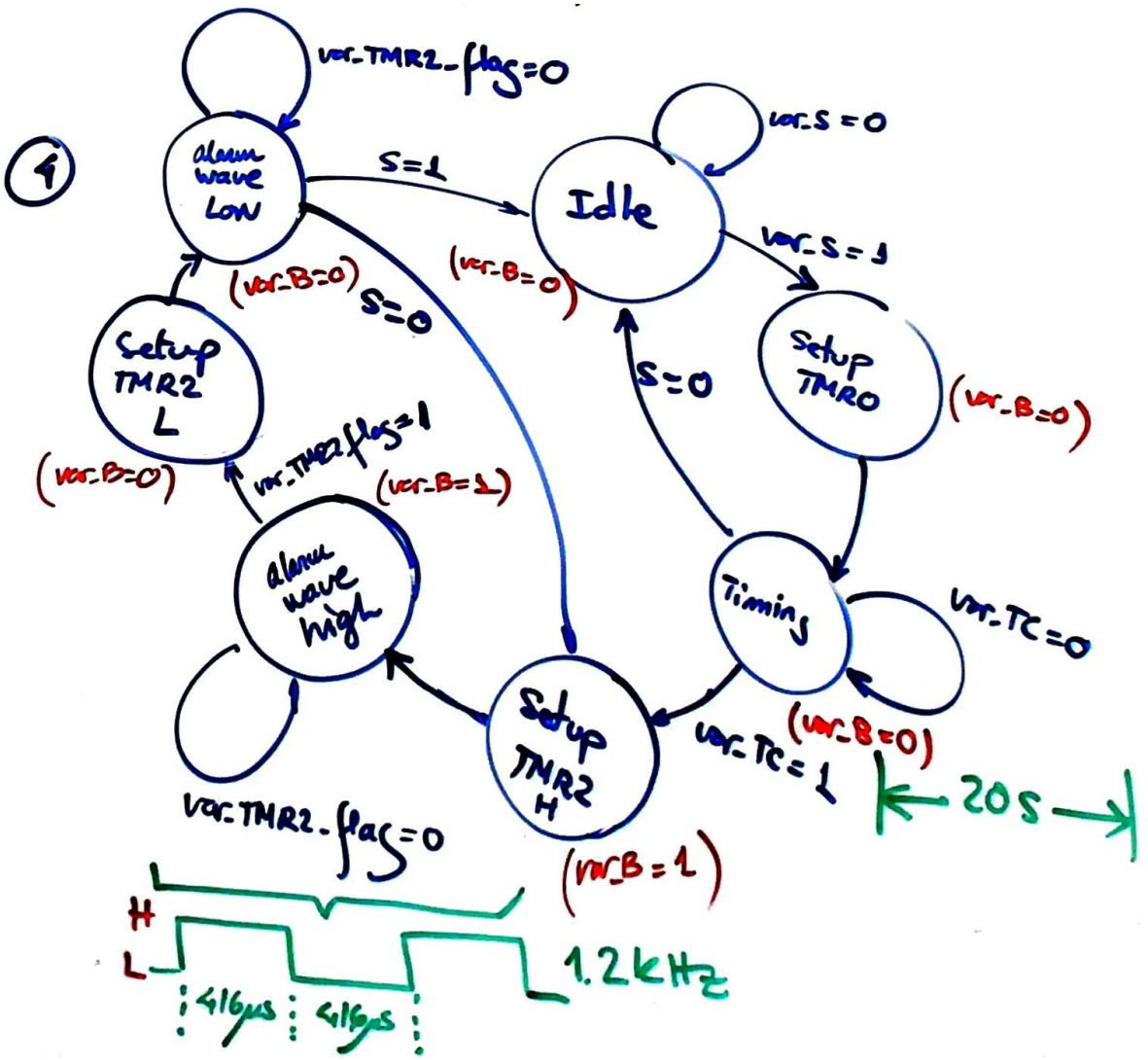
Problem 3


① sensor
+Vcc
R
RC4
C2
OSC1
RC3
OSC2
MCLR_L
C3 15MHz
Buzzer
R
C2, C3 ~pF +Vcc
C

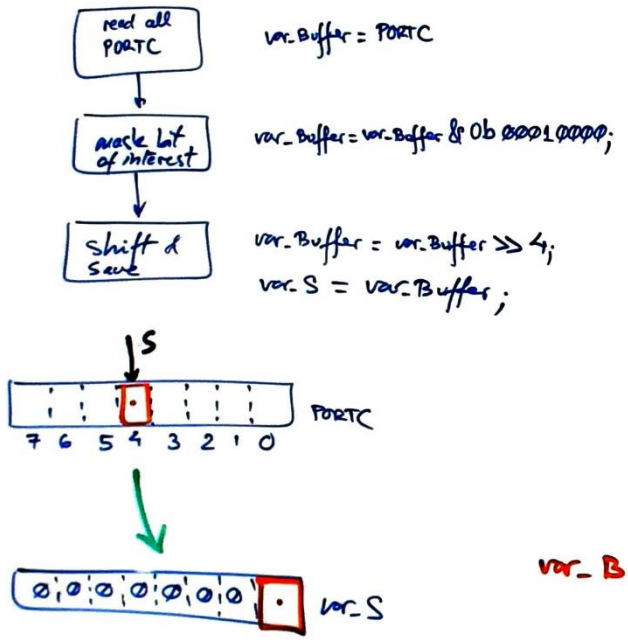

② The idea of FSM embedded in software
S
(RC4)
read inputs
var_S
output logic
var_B
write_outputs
Buzzer
RC3
init_system
(Fosc/4)
OSC1
OSC circuit
Fosc
OSC2
state logic
config bits
TMR0 circuit
N1, N2
TMR0IF
TMR0IE = 1
R
C
Reset circuit
MCLR_L
current_state
N4·N5·N6
TMR2 circuit
TMR2IF
TMR2IE
'1'
(416,6 μs)
var_TMR0_flag
ISR
TMR0IF
TMR2IF
Interrupt vector
GIE = '1'
var_TC
(20s)
Datapath N3
var_TMR0_flag


③ All variables are char type
init_system()
S
Buzzer
var_S
PORTC
7 6 5 4 3 2 1 0
var_B
0 0 0 1 0 0 0 0
TRISC
var_TC
var_TMR2_flg
var_TMR0_flg
TMR0IE = 1
TMR2IE = 1
GIE = 1
enable interrupt from TMR0 and TMR2
current_state
8-bit ASCII value
Enumerating states
A, B, C, D, E
var_Datapath
(N3 determines 8bit or 16bit data type)
Counting TMR0 interrupts as post-scaler

