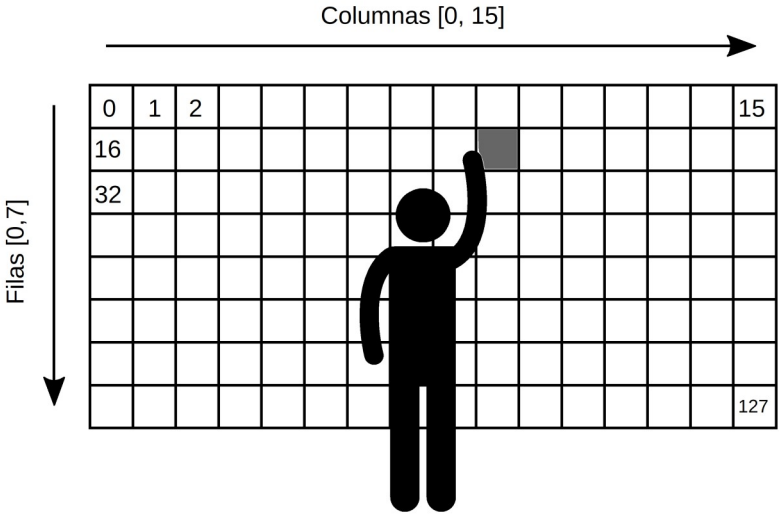
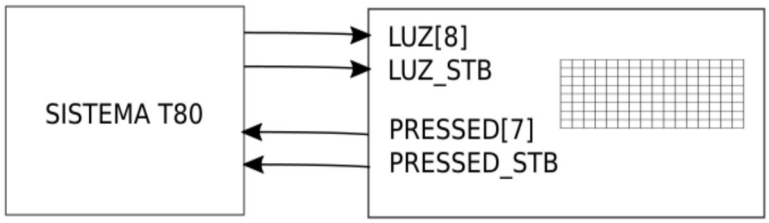
• Nombre y CI en cada hoja • Numere las hojas, indique el total en la primera • Utilice sólo un lado de las hojas	• Incluya un solo problema por hoja • **Sea prolijo** • **Aprobación:** mínimo UN problema

PROBLEMA 1

Para una instalación interactiva se busca diseñar un juego basado en una pared de luces organizada como **8 filas y 16 columnas**. La pared irá encendiendo una luz a la vez en una secuencia aleatoria y el objetivo del jugador es presionar dicha luz para apagarla antes de que transcurra un tiempo límite **T_TIMEOUT.** Mientras el jugador logre presionar la luz correcta antes del tiempo límite el juego continuará y se encenderá la próxima luz de la secuencia aleatoria. En caso que el jugador no llegue a presionar la última luz encendida a tiempo o presione una luz incorrecta, el juego terminará.



La pared de luces con la que se cuenta tiene una interfaz que permite controlar el estado de cada luz individual a través de las señales **LUZ[8]** y **LUZ_STB.** Al detectar un flanco de subida en su entrada **LUZ_STB**, la pared de luz cambiará el estado de la luz identificada por los 7 bits menos significativos presentes en la entrada **LUZ**. El estado de la luz seleccionada se tomará del bit más significativo de **LUZ** (0=OFF, 1=ON). Por otro lado, la pared también ofrece la función de reportar cual fue la última luz presionada a través de las señales **PRESSED[7] y PRESSED_STB.** Mediante un flanco de subida en **PRESSED_STB**, la pared indica que una luz ha sido presionada y la identifica con el valor en **PRESSED correspondiente.** (PRESSED solo es válido durante el flanco en **PRESSED_STB**). Luego de reset la pared comienza con todas las luces apagadas.



El sistema T80 a diseñar debe implementar el juego mencionado controlando la pared de luces a través de su interfaz antes descrita. Luego de un reset, el juego debe estar en un modo "demo" en el que la luz encendida va recorriendo la pared en orden (0, 1, 2 ... 127, 0, 1, 2, ... 127) y cambiando cada **T_TIMEOUT**. Una vez un jugador presione cualquiera de las luces, el juego comenzará y continuará hasta que el jugador pierda. Cuando el jugador pierde (fin del juego), se deberá iniciar una secuencia que encienda inmediatamente todas las luces, que deberán permanecer encendidas hasta el próximo reset del sistema. El sistema funcionará en **Modo 2** de interrupciones, con un **Timer** configurado para detectar el transcurso del tiempo **T_TIMEOUT**.

Se pide

a) Hardware del sistema, incluyendo puertos, periféricos y su decodificación. Se supone ya disponibles memorias ROM y RAM de 32KiB cada una.

b) Rutina de atención a interrupción

c) Rutina principal encargada de leer la última luz presionada y actuar en consecuencia.

d) Inicialización del sistema incluyendo directivas de memoria, inicialización de variables, etc.

