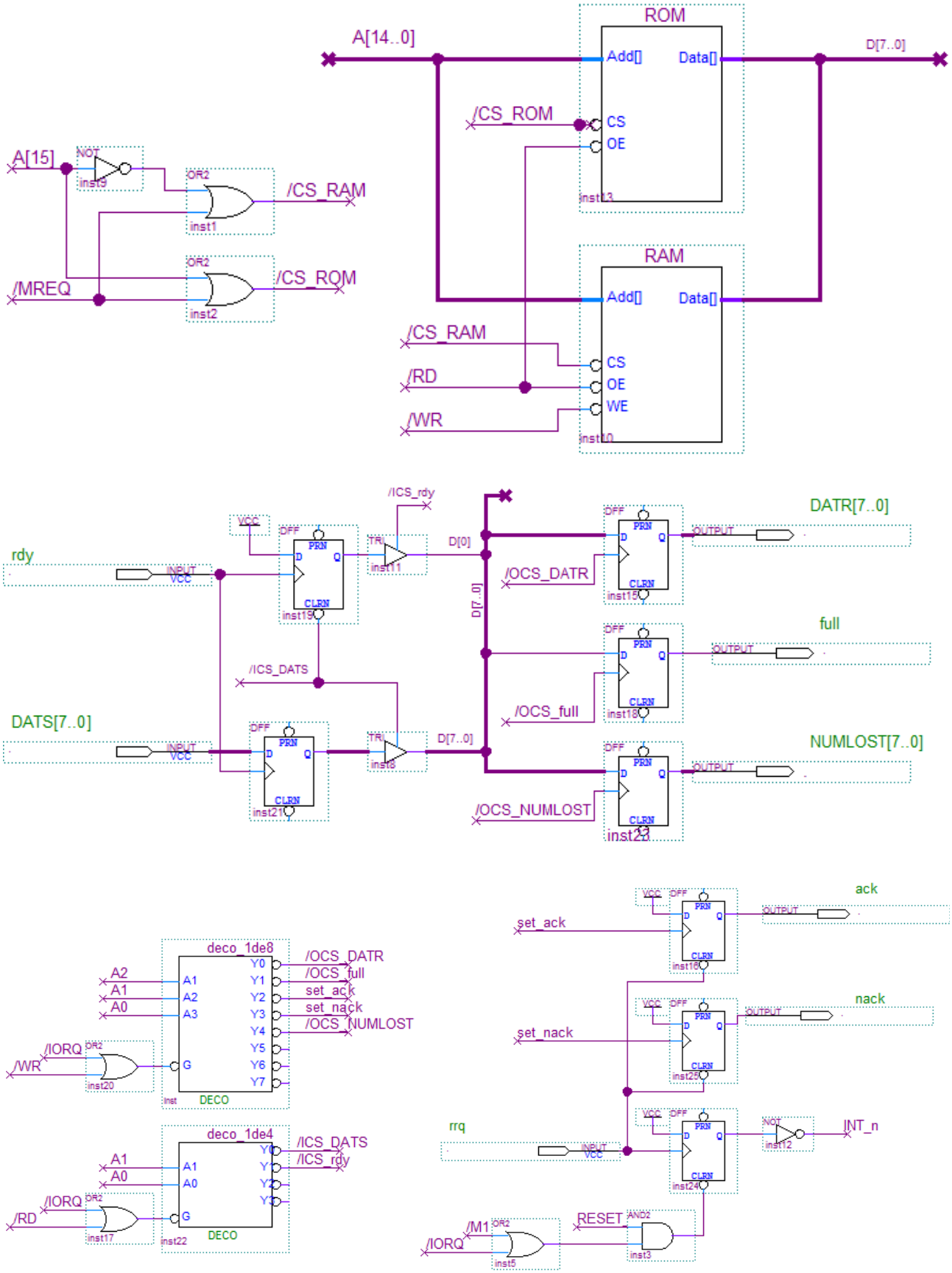
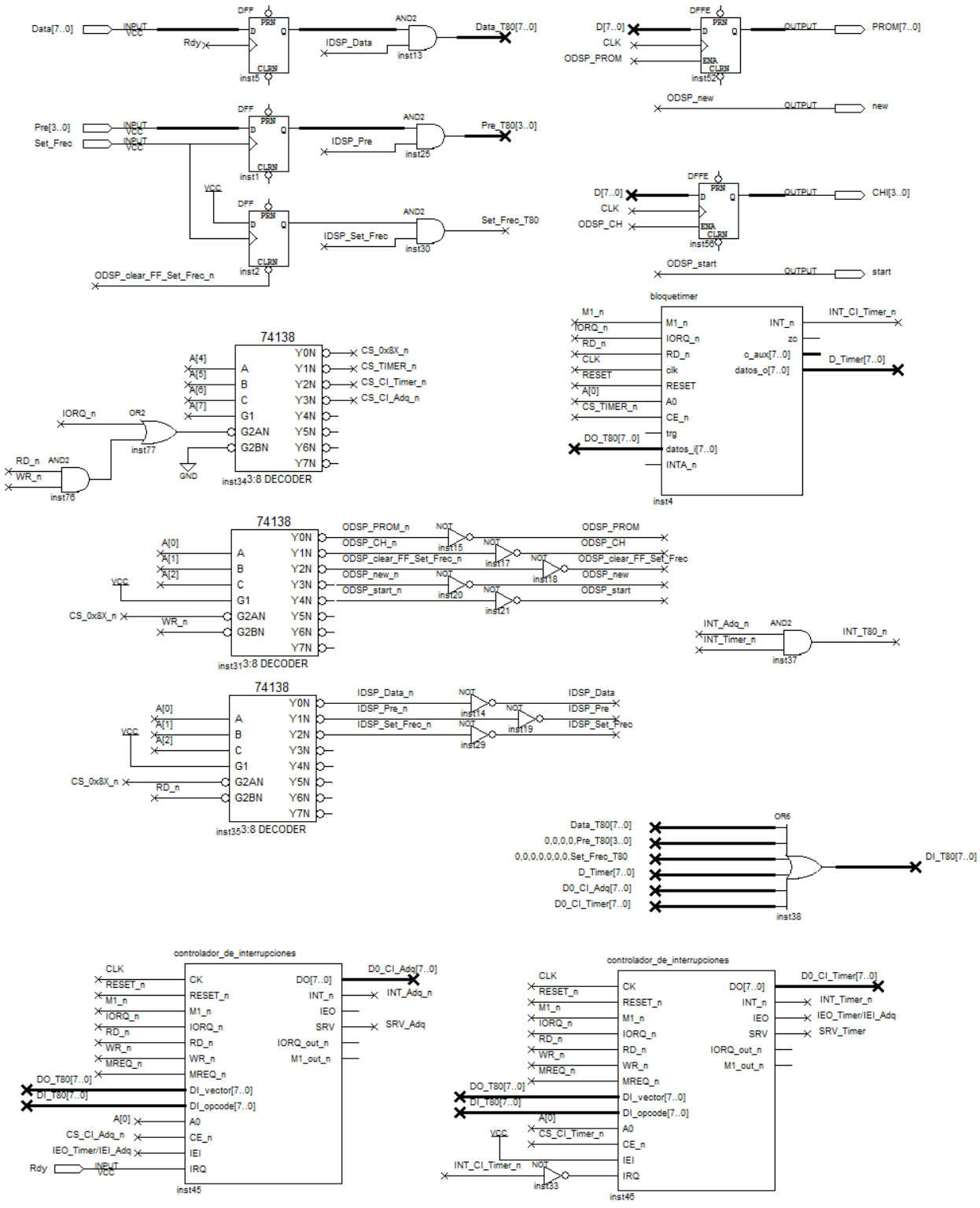
PROBLEMA 1 - a) Hardware



