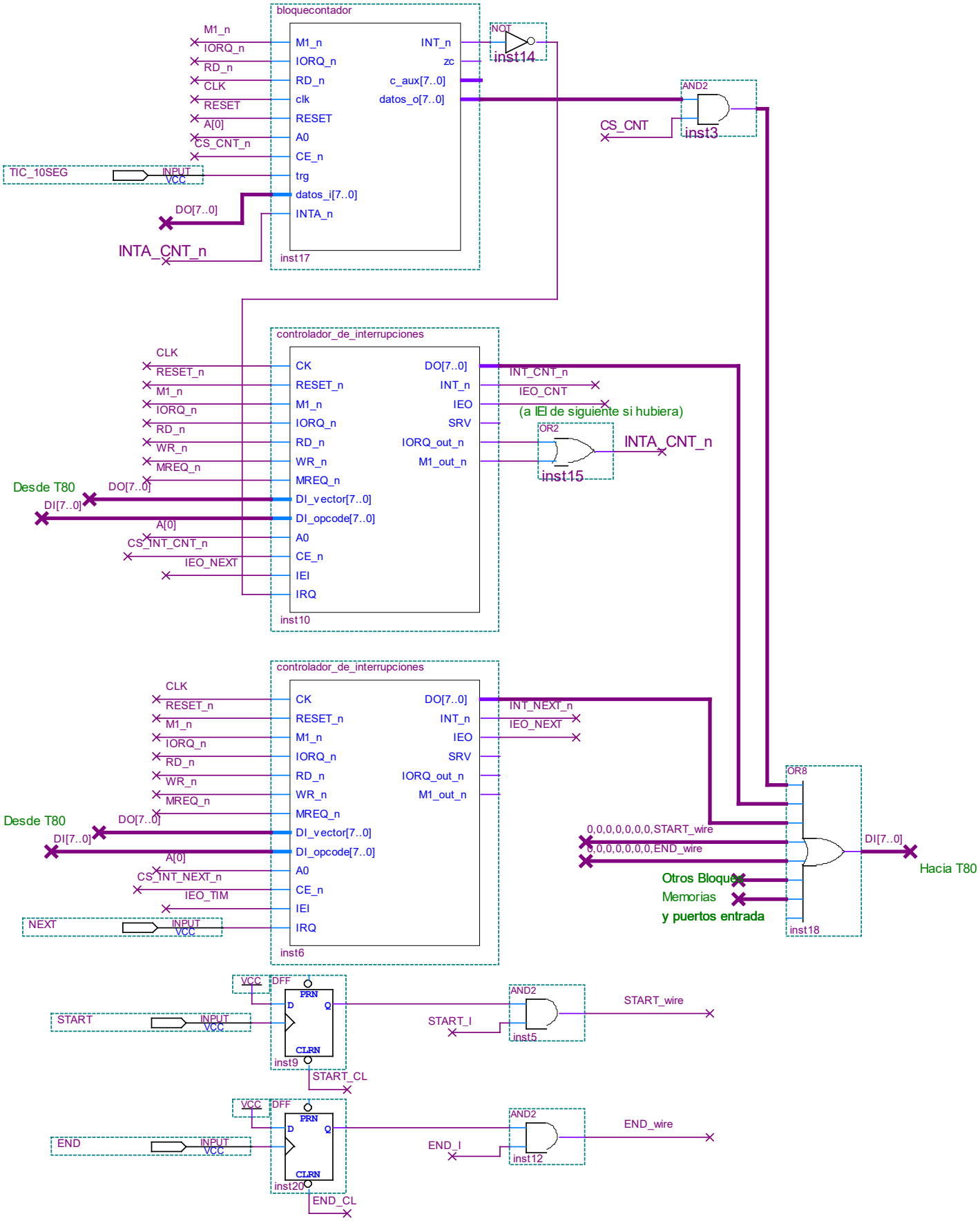
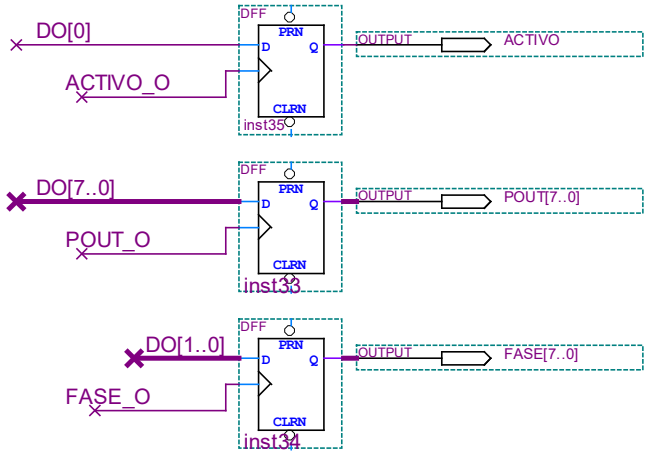
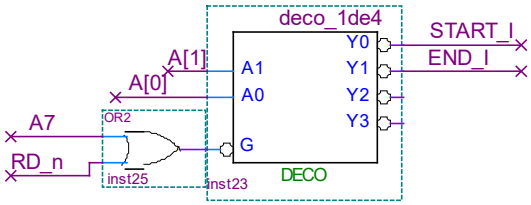
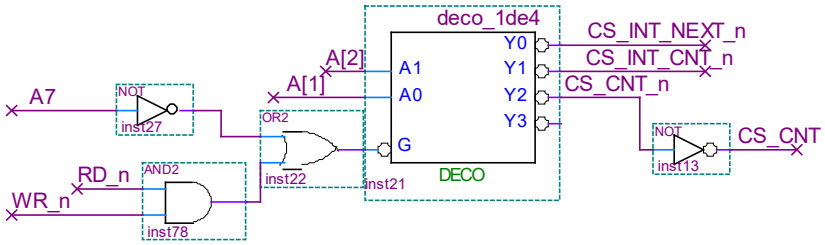
PROBLEMA 1

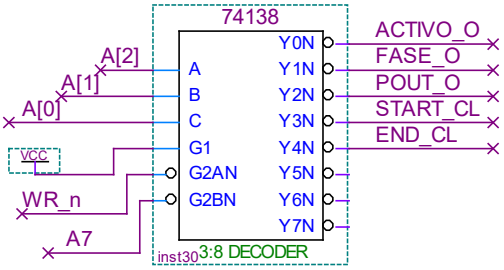
a) Hardware

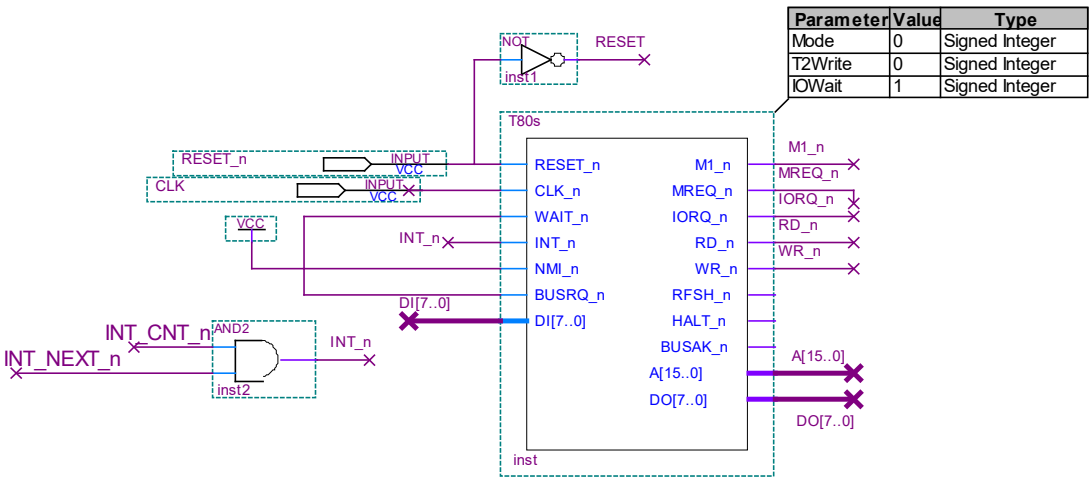












Parameter	Value	Type
Mode	0	Signed Integer
T2Write	0	Signed Integer
IOWait	1	Signed Integer

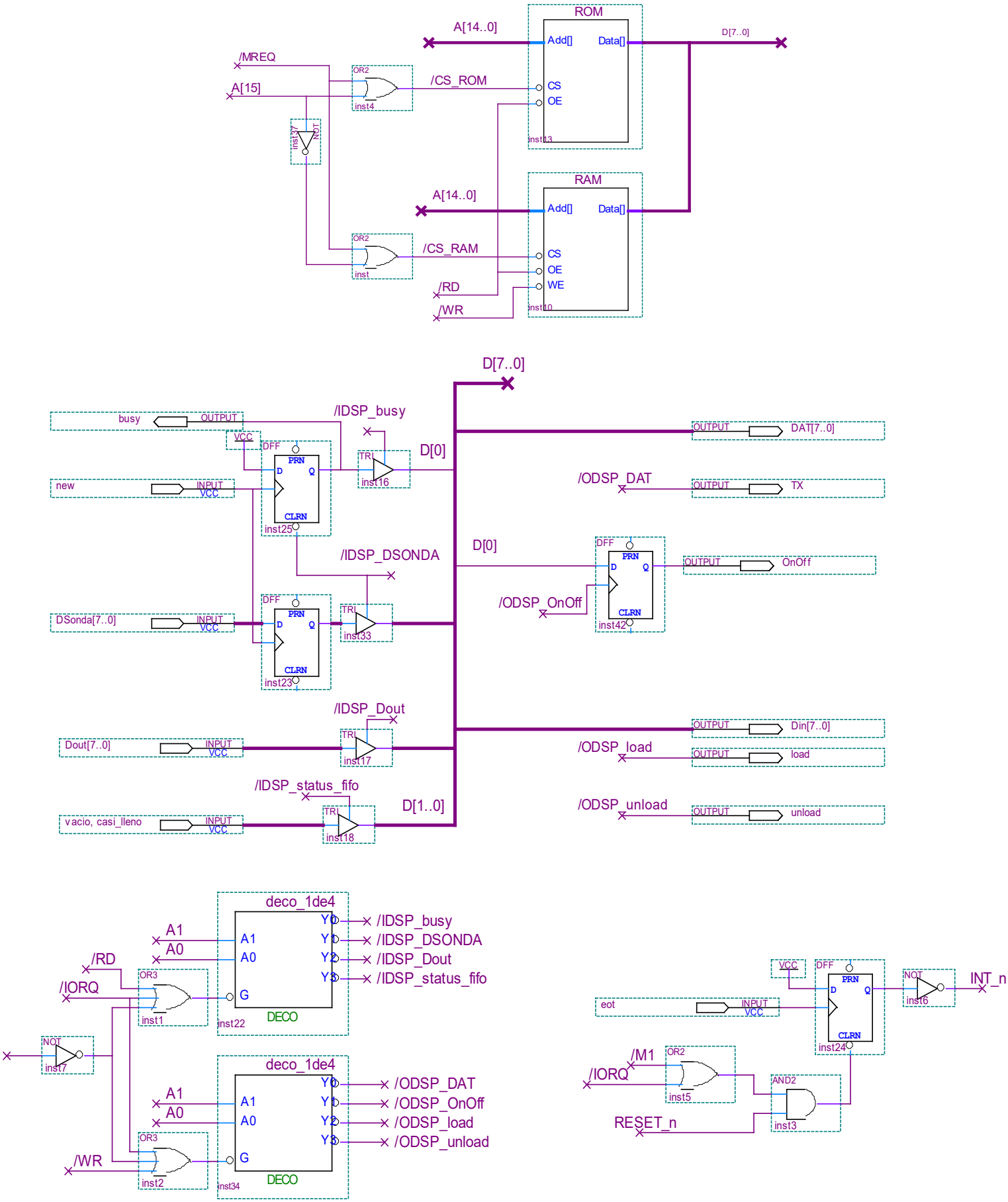
```
;Puerto de entrada
START_I  equ 0x00
END_I    equ 0x01

;Puerto salida
ACTIVO_O equ 0x00
FASE_O   equ 0x01
POUT_O   equ 0x02
START_CL equ 0x04
END_CL   equ 0x05

;Perifericos
CI_NEXT_VI_AD equ 0x80
CI_NEXT_CL_AD equ 0x81
CI_CONT_VI_AD equ 0x82
CI_CONT_CL_AD equ 0x83

CONT_CTE_AD equ 0x84
CONT_CW_AD  equ 0x85
```

```
b)
org 0x1000
rutint_next:
  ei
  push AF

  ld A,(ESTADO)
  cp ESTADO_DETENIDO
  jp Z, fin_rutint_next

  call rut_siquiente_fase

fin_rutint_next:
  pop AF
  reti

rutint_cont:
  jp rutint_next


;---Subrutina auxiliar
rut_siguiente_fase:
  push AF
  push HL

  ld HL, TABLA_FASES
  ld A,(NEXT_FASE)
   sla A ; offset en tabla de fase a ejecutar
  ld L,A

