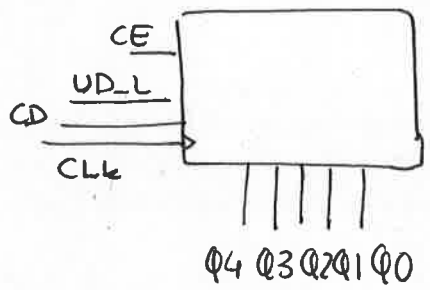
Ideas on the exam solution

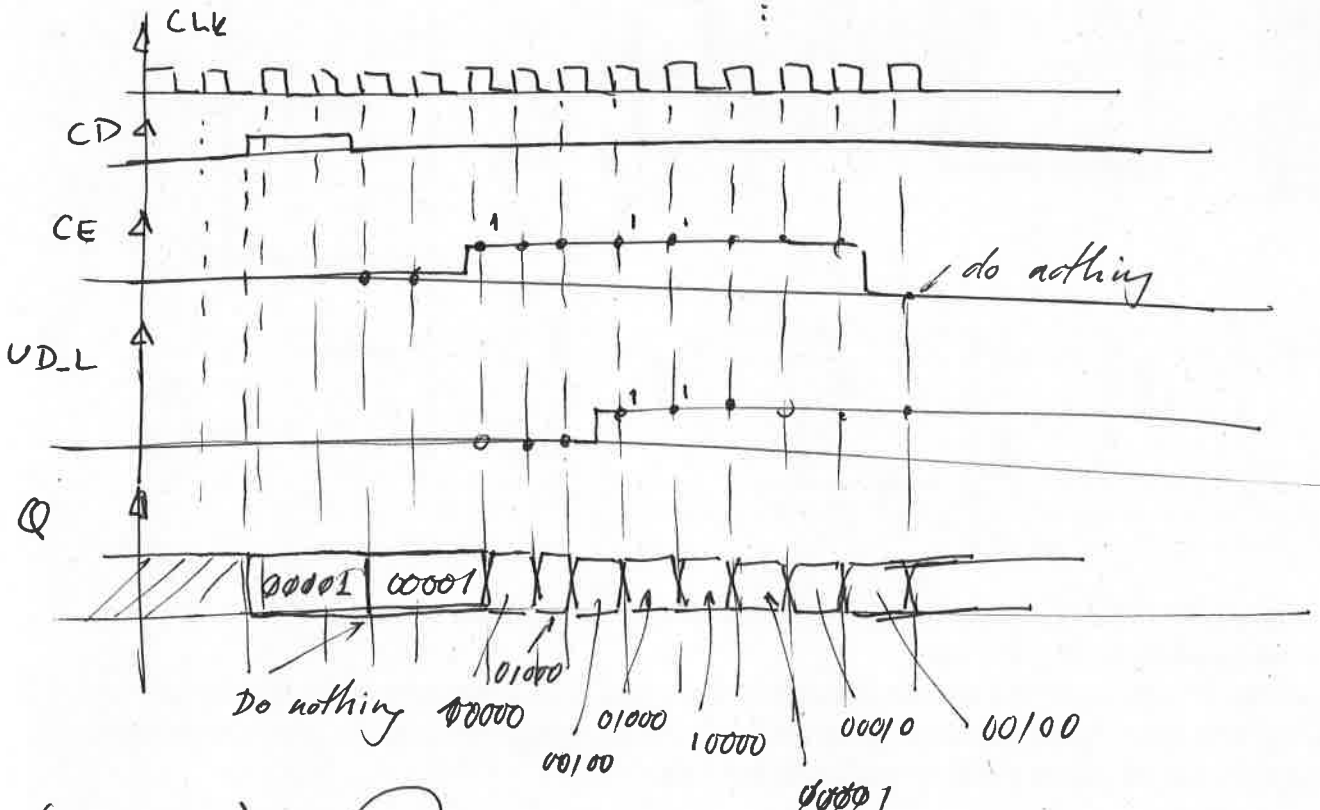
Problem 1.

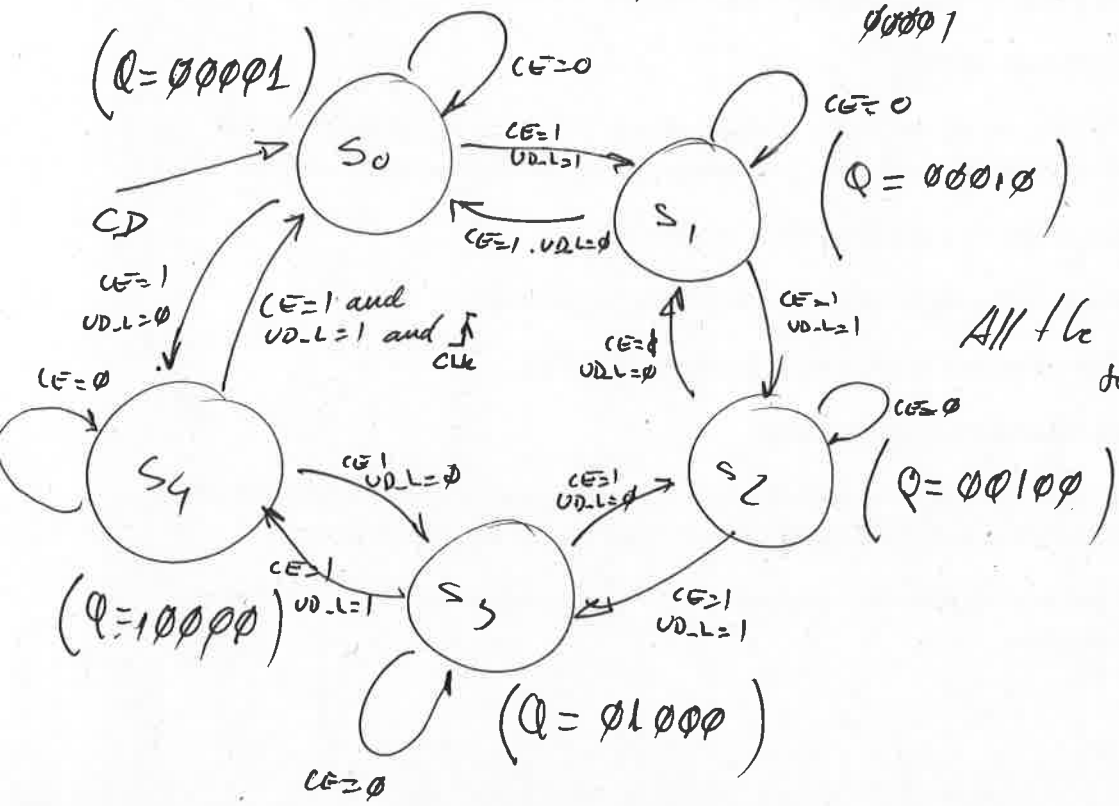
(a)



