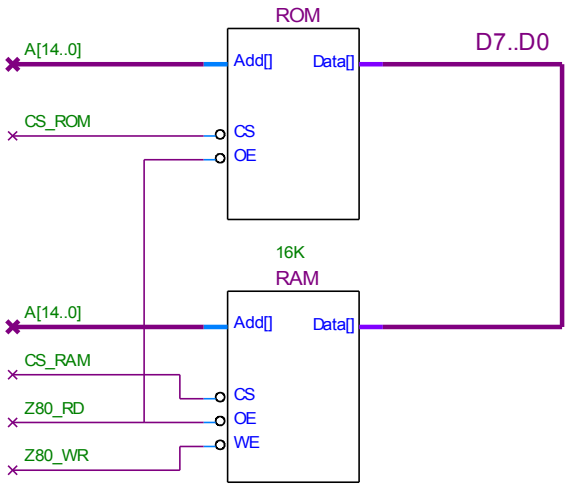
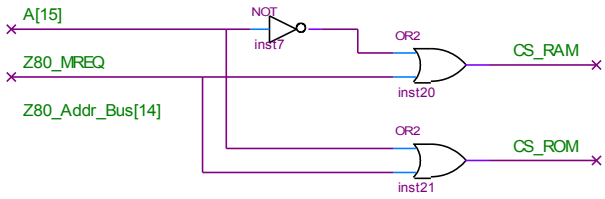
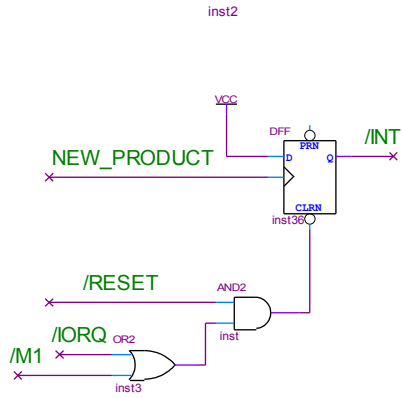
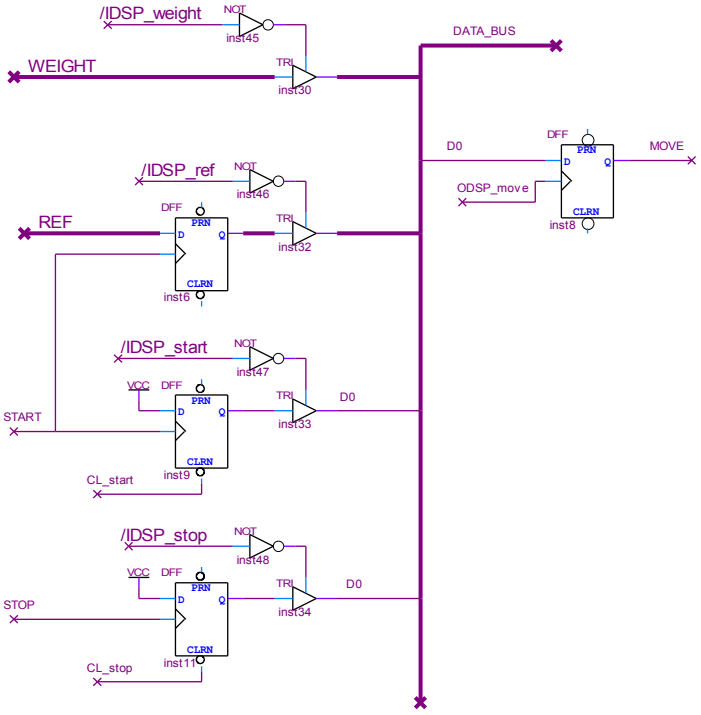
PROBLEMA 1

a) Hardware

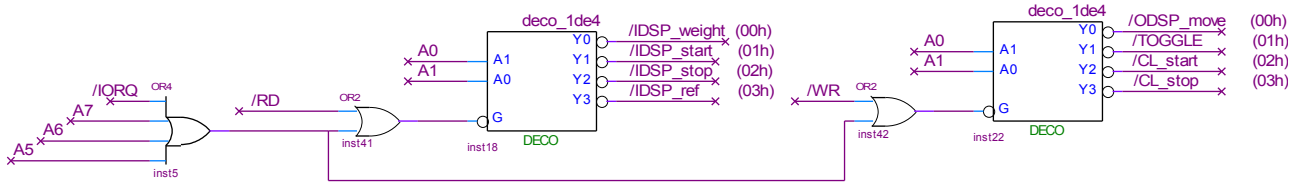








rango	A7A6A5A4	A3A2A1A0
20h a 2Fh	0 0 1 0	x x x x
C0h a DFh	1 1 0 x	x x x x
Puertos a agregar (A7A6A5 = 000)	0 0 0 x	x x x x



