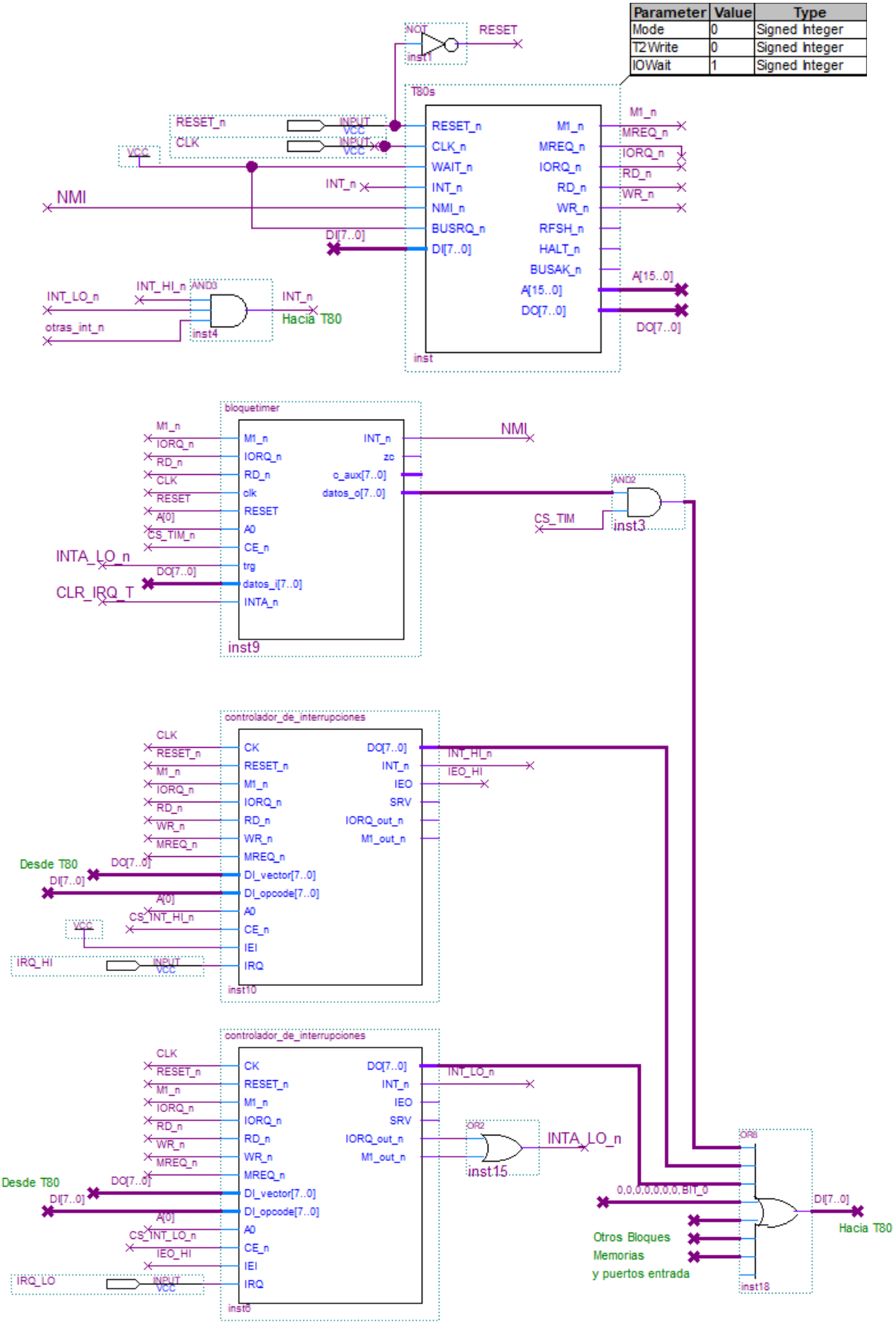
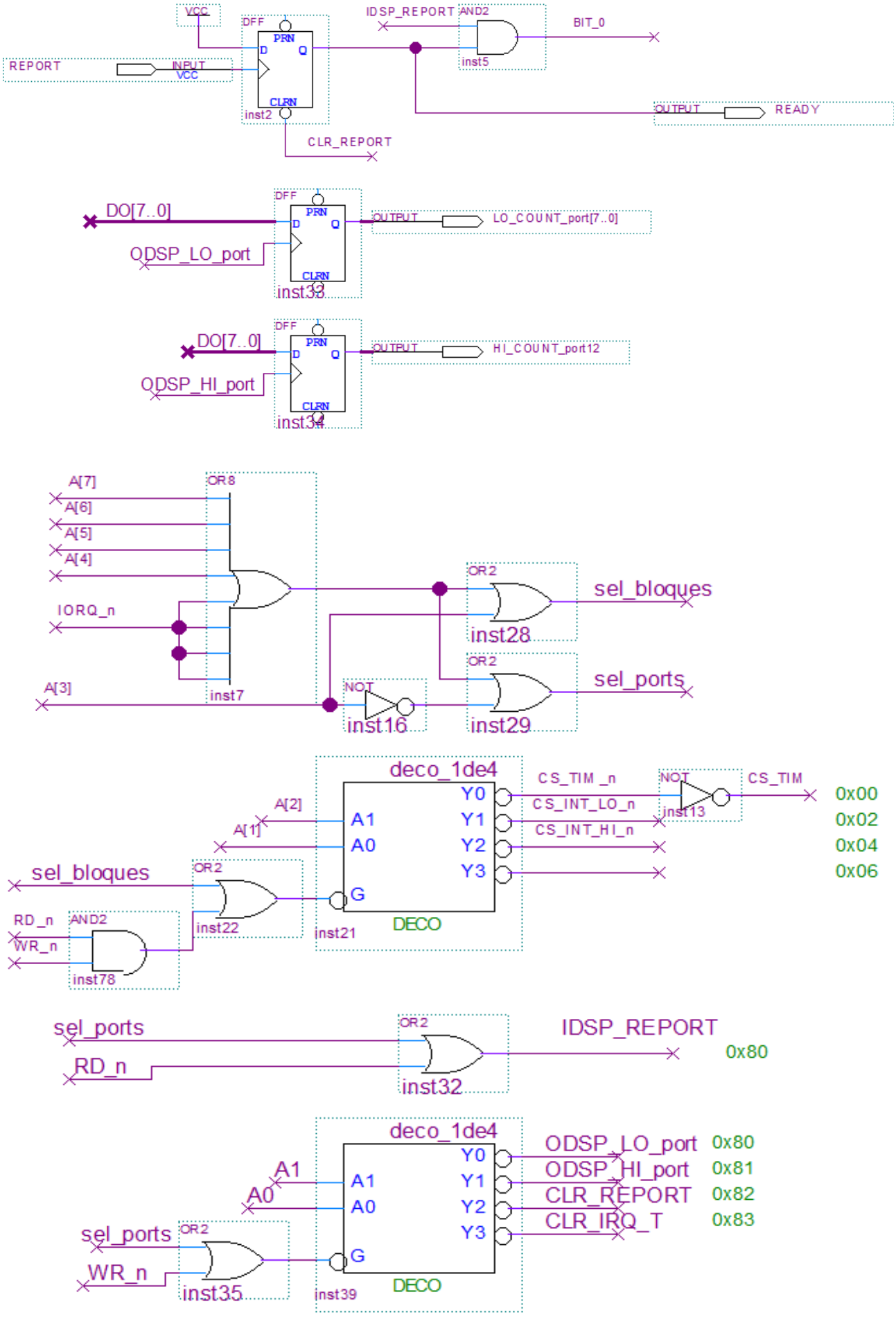
PROBLEMA 1 - a) Hardware