$Q_4 Q_3 Q_2 Q_1 Q_0$	
00001	S_0
00010	S_1
00100	S_2
01000	S_3
10000	S_4
00001	

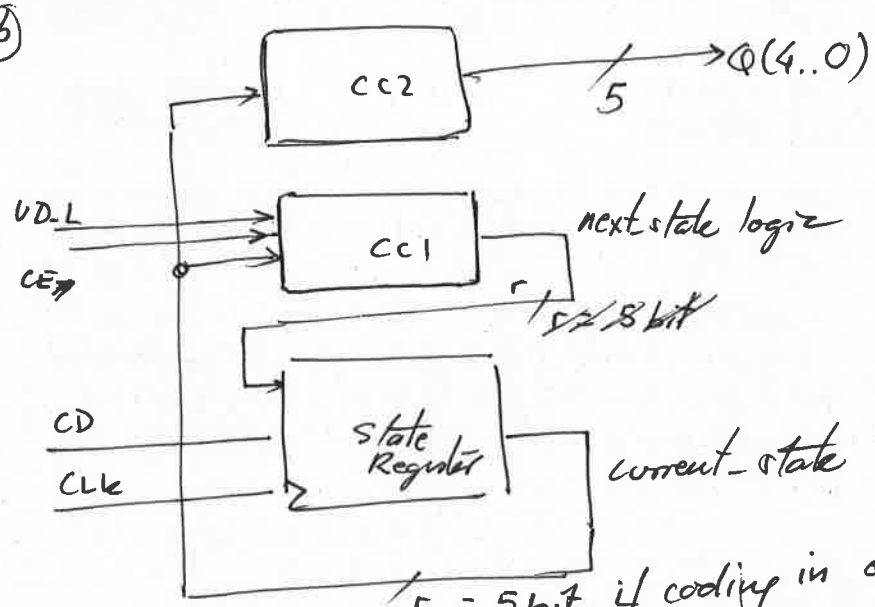
CE	UD_L	Q+(4..0)
0	x	Q
1	1	(Q+1) one-hot 5bit
1	0	(Q-1) one-hot 5bit





All the transitions are sensitive to the CLK ↓

(b)



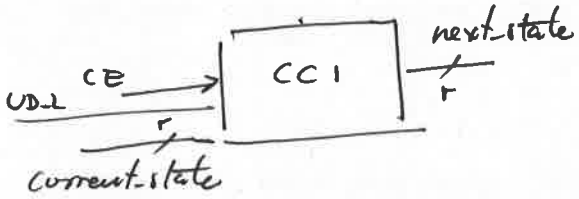
r = 5 bit if coding in one-hot (5 D-FF)

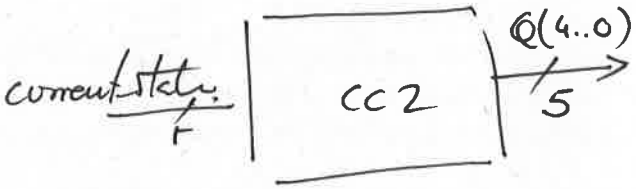
r = 3 bit if coding in binary or Gray (3 D-FF)

(d)

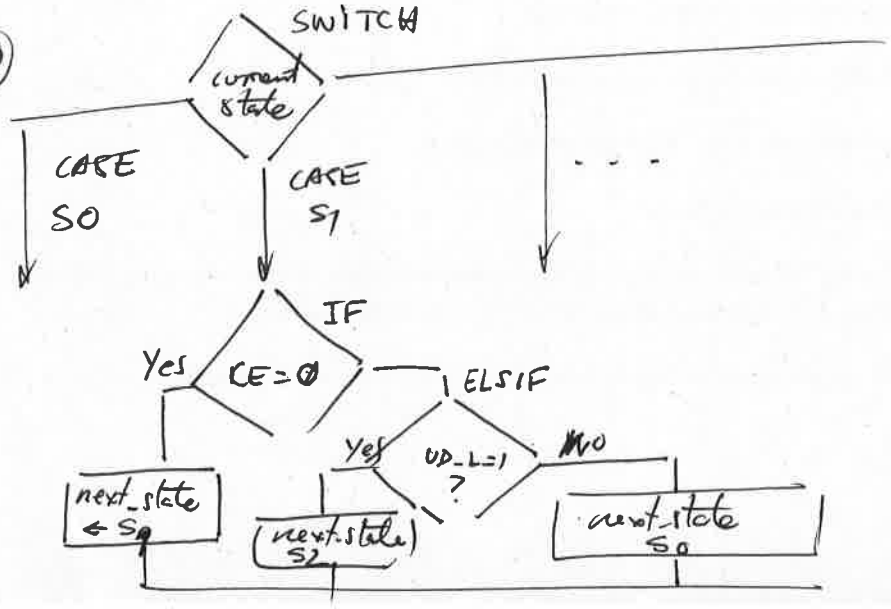
CE	UD_L	current_state	next_state
0	X	S_0	S_0
1	1	S_0	S_1
1	0	S_0	S_4
0	X	S_1	S_1
1	1	S_1	S_2
1	0	S_1	S_0

