PROBLEMA 1

Se quiere dotar a un sistema basado en Z80 de un periférico BOTON, con un comportamiento similar al del botón principal de un ratón. El nivel de reposo de la señal BOTON es alto, por lo que un flanco de bajada indica que se ha presionado el botón y un flanco de subida que se lo ha liberado. Partiendo de un estado de reposo con el botón liberado, se desea diferenciar entre secuencias de click y de doble click como sigue:

- **click**: se presiona el botón, se libera el botón y transcurren Tdoble ms desde la liberación sin un nuevo evento.
- **doble click**: se presiona el botón, se libera el botón y antes de Tdoble ms vuelve a presionarse el botón.

La funcionalidad del botón se debe agregar a través de interrupciones. El sistema ya cuenta con 32K de ROM, 32K de RAM y otros puertos que ocupan la mitad inferior del espacio de E/S. Se debe incorporar un CTC, un puerto de entrada de 8 bits para leer la constante TICS dada por llaves de configuración del sistema (Tdoble = 256 * TICS * Tck) y dos salidas SIMPLE y DOBLE de un bit cada una para indicar cuál fue la última secuencia detectada.
Cuando se valida la detección de una secuencia deberá activarse a nivel alto la salida que corresponda (SIMPLE o DOBLE) hasta que se vuelva a presionar el botón y comience la detección de la siguiente secuencia.

El CTC se utilizará para generar interrupciones al presionarse el botón (programándolo como contador con cte=1) y para generar interrupciones al vencer el tiempo Tdoble. Para esto último se sugiere programar un canal del CTC en modo timer arrancando el flanco de subida de la señal BOTON.

Se pide:

a) Hardware a agregar al sistema.

b) Rutinas de atención a cada una de las interrupciones utilizadas. En ningún caso una rutina de atención a la interrupción debe quedar esperando por un evento del ratón. Cuando se detecta una secuencia se debe de activar la salida (SIMPLE o DOBLE) e invocar las subrutinas SIMPLE_RUT y DOBLE_RUT que se supondrán ya implementadas.

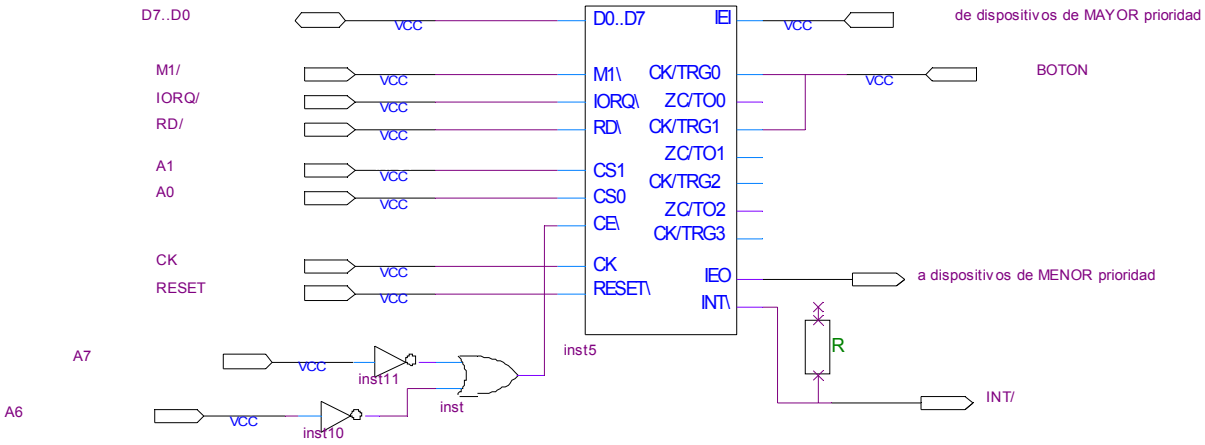
c) Escribir la subrutina **init_boton** que es invocada por la inicialización del sistema como se muestra en el recuadro. Esta subrutina debe inicializar todo lo necesario (CTC, puertos, variables, entradas en la tabla de interrupciones) para que el botón funcione correctamente luego de un reset. Otros dispositivos ya ocupan las 3 primeras entradas de la tabla de interrupciones.

Escribir las directivas de reserva de memoria para todas las variables utilizadas. Se cuenta con memoria libre entre 9000h y A000h.