Notas:

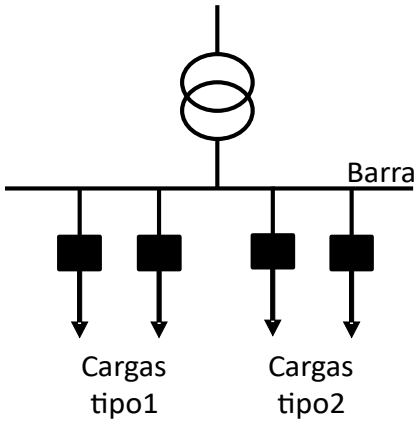
Se dispone de una rutina auxiliar **RAND** que al llamarla devuelve un byte aleatorio en el registro A.

T_TIMEOUT = 0.512s = 2^{13} * 250 * (1/4MHz) con F_{clk} = 4MHz

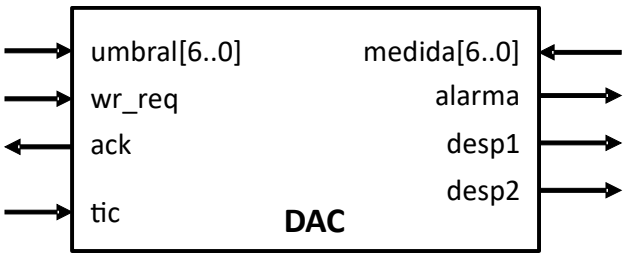
PROBLEMA 2

En una instalación eléctrica un transformador alimenta una barra a partir de la cual se alimentan cargas tipo 1 y cargas tipo 2. Para controlar la tensión en la barra y evitar que baje de determinado nivel se desea implementar un sistema de Despeje Automático de Carga (**DAC**) en un microprocesador **z80**.

La tensión de la barra será sensada periódicamente a través de la entrada **medida[6..0],** si se detectan **4 medidas consecutivas** por debajo del nivel de control se deben despejar las cargas tipo 1 generando un pulso a '1' por la salida **desp1** y se debe encender la salida **alarma** de forma persistente hasta un reincio del sistema. Luego de haber despejado las cargas tipo 1 si se vuelven a detectar **4 medidas consecutivas** por debajo del nivel de control se deben despejar las cargas tipo 2 generando un pulso a '1' por la salida **desp2**, habiendo despejado las cargas tipo 2 se deja de controlar el nivel de tensión hasta un reinicio del sistema. Se dispone de la señal periódica **tic** para ser utilizada como fuente de interrupción trabajando en **modo 1**. En la rutina de atención a la interrupción se debe resolver el control de nivel de tensión y el manejo de las salidas **alarma**, **desp1** y **desp2**.

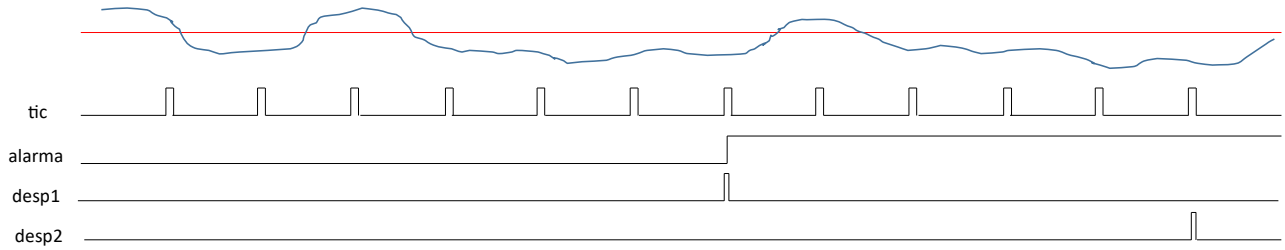


El nivel de control se debe inicializar en **0x40** y puede ser modificado externamente por medio de la entrada **umbral[6..0],** cuando se reciba un pulso corto a '1' por la entrada **wr_req**, se debe modificar el nivel de control con el dato disponible en **umbral[6..0],** el dato se mantendrá en la entrada hasta que se emita un pulso corto a '1' por la salida **ack**.



La actualización del nivel de control se debe resolver en un loop infinito dentro del programa principal intercalándolo con el llamado a la función **otras_tareas()** que atiende otras funcionalidades del sistema y se debe considerar como ya implementada. Si llega un nuevo nivel de control cuando ya se han detectado algunas lecturas por debajo del nivel anterior, debe continuarse la cuenta utilizando el nuevo nivel.

El sistema se completa con memorias **RAM** y **ROM** de **32kB,** las direcciones de los puertos de entrada y salida deben estar comprendidas en el rango **0x80 – 0xBF**.



Se pide:

a) Hardware completo del sistema.
b) Rutina de atención a las interrupciones.
c) Inicialización, programa principal y reservas de memoria.