current_state	Q(4..0)
S_0	0 0 0 0 1
S_1	0 0 0 1 0
S_2	0 0 1 0 0
S_3	0 1 0 0 0
S_4	1 0 0 0

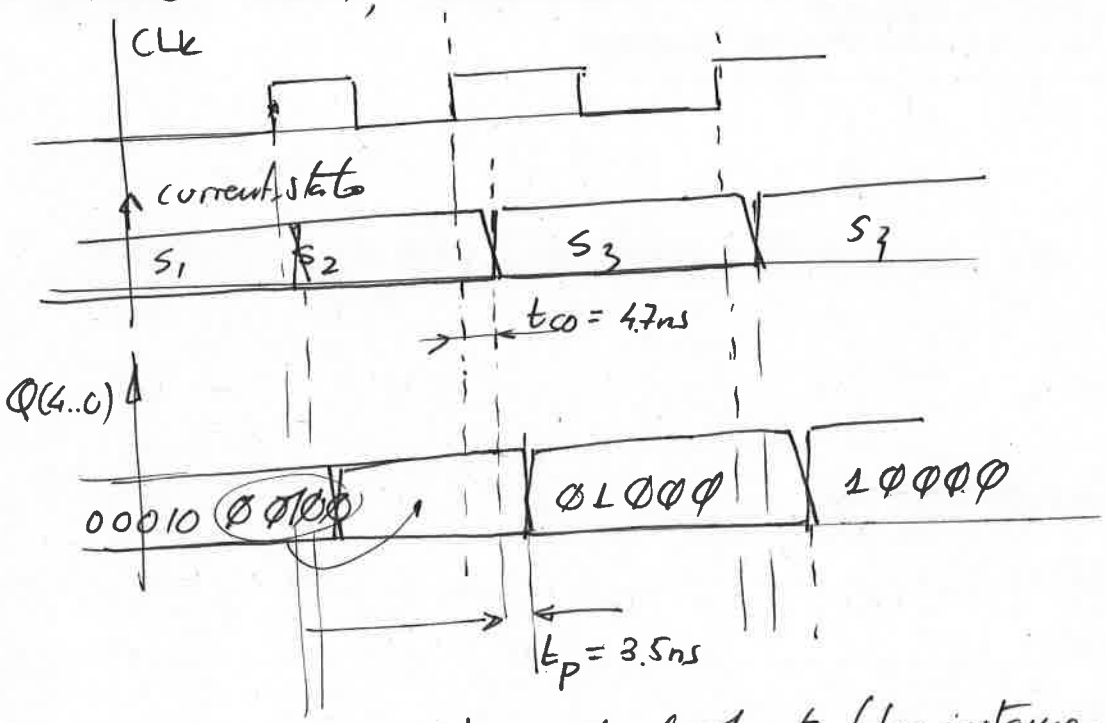


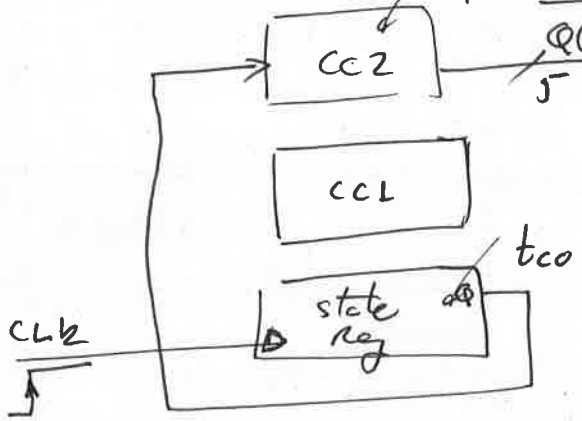


(e)