  ld A,(HL)
  cp 0xFF
  jp Z, detenerse

siguiente:
   out (CONT_CTE_AD),A   ; cargo duracion en contador
  inc HL
  ld A,(HL) ; A= VALOR
  out (POUT_O), A   ; escribo valor
  ld A, CONT_EI_CW
    out (CONT_CW_AD), A    ; reinicio contador
  ld A,(NEXT_FASE)
  out (FASE), A  ; escribo fase actual
  inc A
  ld (NEXT_FASE), A
  jp fin_siguiente_fase

detenerse:
  ld A,0xFF
  out (FASE_O)   ; FASE =0xFF
  ld A,ESTADO_DETENIDO
  ld (ESTADO),A  ; ESTADO=detenido
  ld A, 0
  out (POUT), A   ; escribo valor

fin_siguiente_fase:
  pop HL
  pop AF
  ret
```

```
c)
VI_NEXT equ 0x00
VI_CONT equ 0x02

CONT_EI_CW equ 1010 0000; int habilitadas y SW reset

ESTADO_DETENIDO    equ 0xFF
ESTADO_FUNCIONANDO equ 0x00
```

```
org 0x8000 ; comienzo RAM
NEXT_FASE db 0
ESTADO    db 0

org 0x4000
TABLA_FASES ; alineada a página

org 0x2000
TABLA_INT:
 dw rutint_next
 dw rutint_cont

org 0x0000
ld SP, 0x0000 ; tope de RAM + 1
im 2
ld A, TABLA_INT / 256
ld I,A

ld A, 0xFF
out (FASE_O), A
out (START_CL), A
out (END_CL), A
ld A, 0
out (POUT), A

ld A, ESTADO_DETENIDO
ld (ESTADO), A

ld A, VI_NEXT
out(CI_NEXT_VI_AD), A
