- Nombre y CI en cada hoja - Numere las hojas - Indique el total de hojas en la primera - Utilice solo un lado de las hojas	- Incluya un solo problema por hoja - **Sea prolijo** - **Aprobación:** mínimo UN problema

PROBLEMA 1

Se usará un sistema basado en un Z80 con CTC para generar una onda cuadrada de frecuencia f = 100Hz (período T=10 ms) y ciclo de trabajo variable. Para lograrlo se utilizarán 2 canales del CTC y una señal de frecuencia fija 10KHz (10KHz = 100 x 100Hz). El canal 1 se utilizará para medir el período completo y el canal 2 solo el semiperíodo con salida en nivel alto. Estos 2 canales interrumpirán al Z80 el cual se encargará de generar la onda cuadrada en el puerto de salida ONDA de un bit.

El ciclo de trabajo puede ajustarse entre 5% y 95% en incrementos de 1%. Luego de reset debe valer 50%. Para ajustarlo se cuenta con dos entradas UP y DOWN. Un pulso de corta duración en UP deberá aumentar el ciclo de trabajo en 1%, saturando en 95%. Análogamente un pulso de corta duración en DOWN debe decrementar el valor del ciclo de trabajo en 1%, saturando en 5%.

Se pide:

a) Diseño completo del hardware con conexiones del CTC, salida ONDA, entradas UP y DOWN y memoria (32k de RAM y 32k de ROM). Para otras funciones están utilizadas las direcciones 00h a 3Fh con puertos que se supone implementados.
b) Inicialización de todo el sistema (stack, ctc, sistema de interrupciones, variables, puertos que sea necesario). Además de las interrupciones del CTC hay otra interrupción que comienza en la dirección RUTINT_0 y tiene asignado el primer lugar de la tabla de interrupciones. Para inicializar a las otras funciones deberá invocarse la subrutina init_otros().
c) Rutinas de atención a interrupción de los canales del CTC que manejen la salida ONDA y reprogramen los canales del CTC en forma adecuada. El valor del ciclo de trabajo deberá ser leído desde la variable CICLO manejada por la inicialización y el programa principal.
d) Programa principal. En cada loop del programa principal debe invocarse una vez la subrutina otros() y a su retorno se debe monitorear las entradas UP y DOWN y modificar la variable CICLO en caso de ser necesario. Esto debe repetirse en un loop infinito.

PROBLEMA 2

En un sistema basado en un Z80 se desea disponer de un dispositivo que lo interrumpa cada vez que se escribe en un determinado lugar de la memoria. El dispositivo debe ser programable escribiendo 2 bytes en los puertos entrada/salida INT_H e INT_L que corresponderán a la dirección de memoria en la cual se quiere que interrumpa. Los valores escritos en el dispositivo deben poder ser leídos mediante accesos al espacio de entrada. El sistema trabajara en modo 1 ya que este es el único dispositivo que interrumpe.