(b) For instance CG = 1; UD_L = 1



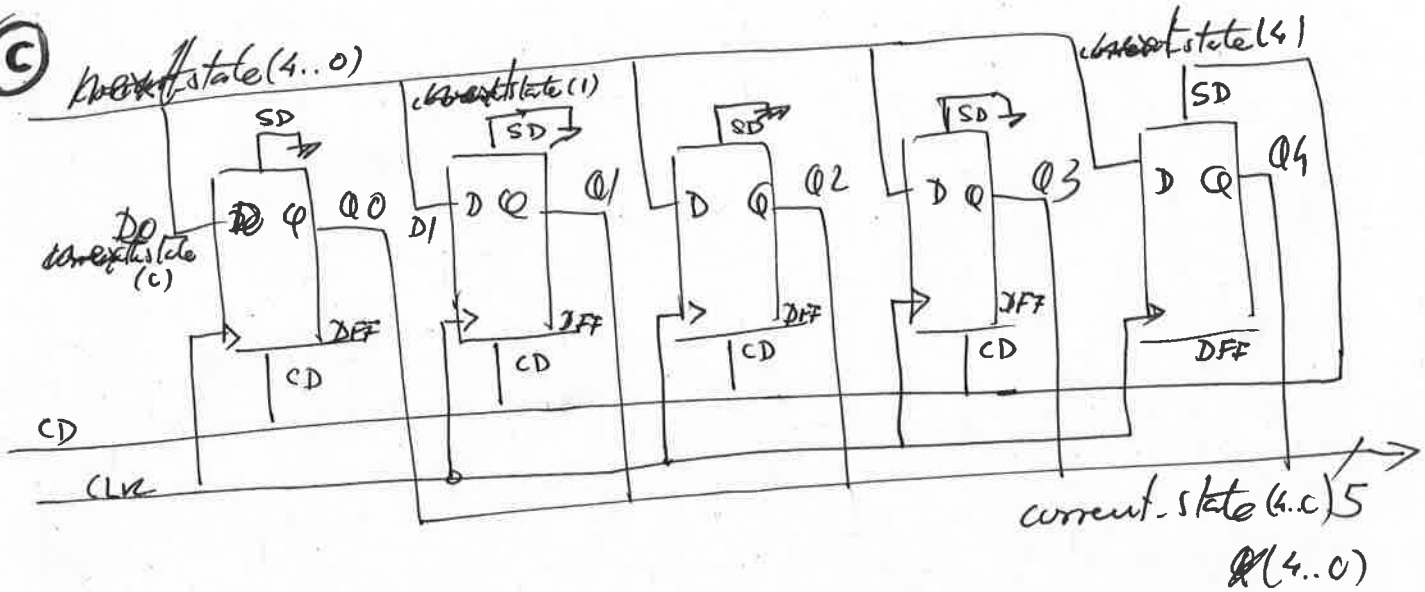


(for instance, if the combinational circuit is solved with POS)

$$f_{max} \leq \frac{1}{4.7\,ns + 3 \cdot 3.5\,ns}$$

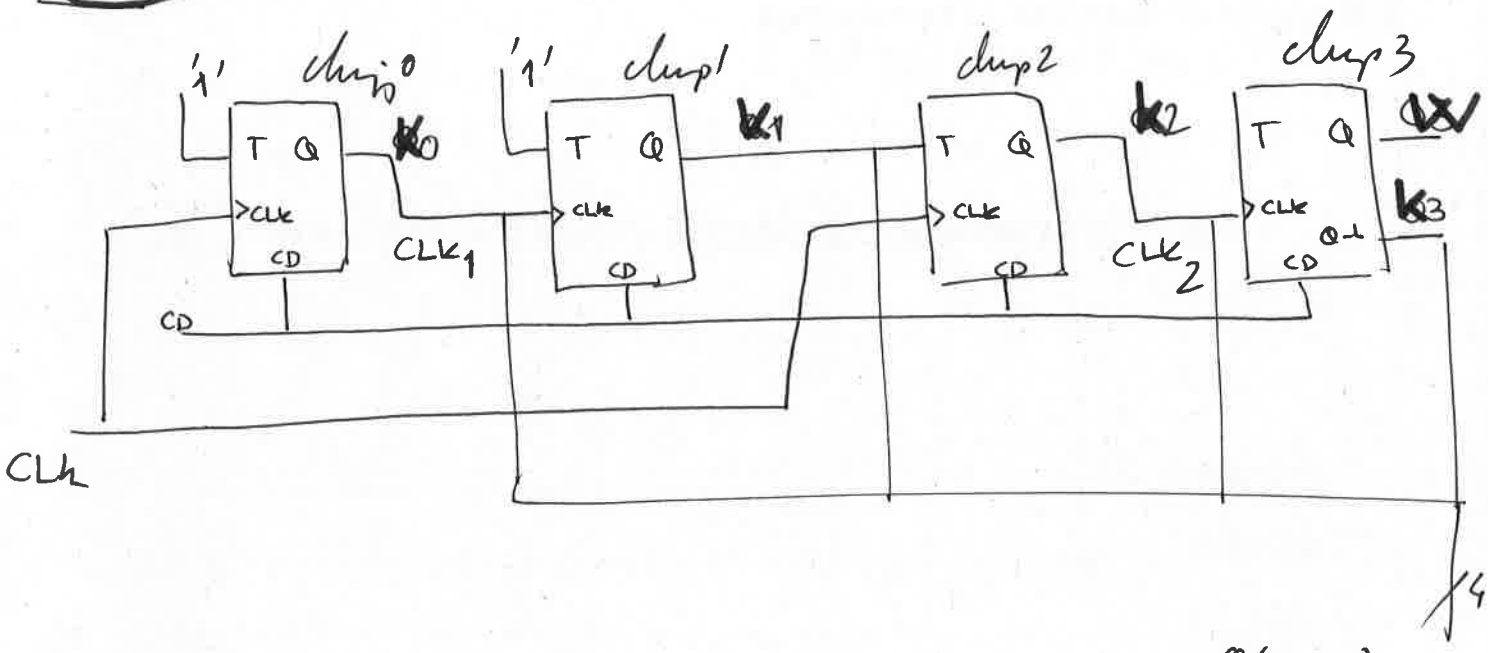
$$f_{max} \leq \frac{1}{15.2\,ns} = 65.78\ MHz$$

(c)