VCC
DFF
PRN
D
Q
CLRN
inst2
REPORT
INPUT
VCC
IDSP_REPORT
AND2
inst5
BIT_0
OUTPUT
READY
CLR_REPORT
DO[7..0]
ODSP_LO_port
OUTPUT
LO_COUNT_port[7..0]
inst33
DO[7..0]
ODSP_HI_port
OUTPUT
HI_COUNT_port12
inst34
A[7]
A[6]
A[5]
A[4]
IORQ_n
A[3]
OR8
inst7
OR2
sel_bloques
inst28
OR2
sel_ports
inst29
NOT
inst16
deco_1de4
A[2]
A[1]
A1
A0
G
Y0
Y1
Y2
Y3
DECO
inst21
CS_TIM _n
CS_INT_LO_n
CS_INT_HI_n
NOT
inst13
CS_TIM
0x00
0x02
0x04
0x06
sel_bloques
OR2
inst22
RD_n
WR_n
AND2
inst78
sel_ports
RD_n
OR2
inst32
IDSP_REPORT
0x80
deco_1de4
A1
A0
sel_ports
WR_n
OR2
inst35
G
Y0
Y1
Y2
Y3
DECO
inst39
ODSP_LO_port 0x80
ODSP_HI_port 0x81
CLR_REPORT 0x82
CLR_IRQ_T 0x83