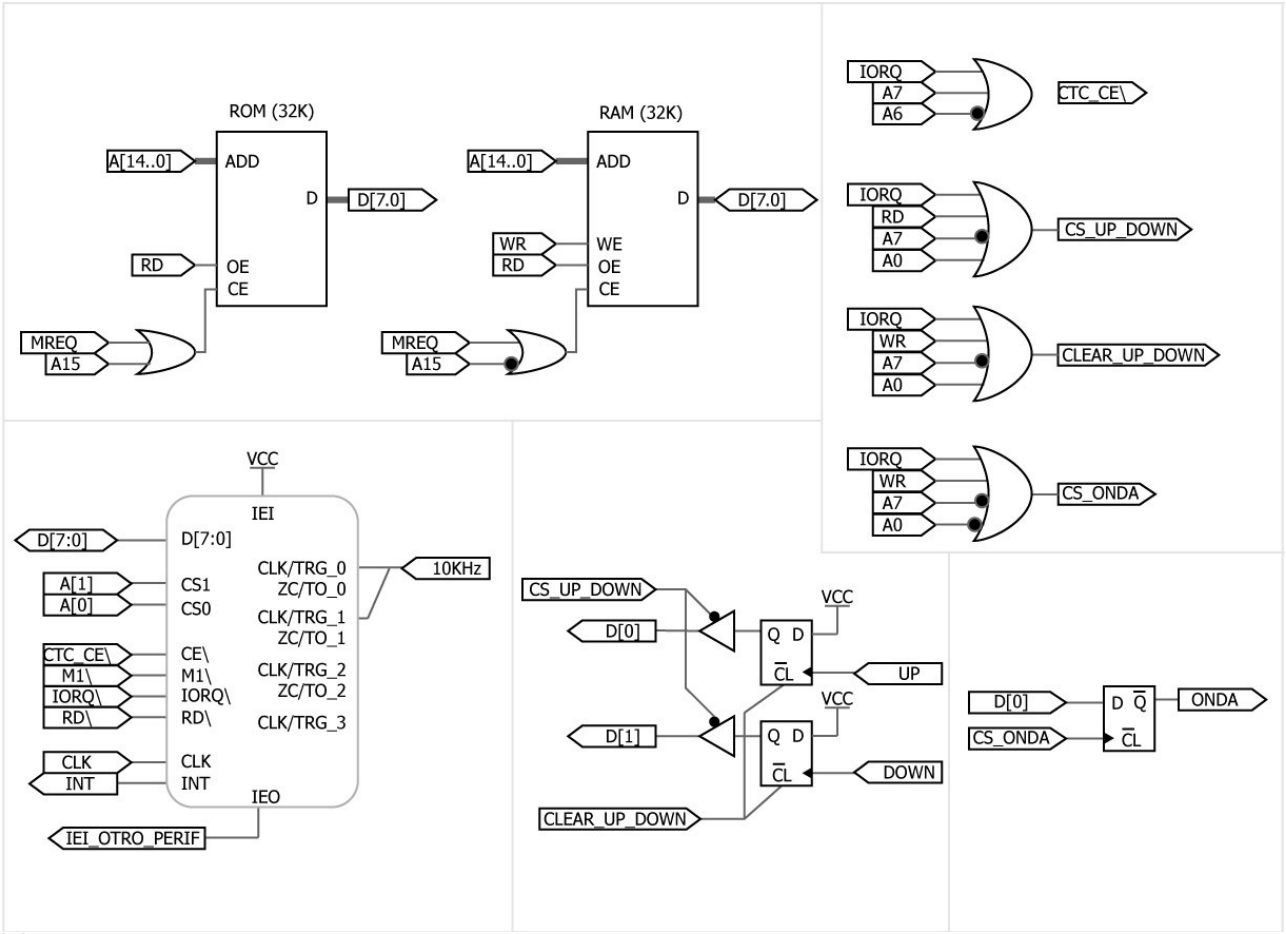
a) Diseño completo del hardware incluyendo:
   - Dispositivo de interrupción programable.
   - Cuatro puertos de salida, SALIDA_i (i=1..4).
   - Memorias del sistema con 32kB de ROM y 32 kB de RAM.
   - Para los puertos de entrada y salida y el dispositivo programable se tiene libre el espacio de entrada salida desde el 90H al CFH.

b) El dispositivo de interrupción programable se utilizará para debuggear un programa, por lo tanto cada vez que el mismo interrumpa se deberá enviar la información del lugar de memoria al que se escribió (el que generó la interrupción) y la dirección de retorno de la interrupción para saber en qué lugar del programa se encuentra. Se pide:

   i) Subrutina armado() que inicializa el dispositivo de interrupción asumiendo que se recibe en HL la dirección de memoria que se utilizará para interrumpir.
   ii) Rutina de atención a interrupciones que mande por los puertos de salida 1 y 2 la dirección que generó la interrupción, y por los puertos 3 y 4 la dirección de retorno de la subrutina de atención a interrupciones. Luego deberá volver al programa principal. No deberá generarse nuevas interrupciones hasta que el sistema se inicialice nuevamente invocando la subrutina armado.

SOLUCIÓN PROBLEMA 1

a)



b)