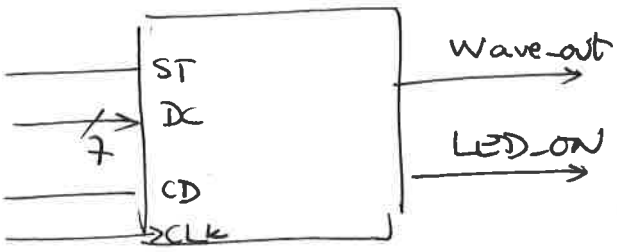
When CD = 1 ⇒ Q = 00000

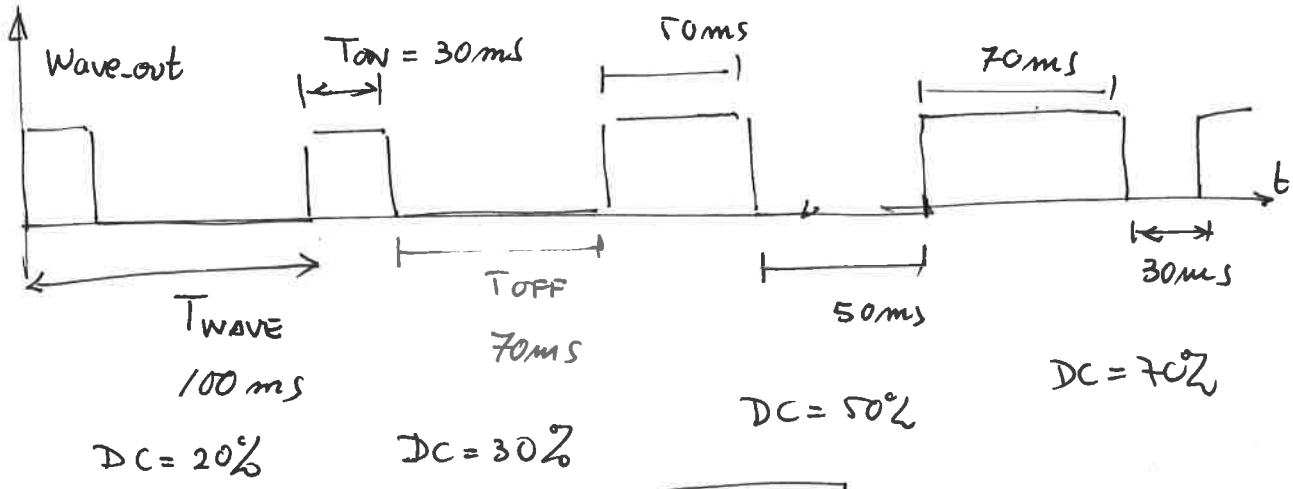
Prob 2

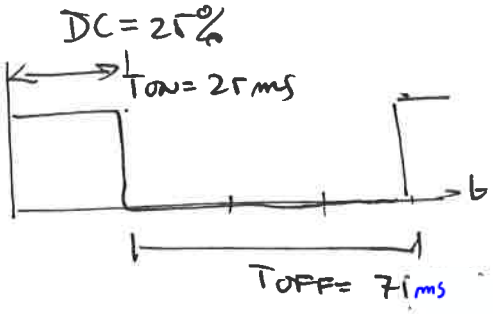


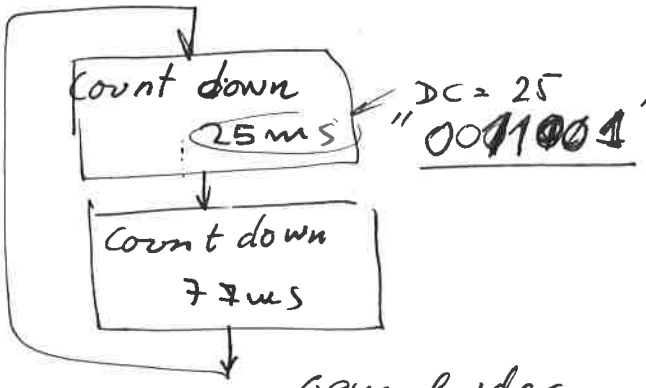
8, 11, 6, 1, 0, 3, 14, 9 ...