out(CI_NEXT_CL_AD), A
ld A, VI_CONT
out(CI_CONT_VI_AD), A
out(CI_CONT_CL_AD), A

EI
loop:
 in A, (START_I)
 bit 0, A
 jp NZ, start

 ld A, (END_I)
 bit 0, A
 jp Z, loop

detener:
 out (END_CL), A
 ld A,ESTADO_DETENIDO
 ld (ESTADO), A

 ld A, 0
 out (POUT_O), A
 ld A, 0xFF
 out (FASE_O), A

 jp loop

start:
 out (START_CL), A
 ld A,ESTADO_FUNCIONANDO
 ld (ESTADO), A

 ld A,(TABLA_FASES)
 out (CONT_CTE_AD),A
 ld A,(TABLA_FASES + 1)
 out (POUT_O),A

 ld A,0
 out (FASE), A  ; escribo fase actual
 ld A,1
 ld (NEXT_FASE), A

 ld A, CONT_EI_CW
 out (CONT_CW_AD), A  ; reinicio contador

 jp loop
```

PROBLEMA 2


ROM
A[14..0]
Add[]
Data[]
D[7..0]
/MREQ
A[15]
OR2
inst4
/CS_ROM
CS
OE
inst13
NOT
inst27
RAM
A[14..0]
Add[]
Data[]
OR2
inst
/CS_RAM
CS
OE
WE
/RD
/WR
inst10
D[7..0]
busy
OUTPUT
/IDSP_busy
VCC
DFF
PRN
D
Q
CLRN
inst25
TRI
inst16
D[0]
OUTPUT
DAT[7..0]
new
INPUT
VCC
/ODSP_DAT
OUTPUT
TX
/IDSP_DSONDA
D[0]
DFF
PRN
D
Q
CLRN
inst42
OUTPUT
OnOff
/ODSP_OnOff
DSonda[7..0]
INPUT
VCC
DFF
PRN
D
Q
CLRN
inst23
TRI
inst33
/IDSP_Dout
OUTPUT
Din[7..0]
Dout[7..0]
INPUT
VCC
TRI
inst17
/ODSP_load
OUTPUT
