- 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 desea diseñar un CONTADOR DE PULSOS utilizando un microprocesador Z80.
El sistema debe llevar la cuenta de los flancos de subida en la señal **tic** como un número binario de 16 bits. El valor de la cuenta se podrá llevar a 0 con un pulso de bajada en la entrada **clr** y se desplegará en un display de 7 segmentos de 5 dígitos. El valor de la cuenta se mantendrá actualizado en memoria RAM en la dirección **CUENTA**.

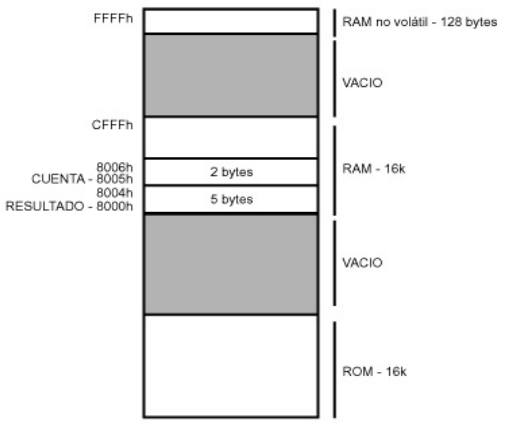
En caso de pérdida de energía no se deberá perder el valor de la cuenta. Para eso se contará con la señal auxiliar **pwrgood** que genera la fuente de alimentación y con un dispositivo de memoria no volátil de 128 bytes.
Cuando se interrumpe la energía la señal **pwrgood** baja a 0 algunos milisegundos antes que la tensión de alimentación caiga a valores inaceptables. Esta señal se utilizará para generar una interrupción no enmascarable en la que se deberá guardar el valor de la cuenta en los dos primeros lugares de la memoria no volátil. Luego de eso el sistema deberá quedar bloqueado a la espera de un reset.

Luego de un reset debe inicializarse el valor de la cuenta desde el valor almacenado en la memoria no volátil. Suponer que la primer vez que se enciende el sistema, la memoria no volatil tendrá todos sus Bytes en 0.

El software deberá organizarse de la siguiente forma:

- El programa principal supervisa la señal **clr** y lleva a cero la cuenta cuando corresponda.
- La señal **tic** deberá interrumpir al sistema. En la rutina de atención a interrupción se deberá incrementar la cuenta y actualizar el display. Se supondrá implementada una subrutina **CONVERT** que recibe un número de 16 bits en el par de registros BC y devuelve el mismo número codificado como 5 dígitos decimales codificados en 7 segmentos a partir de una dirección fija **RESULTADO**, el dígito más alto se encuentra en la dirección mayor.
- La rutina de atención a interrupción no enmascarable deberá almacenar el valor de la cuenta en la memoria no volátil como ya fue indicado.

No hay restricciones sobre la organización del espacio de entrada salida, pero el espacio de memoria debe organizarse como se muestra en el siguiente diagrama:



Se deberá diseñar completamente el sistema:
a) Todo el hardware que sea necesario.
b) Todo el software con excepción de la subrutina **convert** (inicialización, programa principal, rutinas de atención a interrupción y directivas de reserva de memoria que sean necesarias).

Nota: la memoria no volátil no debe usarse como espacio de Stack

Problema 2

Descripción general

Se deberá implementar un reloj digital basado en un microprocesador Z80. El reloj contará con los botones: **sel** y **pulso** para configurar la hora del reloj. Si se presiona el botón **pulso** sin presionar el botón **sel**, se incrementa la hora; si se presiona el botón **pulso** presionando el botón **sel** se incrementan los minutos.

Funcionamiento y requerimientos del sistema

Después del reset el reloj deberá comenzar a desplegar las 00:00hs y comenzará a correr el tiempo en el display. En este display se desplegará la hora en los 2 primeros dígitos y los minutos en los 2 restantes.

El botón **sel** genera una señal en nivel alto todo el tiempo que se mantiene apretado, mientras que el botón **pulso** genera un flanco de subida cada vez que es presionado.

El z80 trabajará en modo 1, recibiendo una interrupción periódica cada 60 segundos, y la rutina de atención a interrupción se encargará de actualizar la hora en una variable en memoria y los display 7 segmentos. En esta rutina, cuando se actualiza la hora debe chequearse que la hora este en el rango de 00 a 23 y los minutos de 00 a 59.

La hora y los minutos deberán guardarse en binario, en las variables:

- **v_hora**: 1 byte para la hora (00 a 23)
- **v_minuto**: 1 byte para minutos (00 a 59).

El programa principal se encargará de inicializar todo el sistema luego de un reset, y monitorear mediante polling los flancos de **pulso**. Como se describió anteriormente, en caso de generarse un flanco en **pulso** con **sel**=0, se deberá incrementar (cuidando de no superar el rango) la variable **v_hora** y desplegar en el display el cambio realizado. En caso de generarse un flanco en **pulso** con **sel**=1, se deberá proceder igual con **v_minuto**.

Se dispone de una rutina **dispdisplay** que recibe en los registros BC la hora a desplegar (B:hora, C:minutos) y en HL una dirección de memoria. Esta rutina devuelve en 4 bytes a partir de la dirección indicada en HL los 4 digitos a desplegar en el display ((HL): unidades de minutos, (HL+1): decenas de minutos, (HL+2): unidades de hora, (HL+3): decenas de hora).

Se equipará el sistema con un CTC que deberá generar interrupciones cada 1 minuto luego de su inicialización. El reloj de el sistema es de 4Mhz, por lo que probablemente se deban encadenar varios canales del CTC.

Se deberá implementar o indicar:

a) Todo el Hardware del sistema, para la memoria se dispone de una ROM de 32K y una RAM de 16K.
b) Todas las directivas de reserva de memoria al ensamblador y declaración de constantes.
c) Todo el software (excepto la rutina **dispdisplay**) :
    - Inicialización necesaria para un correcto funcionamiento luego de un reset (CTC, variables, etc.)
    - Rutina de atención a interrupciones
    - Programa principal.

Nota: 60 seg. @ 4MHz = 240e6 pulsos. 240e6 = 256 * 250 * 250 * 15