b)Inicialización y programa principal

```
IDSP_WEIGHT     EQU 0
IDSP_ST         EQU 1
IDSP_SP         EQU 2
IDSP_REF        EQU 3
ODSP_MOVE       EQU 0
ODSP_TOG        EQU 1
ODSP_ST         EQU 2
ODSP_SP         EQU 3

ORG 0x0000
        LD SP, 0
        LD A, 0
        OUT (ODSP_MOVE), A
        OUT (CL_ST), A
        OUT (CL_SP), A
        LD (posicion), A        ;0x00 -> A, 0xFF -> B
        IM 1
        EI
loop_st:
        CALL otras_tareas
        IN A, (IDSP_ST)         ;loop en espera de start
        BIT 0, A
        JP Z, loop_st
        OUT (CL_ST), A          ;borro ff
        IN A,(IDSP_REF)
        LD (ref), A             ;leo y guardo en memoria ref
        LD A, 0xFF
        OUT (ODSP_MOVE), A      ;prendo cinta
loop_sp:
        CALL otras_tareas
        IN A, (IDSP_SP)         ;loop en espera de stop
        BIT 0, A
        JP Z, loop_sp
        OUT (CL_SP), A          ;borro ff
        LD A, 0
        OUT (ODSP_MOVE), A      ;apago cinta
        JP loop_st
ORG 0x8000
ref:            DB
posicion:       DB
```

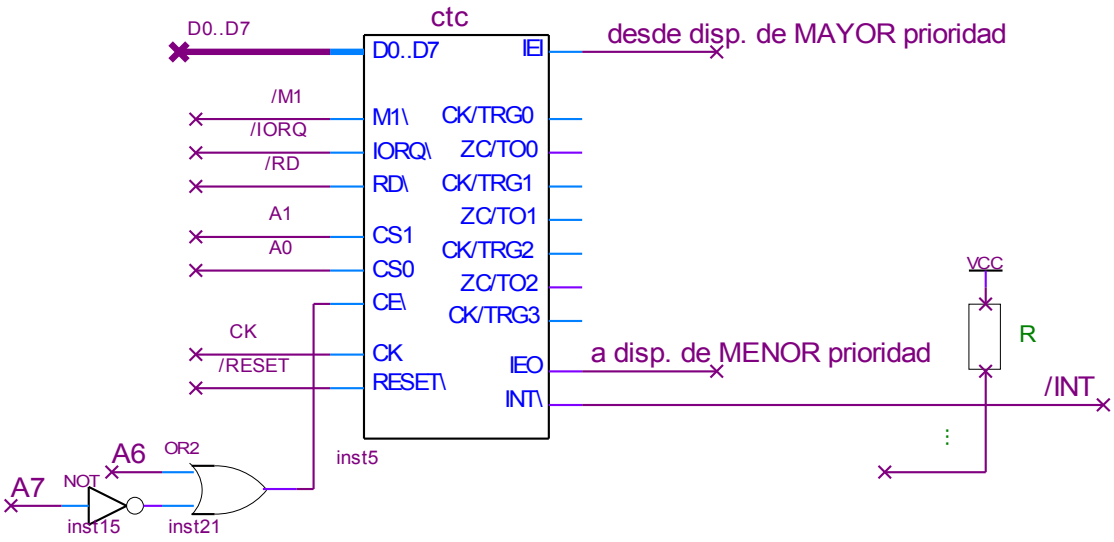
c)Rutina atención a interrupción

```
ORG 0x0038
        PUSH AF
        PUSH BC
        IN A, (IDSP_WEIGHT)
        LD B, A
        LD A, (ref)
        CP B                    ;referencia - peso
        JP C, mayor             ;si es negativo salto
menor:  LD A, (posicion)
        BIT 0, A                ;si posicion = 0x00 ya estoy en A y no tengo que
        JP Z, fin               ;hacer nada
        JP toggle               ;si no voy a mover el brazo
mayor:  LD A, (posicion)
        BIT 0, A                ;si posicion = 0xFF ya estoy en B y no tengo que
        JP NZ, fin              ;hacer nada
toggle:                         ;si no muevo el brazo
        CPL                     ;complemento posicion
        LD (posicion), A
        OUT (ODSP_TOG), A       ;muevo brazo
fin:    POP BC
        POP AF
        EI
        RET
```