load
/IDSP_status_fifo
D[1..0]
/ODSP_unload
OUTPUT
unload
vacio, casi_lleno
INPUT
VCC
TRI
inst18
deco_1de4
A1
A0
G
Y0
Y1
Y2
Y3
DECO
inst22
/IDSP_busy
/IDSP_DSONDA
/IDSP_Dout
/IDSP_status_fifo
/RD
/IORQ
OR3
inst1
NOT
inst7
deco_1de4
/ODSP_DAT
/ODSP_OnOff
/ODSP_load
/ODSP_unload
/WR
OR3
inst2
DECO
inst34
VCC
DFF
PRN
D
Q
CLRN
inst24
NOT
inst6
INT_n
eot
INPUT
VCC
/M1
/IORQ
OR2
inst5
AND2
inst3
RESET_n

b) todo el software

```
busy   EQU 0x00
DSONDA EQU 0x01
Dout   EQU 0x02
s_fifo EQU 0x03
bit_vacio EQU 1
bit_casi_lleno EQU 0

DAT    EQU 0x00
OnOff  EQU 0x01
load   EQU 0x02
unload EQU 0x03

; .text desde 0x0000
; .data desde 0x8000

.text
;; -- inic --
    ;; stack pointer
   ld sp, 0
    ;; var y puerto OnOff=0
   ld a, 0
   ld (vOnOff), a
   out (OnOff),a
    ;; borrar ff busy
   in a, (busy)
    ;; interrupciones
   im 1
   ei
   jp prog_ppal

;; -- isr --
;
; preservar estado
; si vacío {
;   OnOff = 0
; }
; else{
;   sacar dato de fifo
;   enviarlo a transmisor
; }
; restaurar estado

org 0x0038
isr:
   push af
   in a, (s_fifo)
   bit bit_vacio, a
    ;; si vacío {
    ;;   OnOff = 0