Prob 3

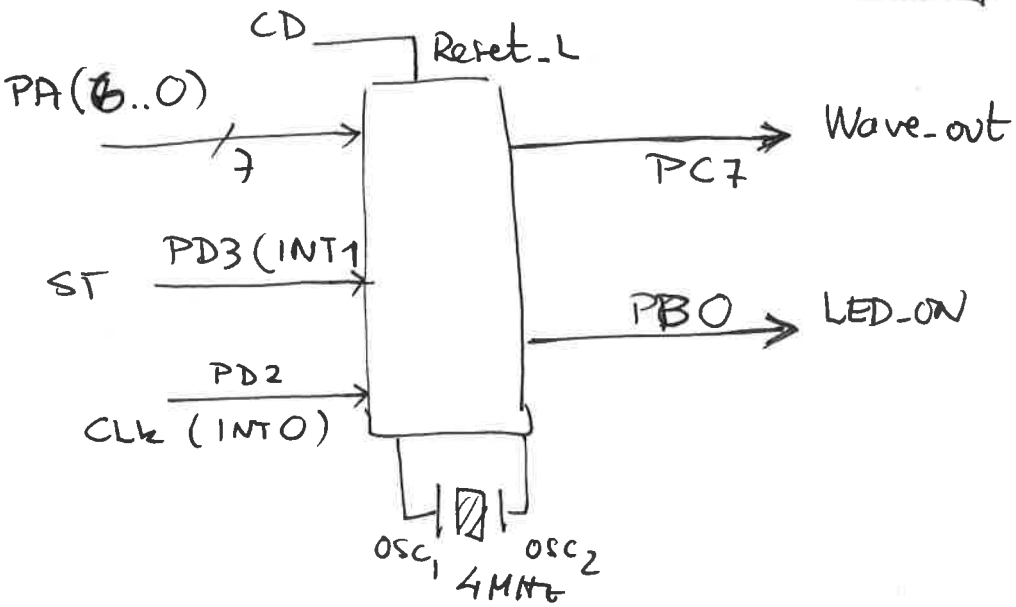




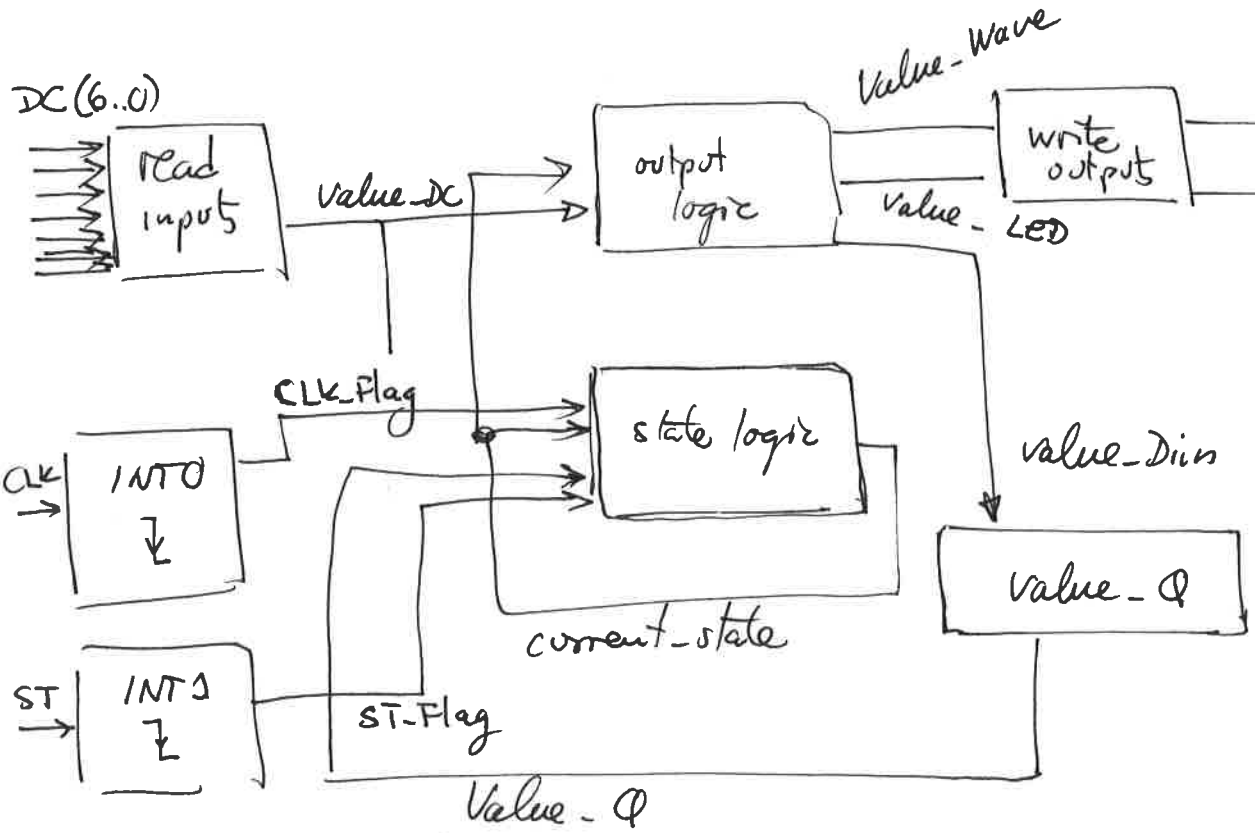


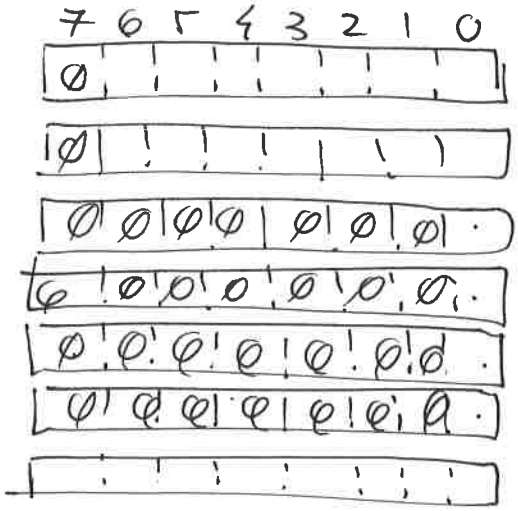


(1)



(2) The basic idea is a FSM that is able to run the system generating the waveforms when the user has clicked the ST button