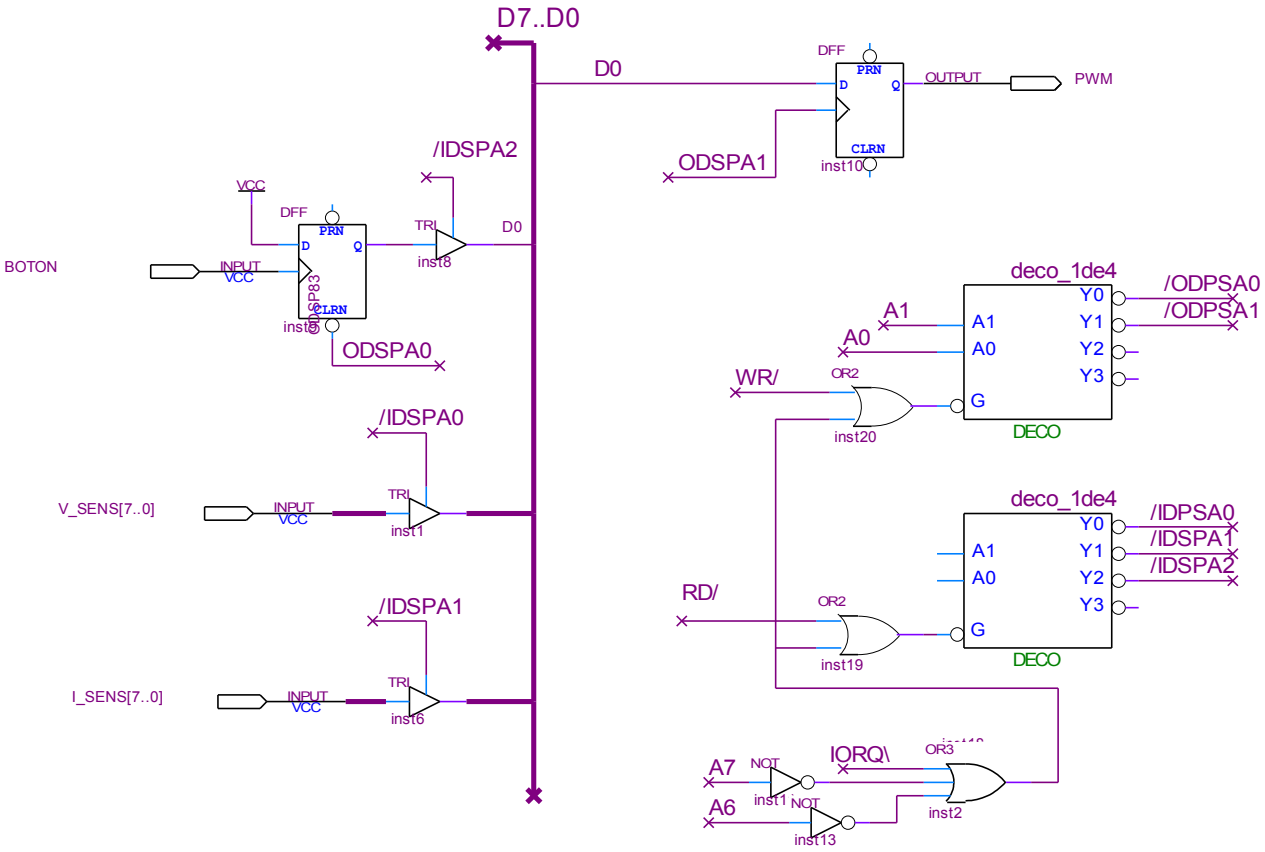
PROBLEMA 2

a) Hardware

CTC a partir de dirección 0x80



Puertos (entrada botón con FF, entradas i_sens y v_sens, salida pwm) desde la dirección 0xA0