b) isr_timer

```
; isr_timer_timer:
; preservar estado
; si (estado_lo != en_serv {
;    reprogramar timer
; else{
;    incrementar cnt_lo
;    si estado_hi == en_serv {
;       incrementar cnt_hi
;    }
; }
; borrar peticion bloque timer
; restaurar estado
; retn
   org 0x66  ;; nmi
isr_timer:
   push af
   push hl
   in a, (INT_LO+1)
   and a, 00000011B
   cp E_SERV
   jr z, elseIf_lo
   ld a, CW_TIMER
   out (TIM+1), a
   jr finIf_lo
elseIf_lo:
   ld hl, cnt_lo
   inc (hl)
   in a, (INT_HI+1)
   and a, 00000011B
   cp E_SERV
   jr z, finIf_lo
   ld hl, cnt_hi
   inc (hl)
finIf_lo:
   out (CLR_IRQ_T), a
   pop hl
   pop af
   retn
```

c) rutina HANDLE_REPORT

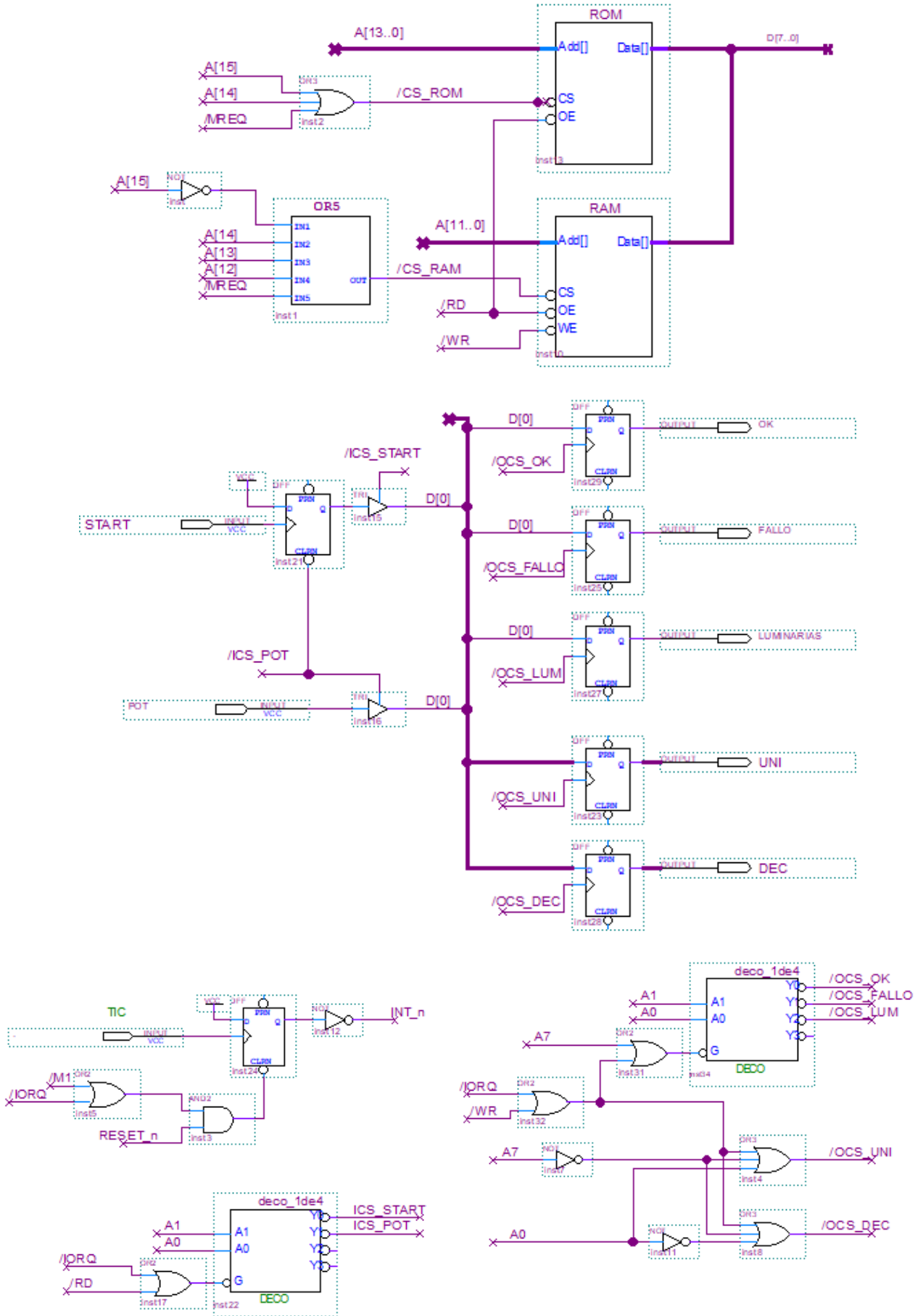
```
; si FF_report {
;    leo contadores
;    escribo puertos
;    reinicio contadores
;    borro FF_report y genero flanco
READY
; }
handle_report:
   in a, (FF_REPORT)
   bit 0, a
   jr z, finsi_report
   ld a, (cnt_hi)
   out (HI_port), a
   ld a, (cnt_lo)
   out (LO_port), a
   ld a, 0
   ld (cnt_hi), a
   ld (cnt_lo), a
   out (CLR_REPORT), a
finsi_report:
   ret
```