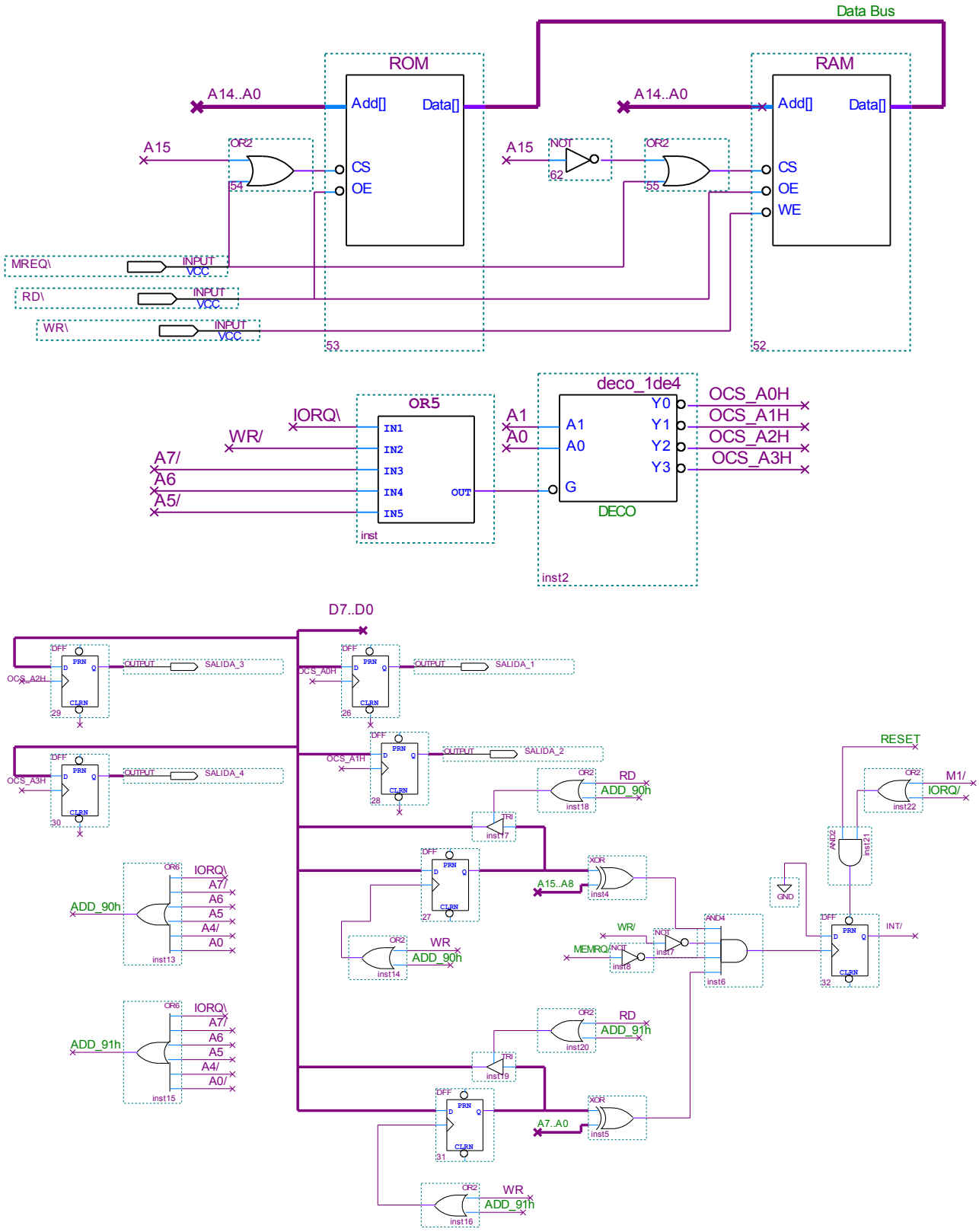
```
CTC_0 equ 40h
CTC_1 equ 41h
ONDA  equ 81h
UP_DOWN equ 80h
CLEAR_UP_DOWN equ 80h


org 4000h
TABLA: dw RUTINT_0    ;00
       dw             ;02
       dw             ;04
       dw             ;06
       dw RUTINT_CTC0 ;08
       dw RUTINT_CTC1 ;A0

org 8000h
CICLO: db


org 00h
ld sp,00h
im 2
ld a,TABLA/256
```

```
ld I,a
ld a,50
ld (ciclo),a

ld a,0000 1000b         ;interrupciones
out (CTC_0),a

ld a,11x1x111b          ;config Puerto 0
out (CTC_0),a
ld a,(CICLO)
out (CTC_0),a

ld a,11x1x111b          ;config Puerto 1
out (CTC_1),a
ld a,100
out (CTC_1),a

call Init_otros
ei
jp PPAL

c)
org 5000h
RUTINT_CTC0:
ei
push af
ld a,0
out (ONDA),a
ld a,01x1x011b          ;deshabilito int. de canal 0
out (CTC_0),a
pop af
reti

RUTINT_CTC1:
ei
push af
ld a,1
out (ONDA),a
ld a, 11x1x111b         ;config Puerto 0
out (CTC_0),a
ld a,(CICLO)
out (CTC_0),a
pop af
reti

d)
org 1000h
PPAL:
call OTROS

in a,(UP_DOWN)         //UP bit 0, DOWN bit 1
out (CLEAR_UP_DOWN),a
ld b,a
bit 0, b
jr z, CHEQUEO_DOWN
AUMENTAR:
ld a, (CICLO)
cp 95
jr z, DISMINUIR
inc a
ld (CICLO),a
```

```
CHEQUEO_DOWN:
bit 1, B
jr z, PPAL
ld a, (CICLO)
cp 5
jr z, PPAL
dec a
ld (CICLO),a

jr PPAL
```

SOLUCIÓN PROBLEMA 2

a)



b)

i)

```
INT_H equ 90h
INT_L equ 91h
```

```
SALIDA_4 equ 0A3h
SALIDA_3 equ 0A2h
SALIDA_2 equ 0A1h
SALIDA_1 equ 0A0h

org EN_ROM
ARMADO:
      push af
      ld a,h
      out (INT_H),a
      ld a,l
      out (INT_L),a
      im 1
      pop af
      ei
      reti
```

ii) ORG 38H

```
      push ix
      ld ix,00h
      add ix,sp    ; ix = sp
      push af
      push hl
      inc ix
      inc ix
      inc ix
      ld a,(ix)
      out (SALIDA_3),a       ;byte bajo direccion retorno
      inc ix
      ld a,(ix)
      out (SALIDA_4),a       ;byte alto direccion retorno
      in a,(int_h)
      out (SALIDA_1),a
      in a,(int_l)
      out (SALIDA_2),a
      pop hl
      pop af
      pop ix
      ret
```