   jr z, else_vacio
   ld a, 0
   ld (vOnOff), a
   out (OnOff), a
   jr fin_isr
else_vacio:
    ;; else{
     ;;   sacar dato de fifo
   in a, (Dout)
   out (unload), a
     ;;   enviarlo a transmisor
   out (DAT), a
fin_isr:
   pop af
   ei
   ret

;; -- prog ppal --

; forever{
;   si nuevo dato sonda{
;     leo DSONDA
;      escribo en FIFO
;   }
;   si (OnOff == 0) {
;     si FIFO casi lleno{
;       OnOff=1
;       sacar dato de fifo
;       enviarlo a transmisor
;      }
;   }
; }

prog_ppal:
   in a, (busy)
   bit 0, a
   jr z, fin_si_busy
    ;; si nuevo dato sonda{
    ;;   leo DSONDA
    ;;   escribo en FIFO
   in a, (DSONDA)
   out (load), a
   ld a, (vOnOff)
   or a
   jr nz, fin_si_OnOff
    ;; si (OnOff == 0) {
   in a, (s_fifo)
   bit bit_casi_lleno, a
   jr z, fin_si_casi_lleno
    ;; si FIFO casi lleno{
    ;;   OnOff=1
   ld a, 0xFF
   out (OnOff), a
   ld (vOnOff), a
    ;;   sacar dato de fifo
   in a, (Dout)
   out (unload), a
    ;;   enviarlo a transmisor
   out (DAT), a
fin_si_casi_lleno:
fin_si_OnOff:
fin_si_busy:
   jr prog_ppal

.data
vOnOff: db 0
```