Value_Q The counter

Value_DC

CLK_Flag

ST_Flag

Value_Wave

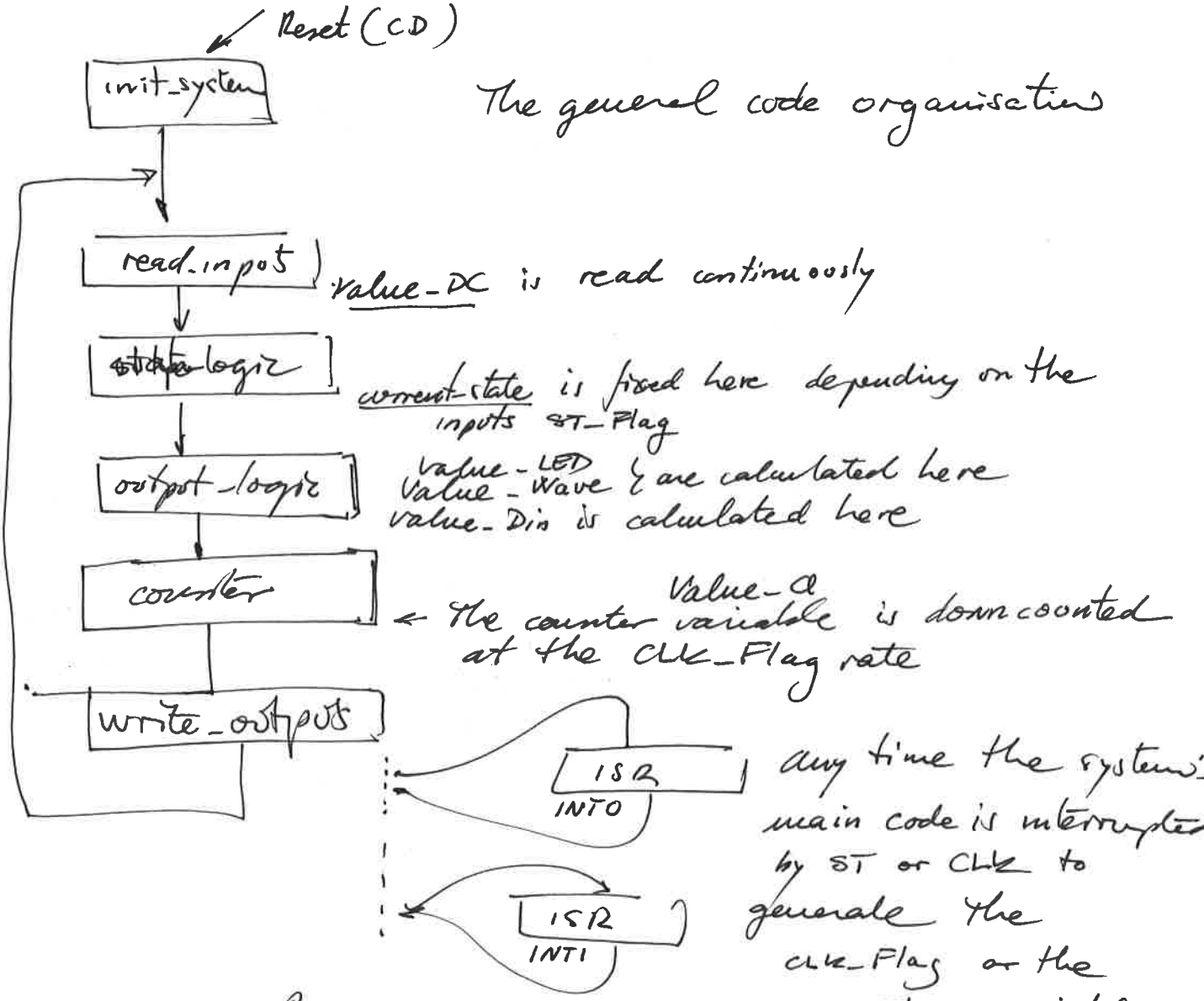
Value_LED

current_state → to hold a value
ASCII, for instance, representing
the state of the FSM

DC pins will be read

CLK, ST will generate interrupts

Value_Wave will be written to the pin Wave_out
Value_LED will be written to the pin LED_ON