b) Inicialización y reserva memoria

```
CTC0     EQU  0x80
CTC1     EQU  0x81
V_SENS   EQU  0xA0
I_SENS   EQU  0xA1
boton    EQU  0xA2
cl_boton EQU  0xA0
PWM      EQU  0xA1

IREF     EQU  85
VFLOT    EQU  120
n_ton_min EQU 10
APAGADO  EQU 0
        ; PRENDIDO = not APAGADO
PRENDIDO EQU 0xFF

vector   EQU 0
   ;canal0 ei, timer, pre=256,
autotrg
CWctc0   EQU 1011 0111B
ctectc0  EQU 100
   ;canal1 ídem canal 0
CWctc1   EQU 1011 0111B
   ;detención canal1: di, no sigue
cte
CWdi     EQU 0011 0001B

    ORG 0x2000
tabla:
   DW rutint_periodica
   DW rutint_ton

    ORG 0x8000
n_ton:       db
modo:        db
cont10veces: db

   ORG 0
   ld SP, 0
   im 2
   ld I, tabla/256  ; tabla en rom
   ld a, APAGADO
   ld (modo), a

   ld a, vector     ; vector de int.
   out (CTC0), a
   ld a, CWctc0     ; arranco int
periódica
   out (CTC0), a
   ld a, ctectc0
   out (CTC0), a

   out (cl_boton), a ; borro FF
boton
   ld a, 0           ; inicio pwm=0
   out (PWM), a
   EI
   ; ...
   ; siguen otras inicializaciones
   ; y programa principal
```

c) rutina interrupción periódica

```
;;;;;-----
; rutint_periodica
;   Si llegó pulso botón{
;     complemento modo
;     Si ( modo encendido) {
;       inicializo cont10veces
;       inicializo var n_ton
;     }
;   }
;   Si (modo encendido) {
;     pwm = 1
;     arranco temporizador ton
;     modifico cont10veces
;     Si es la décima int {
;       n_ton = new_n_ton(n_ton)
;       inicializo contador
;     }
;     Si tensión > VFLOT {
;       modo = apagado
;     }
;   }

rutint_periodica:
    ei
     push af
     in a, (boton)
     bit 0, a
     jr z, finsiboton
         ; si boton pulsado{
            ; borro ff boton
     out (cl_boton), a
           ; complemento modo
     ld a, (modo)
     cpl
     ld (modo), a
     or a
     jr z, finsiencendido
           ; si modo encendido{
               ; cont10veces = 10
               ; n_ton = n_ton_min
    ld a, 10
     ld (cont10veces), a
     ld a, n_ton_min
     ld (n_ton), a
finsiencendido:
finsiboton:
    ld a, (modo)
     or a
     jr z, finsiencendido2
        ; si modo encendido{
            ; pwm = 1
     ld a, 0xFF
     out (pwm), a
        ; arranco ton (cte = n_ton)
     ld a, CWctc1
     out (CTC1), a
     ld a, (n_ton)
     out (CTC1), a
```

```
        ; dec cont10veces
      dec (cont10veces)
      jr nz, finsicont
         ; si cont10veces = 0{
                ; actualizo n_ton
      ld a, (n_ton)
      call new_n_ton
      ld (n_ton), a
              ; inic cont10veces
      ld a, 10
      ld (cont10veces), a
finsicont:
    in a, (vsens)     ; leo tension
     cp a, VFLOT
     jr nc, finsivsens
        ; si V > VFLOT{
            ;modo = APAGADO
     ld a, APAGADO
     ld (modo), a
finsivsens:
finsiencendido2:
    pop af
     reti


;;;;;-----
; la subrutina new_t_on que
; se suponía dada podría ser asi:
;new_n_ton() {
;   leo corriente
;   si I != IREF{
;      si I < IREF{
;         n_ton = n_ton + 1
;         si n_ton > n_ton_max{
;            n_ton = n_ton_max
;         }
;      }else{
;         n_ton = n_ton - 1
;         si n_ton < n_ton_min{
;            n_ton = n_ton_min
;         }
;      }
;    }
; }
actualizo_n_ton:
    in a, (isens)   ; leo corriente
    cp a, IREF
    jr z, finsidistinto
      ; si I != IREF{
    jr nc, else
         ; si I < IREF{
           ; n_ton = n_ton + 1
           ; saturando en n_ton_max
    ld a, (n_ton)
    inc a
    cp a, n_ton_max
    jr c, finsaturomax
    ld a, n_ton_max
finsaturomax:
    jr finsimenor
else:
           ; n_ton = n_ton - 1
           ; saturando n_ton_min
     ld a, (n_ton)
     dec a
     cp a, n_ton_min
     jr c, finsaturomin
     ld a, n_ton_min
finsaturomin:
finsimenor:
finsidistinto:
     ret
```

d) interrupción Ton

```
;;;;;-----
rutint_ton:
   ei
   push af
   ld a, 0       ; pwm = 0
   out (pwm), a
   ld a, CWdi     ; detengo
interrupciones ton
   out (CTC1), a
   pop af
   reti
```