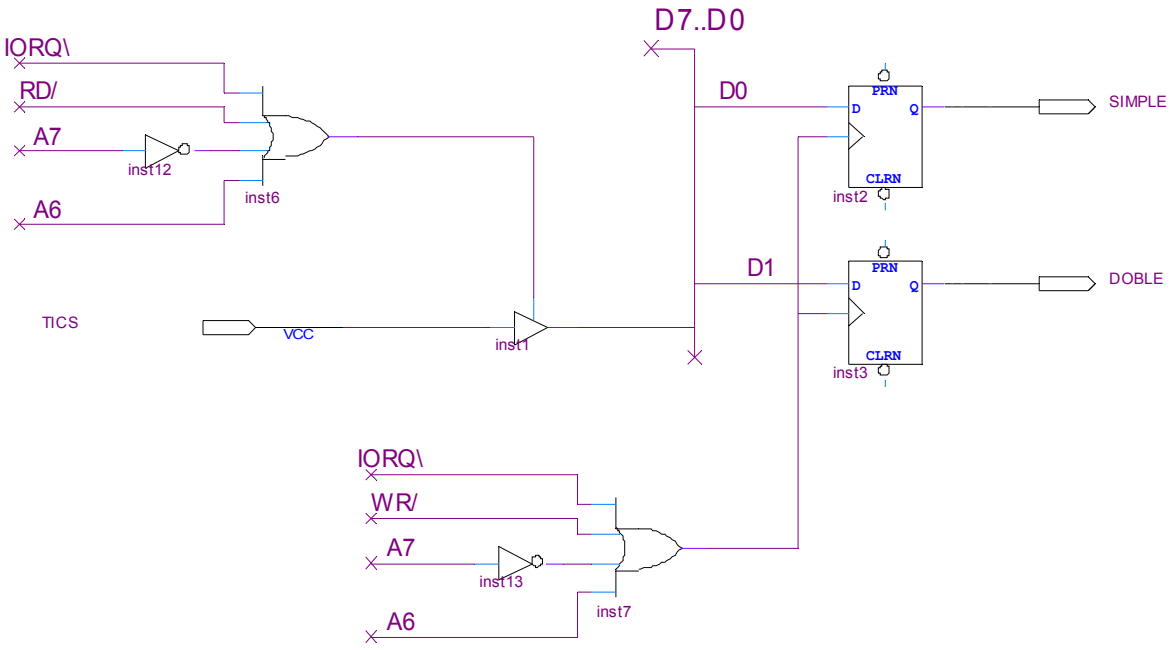
```
        org 0
ini:

   ld sp, 0
   im 2
   ld a, 80h
   ld i, a
   call init_sistema
   call init_boton
   ei
   jp principal
```

a) Hardware a agregar al sistema.

CTC



PUERTOS



parte b)

```
CONTADOR    equ 0C0h
TIMER       equ 0C1h
REPORT      equ 80h
ARMAR_TIMER_CW    equ 1011 1111 ;(EI|TIMER|256|RISING|TRG|CONST|RESET|CW)
DISABLE_TIMER_CW  equ 0011 1111 ;(DI|TIMER|256|RISING|TRG|CONST|RESET|CW)
REPORT_SIMPLE     equ 01h
REPORT_DOBLE      equ 02h
REPORT_CLEAR      equ 00h

org 9000h
NUMERO_FLANCOS_BAJADA db
```

```
org algun_lugar_en_ROM

RUT_FCO_BAJADA:
      ei
      push AF
      ld A, (NUMERO_FLANCOS_BAJADA)
      cp 0
      jp z, PRIMER_FLANCO_BAJADA

SEGUNDO_FLANCO_BAJADA:
; si estoy aqui, hubo un flanco de bajada antes que Tdoble
; deshabilito interrupciones del timer y reporto DOBL
; borro cuenta de flanco
      ld A,0
      ld (NUMERO_FLANCOS_BAJADA), A
      ld A, DISABLE_TIMER_CW
      out (TIMER), A
      out (TIMER), A    ; luego de un SW reset se debe reprogramar CTE
                        ; mando cualquier palabra pues no se va a utilizar
      ld A, REPORT_DOBLE
      out (REPORT), A
      call DOBLE_RUT
      jp RET_RUT_FCO_BAJADA

PRIMER_FLANCO_BAJADA:
;primer flanco de bajada, armo timer para que comience
;a contar con flanco de subida para que cuente Tdoble
; desde flanco de subida
;borro puerto de salida REPORTE
;actualizo cuenta de flancos
      ld A, ARMAR_TIMER_CW
      out (TIMER), A
      in A, (TICS)
      out (TIMER), A
      ld A, REPORT_CLEAR
      out (REPORT), A
      ld A, 01
      ld (NUMERO_FLANCOS_BAJADA), A

RET_RUT_FCO_BAJADA:
      pop af
      reti


RUT_TIMER:
      ei
      push af
; si estoy aqui, hubo un solo click antes que Tdoble
; deshabilito interrupciones del timer y reporto SIMPLE
; borro cuenta de clicks
      ld A,0
      ld (NUMERO_FLANCOS_BAJADA), A
      ld A, DISABLE_TIMER_CW
      out (TIMER), A
      ld A, REPORT_SIMPLE
      out (REPORT), A
      call SIMPLE_RUT

RET_RUT_TIMER:
      pop af
```

```
      reti
```

parte c)

```
INI_CONT_CW       equ 1100 0111B ;(EI|CONT|X|FALLING|X|CONST|RESET|CW)
INI_TIMER_CW      equ 0011 1111B ;(DI|TIMER|256|RISING|TRG|CONST|RESET|CW)
TICS              equ 80h
VECT_CTC          equ 08h

org otro_lugar_ROM

INIT_BOTON:
;actualizo tabla interrupciones (ocupadas 00, 02, 04. Utilizo 08 y 0A)
      ld HL, RUT_FCO_BAJADA
      ld (8008h), HL
      ld HL, RUT_TIMER
      ld (800Ah), HL
;cargo vector de interrupciones
      ld A, VECT_CTC
      out (CONTADOR), A
;programo CTC canal 0 como contador con cte 1
      ld A, INI_CONT_CW
      out (CONTADOR), A
      ld A, 01h
      out (CONTADOR), A
;programo CTC canal 1 como timer sin int con cte TICS
      ld A, INI_TIMER
      out (TIMER), A
      in A, (TICS)
      out (TIMER), A
;borro variable NUMERO_FLANCOS_BAJADA y puertos SIMPLE y DOBLE
      ld A, 0
ld (NUMERO_FLANCOS_BAJADA), A
      OUT (REPORT), A
      ret
```