The general code organisation

value_DC is read continuously

current_state is fixed here depending on the inputs ST_Flag

Value_LED, Value_Wave & are calculated here
Value_Din is calculated here

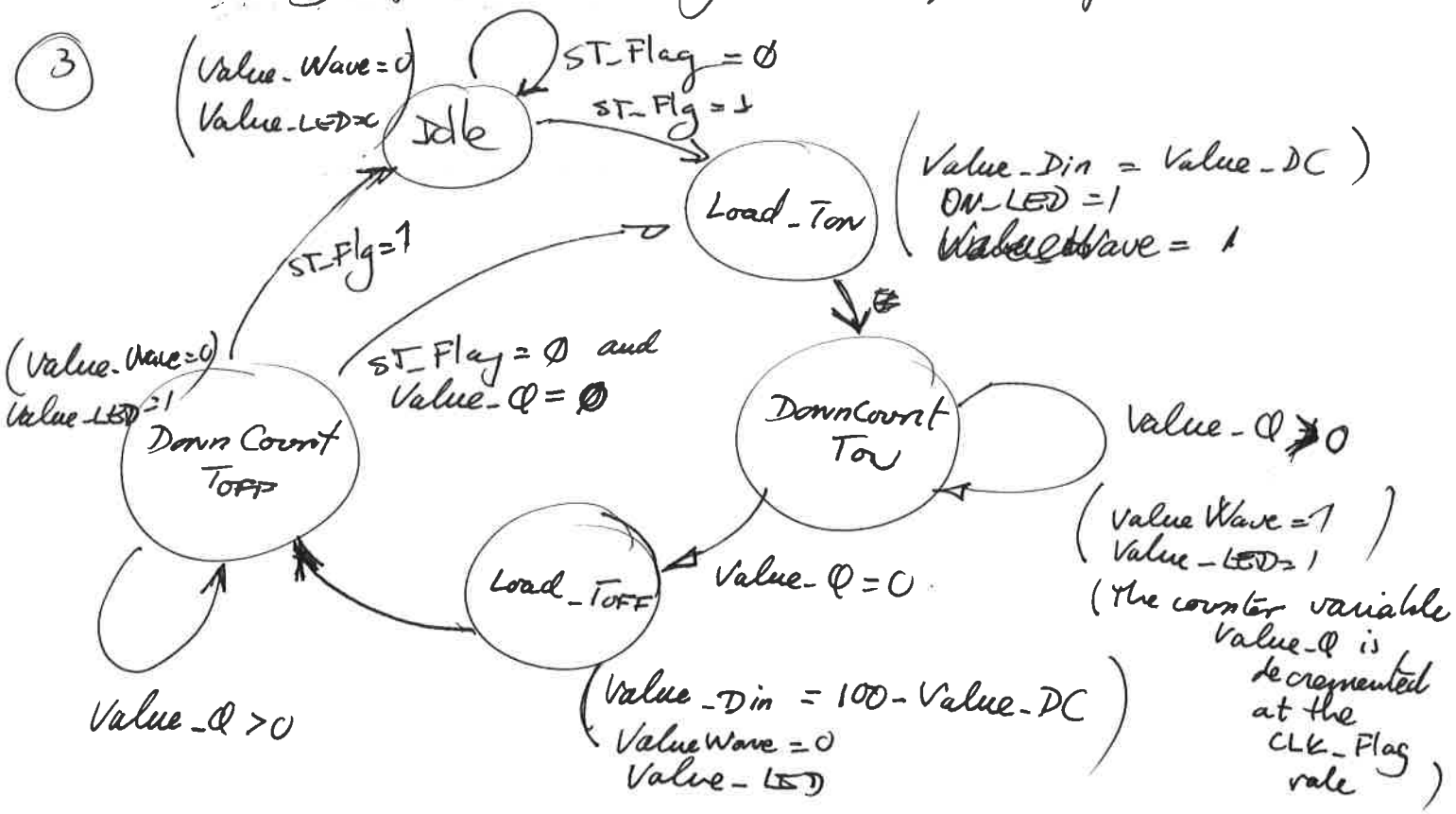
Value_Q
← the counter variable is downcounted at the CLK_Flag rate

any time the system's main code is interrupted by ST or CLK to generate the CLK_Flag or the ST_Flag variable

CLK_Flag for the counter

ST_Flag for controlling the sequence of states

(3)



(4) Initialise an input means writing a '0' at the corresponding DDRx register

Initialise a pin to be an output means writing a '1' at the corresponding DDRx

DDRA = 0b 0 0 0 0 0 0 0 0 all input DC

DDRB = 0b 1 1 1 1 1 1 1 1 all output PORTB0

DDRC = 0b 1 1 1 1 1 1 1 1 all output

PC7 = output

Wave_out

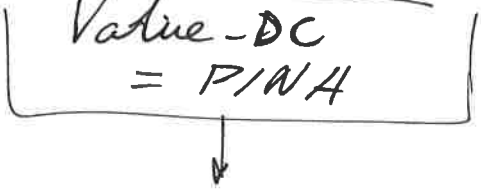
DDRD = 0b 1 1 1 1 0 0 1 1

7 6 5 4 3 2 1 0

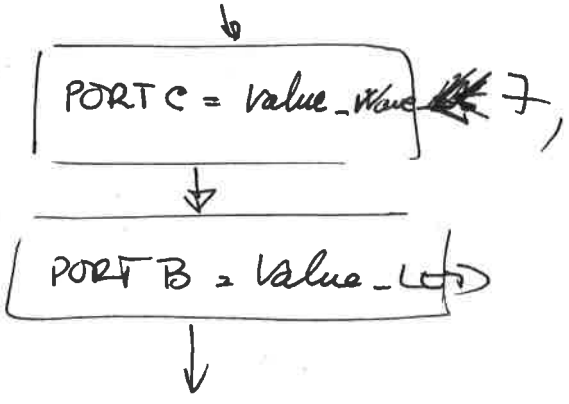
Input CLK

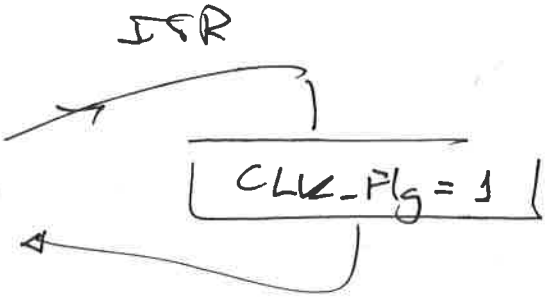
Input ST

(5) read_input



write_output





(6)

current_state	ST_CLK	Value_Q	current_state+
Idle	0	x	Idle
Idle	1	x	Load_TON
Load_TON	x	x	DownCount_TON
Down_Count_TON	x	0	Load_TOFF
Down_count_TON	x	>0	Down_Count_TON
Load_TOFF	x	x	DownCount_TOFF
DownCount_TOFF	1	x	Idle
DownCount_TOFF	0	0	Load_TON
DownCount_TOFF	0	>0	DownCount_TOFF

current_state	Value_RED	Value_Wave	Value_Q
Idle	0	0	x
Load_TON	1	1	Value_DC
Down_Count_TON	1	1	(It is decremented every INTERRUPT CLK Flag)
Load_TOFF	1	0	100 - Value DC
DownCount_TOFF	1	0	(It is decremented every interrupt CLK_Flag