b) Todo el software

```
;;; Puertos entrada
DATS       equ  0 ; memorizado x rdy
rdy        equ  1 ; salida FF rdy
;;; Puertos salida
DATR       equ  0 ; 8 bits
full       equ  1 ; 1 bit
pNLOST     equ  4 ; 8 bits
;;; FFs salida
set_ack    equ  2 ; prende FF, rrq borra
set_nack   equ  3 ; prende FF, rrq borra

; --- Programa principal
; forever {
;     espero rdy
;     leo dats y borro rdy
;     escribo en cola
;     si pude escribir entonces
;         lleno = 0
;     si no
;         lleno = 1
;         si (NLOST != 255 )
;               inc NLOST
; }

      ORG 100H
forever:
espero:
      in a, (rdy)
      bit 0, a
      jr z, espero
      in a, (dats)
      call PutCola
      jr z, exito
      ld a, (NLOST)
      cp 255
      jr z, finsi255
      inc a
      ld (NLOST), a
      out (pNLOST), a
finsi255
      ld a, 1
      jr sigo
exito:
      ld a, 0
sigo: out a, (lleno)
      jr espero

; --- rutina de atencion a interrupcion
; preservar estado
; leer dato de cola
; si hay datos entonces
;     escribo DATR
;     prendo ack
; si no
;     prendo nack
; restauro estado
; habilito interrupciones
; retorno

      ORG 38H
rutint:
      push af
      call GetCola
      jr nz, prendonack
      out (DATR), a    ; escribo dato
      out (setack), a ; prendo ack
      jr finrutint
prendonack:
      out (setnack), a ; prendo nack
finrutint:
      pop af
      ei
      ret

; --- inicialización
; stack
; modo interrupciones
; variables y puertos
; habilitar interrupciones

      ORG 0
      ld sp, 0
      IM 1
      ld a, 0
      ld (NLOST), a
      out (pNLOST), a
      out (DATR), a
      out (full), a
      ei
      jp forever

; --- reserva memoria

      ORG 0x8000
NLOST:  db
```