d) inicialización, reservas

```
TIM     EQU 0x00
INT_LO  EQU TIM+2
INT_HI  EQU TIM+4

FF_REPORT  EQU 0x08
LO_port    EQU 0x08
HI_port    EQU LO_port+1
CLR_REPORT EQU LO_port+2
CLR_IRQ_T  EQU LO_port+3

E_IDLE  EQU 1
E_PEND  EQU 2
E_SERV  EQU 3

;; ei, soft reset, arranque por trigger
 ;; pre = 6, 2^6=64
 ;; cte = 8, div = (8+1)x64
CW_TIMER  EQU 11110110B
CTE_TIMER EQU 8

;; sección .text a partir 0x0000
.text
   org 0
   ld sp, 0
   im 2
   ld hl, tabla_int
   ld a, h
   ld i, a
   ld a, 0
      ;; controladores int
   out (INT_HI), a
   out (INT_HI+1), a
   ld a, 2
   out (INT_LO), a
   out (INT_LO+1), a
      ;; timer
   ld a, CTE_TIMER
   out (TIM), a
   ld a, (CW_TIMER)
   out (TIM+1), a
      ;; variables y puertos
   ld a, 0
   ld (cnt_hi), a
   ld (cnt_lo), a
   out (LO_port), a
   out (HI_port), a

   call init_gral
   ei
   jp main

   org 0x66  ;; nmi
isr_timer:
   ;; ya incluido en parte b
handle_report:
   ;; ya incluido en parte c

main:
   ;;; loop de prog. Ppal existente
```

```
init_gral:
   ;;; inic. resto sistema

isr_hi:
   ;;; isr int.  alta prioridad

isr_lo:
   ;;; isr int. baja prioridad

   org 0x0200
tabla_int:
   DW isr_hi
   DW isr_lo

;; sección .data a partir de 0x8000
.data
cnt_hi: DB 0
cnt_lo: DB 0

.end
```

PROBLEMA 2 - a) Hardware



16K Rom (0000h a 3FFFh) - 4K Ram (8000h a 8FFFh)