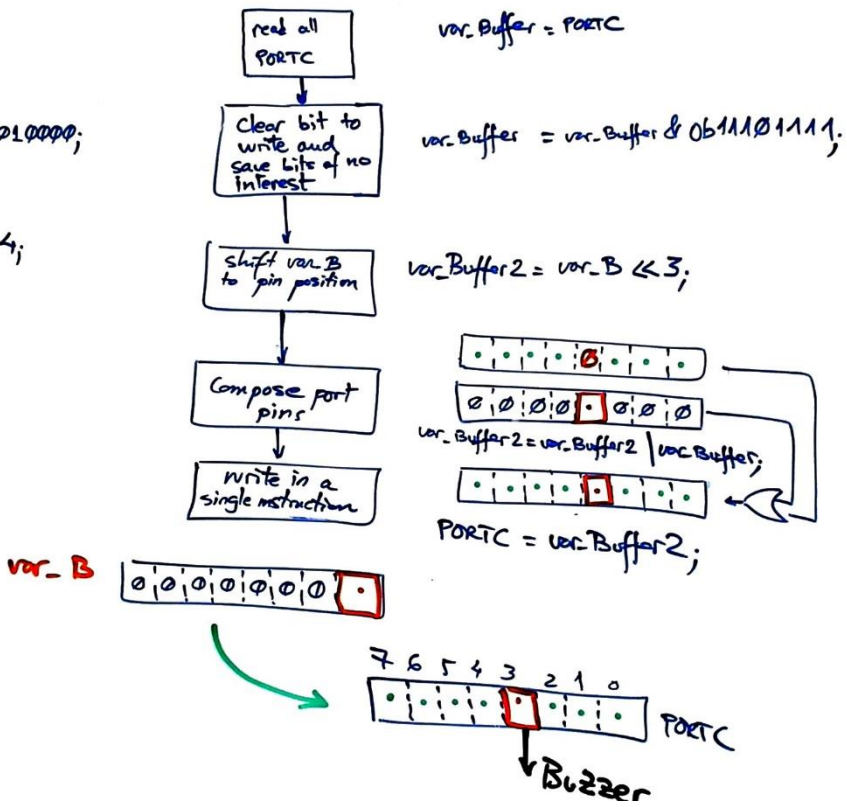
④



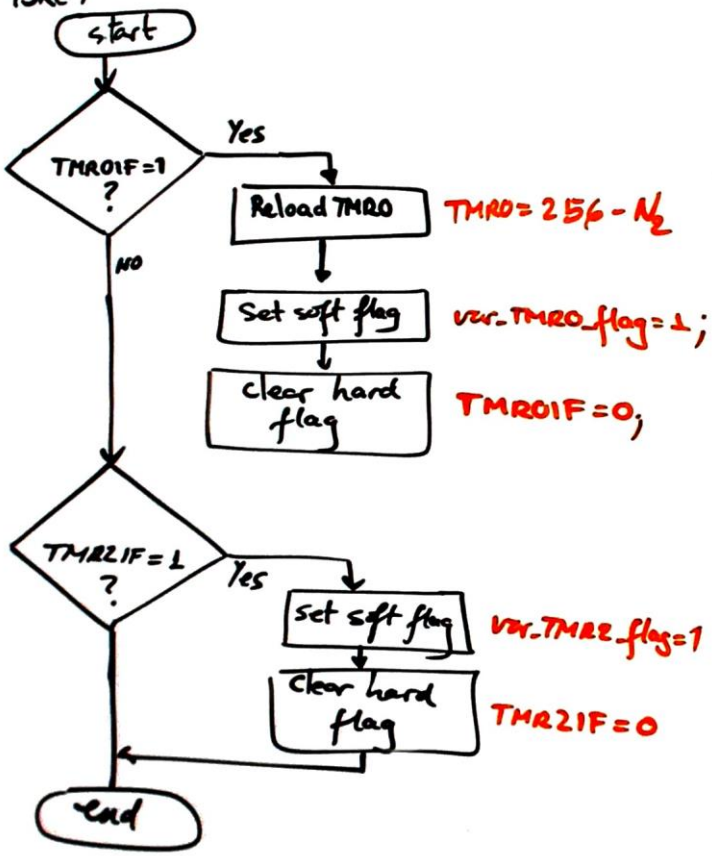
⑤ read_inputs()



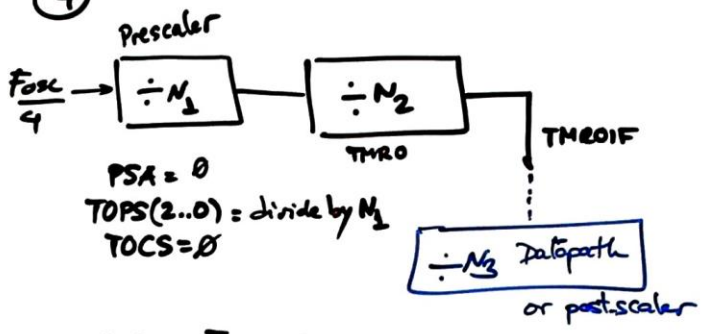
write_outputs()



⑥ ISRC)



⑦



$T_P = T_{osc} \cdot 4 \cdot N_1 \cdot N_2 \cdot N_3$

$20s = \frac{1}{15MHz} \cdot 4 \cdot N_1 \cdot N_2 \cdot N_3 \simeq 19,97s$

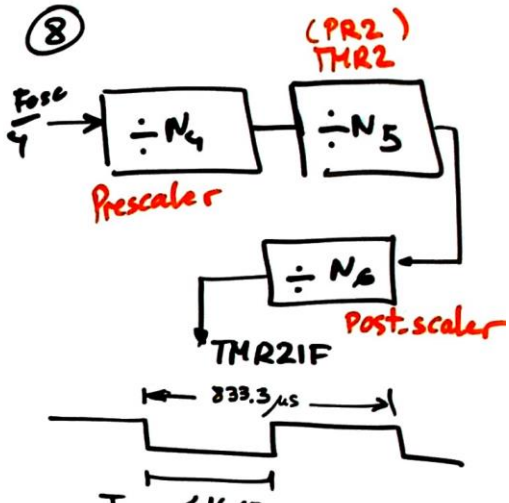
$8ms \cdot 2500$

$N_1 \cdot N_2 = \frac{8ms}{4} 15MHz = 30000$

$256 \cdot 117 = 29952 \simeq$

TOPS(2..0) = "111"

⑧



$T_L = 416,67\mu s = T_H$

$T_L = \left(\frac{4}{F_{osc}}\right) \cdot N_4 \cdot N_5 \cdot N_6$

$\frac{416.67\mu s}{(4/15MHz)} = N_4 \cdot N_5 \cdot N_6$

$1562.5 \simeq N_4 \cdot N_5 \cdot N_6$

$1 \cdot 11 \cdot PR2 = 142$

$f_{Buzzer} = 1200.38 Hz$