PROBLEMA 2 – Parte a) Hardware



PROBLEMA 2 – Software

b) Inicialización:

```
.equ  Data 0x80          ;P Entrada
.equ  PROM 0x80          ;P Salida
.equ  CH  0x81           ;P Salida
.equ  Pre 0x81           ;P Entrada
.equ  Set_Frec 0x82      ;P Entrada
.equ  clear_FF_Set_Frec 0x82 ;P Salida
.equ  new 0x83           ;P Salida
.equ  start 0x84         ;P Salida
.equ  Timer_CTE                0x90
.equ  Timer_CTRL               0x91
.equ  Controlador_Timer 0xA0
.equ  Clear_Cont_Timer         0xA1
.equ  Controlador_Adq          0xB0
.equ  Clear_Cont_Adq           0xA1
; Configuración del timer
.equ  cte_Timer   0x7C  ; 125 - 1
;(EI=1, trg=X, swRst=0, trgInit=0,
; Presc=0111)
.equ  word_Timer  0x87
; Direcciones bajas de las ISR
.equ  vector_timer      0x00
.equ  vector_adq        0x02

; Inicialización
.org 0x0000
LD SP,0x0000 ;Ini. puntero al Stack
; Carga de tabla de interrupciones
LD HL,Tabla_ISR
LD I,H
; Inicialización de controladores
LD A,vector_timer
OUT (Controlador_Timer),A
LD A,vector_adq
OUT (Controlador_Adq),A
; Inicialización del Timer
LD A,cte_Timer
OUT (Timer_CTE),A
LD A, word_Timer
OUT (Timer_CTRL),A
; Borrado de peticiones pendientes.
OUT(clear_FF_Set_Frec),A
OUT(Clear_Cont_Timer),A
OUT(Clear_Cont_Adq),A

IM2          ; Modo 2 de interrupciones
EI           ; Hab.de interrupciones
JP ppal      ; Salto al prog. Ppal

.org 0X4000 ;En ROM
Tabla_ISR:
      dw rutint_timer
      dw rutint_adquisidor

.org 0x9000 ;En RAM
contCH:           db
sumaCH:           db   ;Suma canales
```

c) Rutinas de atención a interrupciones

```
.org 0x2000
rutint_timer:
PUSH AF
LD A, 0x00  ; Lectura del primer canal
OUT (CH),A
LD (contCH),A ; Reset cont.de canales
LD(sumaCH),A    ; Reseteo de la suma
POP AF
EI
RETI

rutint_adquisidor:
PUSH AF
PUSH HL
EI
; Lectura de canal solicitado
IN A,(Data)
LD HL,sumaCH ; HL = dirección de la suma
ADD A,(HL)
LD (sumaCH),A
; Incremento del contador
LD A,(contCH)
CP 0x03 ; Verifica si el canal leído fue
el último (0x03)
JP Z,calcProm
INC A
LD (contCH),A
OUT(CH),A   ;Solicitud de sig. canal
OUT(start),A
JP finSecuencia
; Cálculo y act. del promedio calcProm:
LD A,(sumaCH)
SRL A       ; PROM = sumaCH/4
SRL A
OUT(PROM),A
OUT(new),A
finSecuencia:
POP HL
POP AF
RETI
```

d) Programa principal

```
.org 0x1000
ppal:
IN A,(Set_Frec)
CP 0x00
JP Z,ppal
; Seteo de la nueva frecuencia f_adq
LD A, word_Timer
AND 0xF0
LD B,A ; B = parte alta de la palabra
IN A,(Pre)
ADD A,B
LD (Timer_CTRL),A ; Act. Cuenta timer.
OUT(clear_FF_Set_Frec),A  ; Limpia el FF
para escuchar nuevo seteo
JP ppal;
```