Dir	I	O
00h	START	OK
01h	POT	FALLO
02h		LUM
80h		UNI
81h		**DEC**

==========

b)

```
; TICS A CONTAR
; 60min*4 tics/min + 10 = 250 ,
;     alcanza con 1 byte

ST_FIN                  equ 00h
ST_ENCENDIDO            equ 01h

TIC_1HORA               equ 240
TIC_FIN                 equ 250


; incremento cuenta de TICs
; si cuenta de TICs = TIC_FIN
;      estado = ST_FIN
; sino si cuenta de TICs >= TIC_1HORA
;      pongo en salida LUMINARIAS valor
;                    de ESTADO_LUM
;      ESTADO_LUM = !ESTADO_LUM
;
; divido cuenta de TICs en 4
; actualiza dsiplay 7 segmentos


 org 38h
rutint:
  push AF
  push BC

  ld A,(TIC_COUNT)
  inc A
  ld (TIC_COUNT),A

  cp TIC_FIN
  jp NZ, verifico_1hora ; no llegue a
                  ;fin, estoy en 1hora?

fin_ensayo:
  ld A,ST_FIN
  ld (ESTADO),A
  jp mostrar_hora

verifico_1hora:
  cp TIC_1HORA  ; TIC_COUNT - TIC_1HORA
  jp M, mostrar_hora ; no transcurrio
                     ; la primera hora

ciclado:
   ld A,(ESTADO_LUM)
   cpl
   out (LUMINARIAS), A
   ld (ESTADO_LUM), A

mostrar_hora:
  ld A,(TIC_COUNT)
  srl A
  srl A  ; A= minutos transcurridos
  call RUT_BINA7SEG
  ld A, C
  out (UNIDADES),A
  ld A, B
  out (DECENAS),A

  pop BC
  pop AF
  ei
  reti
```

==========

c)

```
TOPE_RAM        EQU 8FFFh
BASE_RAM        EQU 8000h

PORCENTAJE_ADMISIBLE equ 90
LUM_ON          EQU     0xFF
LUM_OFF          EQU    0

 org BASE_RAM
 ESTADO             DB
 TIC_COUNT          DB
 ESTADO_LUM         DB
 POTENCIA_UMBRAL    DB

 org 0000h
 ld SP, TOPE_RAM+1 ; ini. SP y modo int
 im1
 call rut_inicializo_sistema

espero_start:
 in A,(START_F)
 bit 0,A
 jp Z, espero_start

comienzo_ensayo:
 ld A, LUM_ON  ; enc.luminarias
 out (LUMINARIAS), A
 ld (ESTADO_LUM),A
 in A,(POWER)        ;mide pot. inicial
 ld B, PORCENTAJE_ADMISIBLE
 call percent
 ld (POTENCIA_UMBRAL),A ; umbral = 90%
                        ; pot inicial

 ld A,ST_ENCENDIDO
 ld (ESTADO),A
```

```
 ei                 ; comienza cuenta

 ld A, (POTENCIA_UMBRAL)
 ld B,A

monitoreo_hasta_fin_ensayo:
 ld A,(ESTADO)
 cp ST_FIN
 jp Z, ok    ; fin ensayo sin fallas

 ld A,(ESTAD_LUM)
 cp LUM_ON
 jp NZ, monitoreo_hasta_fin_ensayo

 in A, (POT)
 cp B         ; pot_medida - pot_umbral
 jp M, fallo  ; negativo -> fallo

 jp  monitoreo_hasta_fin_ensayo

ok:
 ld A,1
 out (OK),A
 jp fin

fallo:
 ld A,1
 out (FALLO),A

 fin:
 di
 ld A, APAGAR_LUM
 out (LUMINARIAS),A
```

```
espero_start1:
 in A,(START_F)
 bit 0,A
 jp Z, espero_start1

 call rut_inicializo_sistema

jp comienzo_ensayo


rut_inicializo_sistema:
 ld A,0
 out (OK), A      ; apago OK y FALLO
 out (FALLO), A
 ld (TIC_COUT), A  ; reseteo cuenta TICs
 call RUT_BINA7SEG ; pongo a 0 display
 ld A, C
 out (UNIDADES),A
 ld A, B
 out (DECENAS),A
 in A,(POT)       ; borro FF start
 ld A,LUM_OFF
 out (LUMINARIAS), A ; apago luminarias
 ret
```