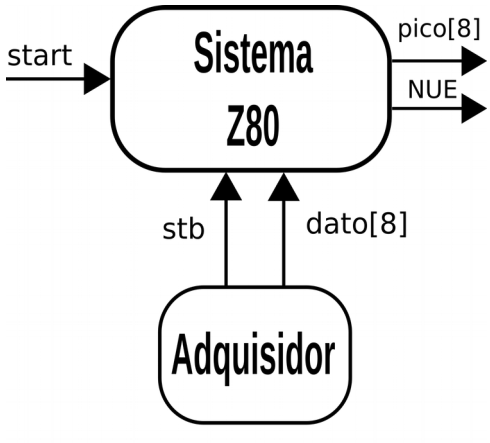
• Nombre y CI en cada hoja • Numere las hojas, indique el total en la primera • Utilice sólo un lado de las hojas	• Incluya un solo problema por hoja • **Sea prolijo** • **Aprobación:** mínimo UN problema

PROBLEMA 1

Se desea diseñar un sistema basado en Z80 que permita detectar picos en una señal de electrocardiograma (ECG). Para ello se cuenta con un sistema como el mostrado en la figura.



El adquisidor obtiene muestras de la señal de ECG a una frecuencia fija. Al obtener una nueva de ellas, coloca el valor en la salida **dato[8]** y genera un pulso a 1 de corta duración en su salida **stb**.

La señal **stb** deberá interrumpir al sistema, el cual se encontrará funcionando en modo 1, y la rutina de atención a la interrupción deberá guardar el valor de la muestra en memoria. Para ello se debe implementar una cola circular de 256 lugares, desde la dirección 0x8000 a 0x80FF. Esto quiere decir que, luego de un reset, las muestras se deberán comenzar a guardar en la dirección 0x8000 y cuando se llegue a 0x80FF se continuará nuevamente con la dirección 0x8000.

En la siguiente figura se muestra una parte de un ECG y un zoom en uno de los picos a detectar.

Para detectar el pico, se debe de analizar de a 3 muestras. Se encuentra un pico en la muestra 2, si el valor de la muestra es mayor al umbral (preestablecido en 0x80), y se cumple que la muestra 1 y la muestra 3 son menores a la muestra 2.

Cuando se detecta un flanco de subida en la señal **start**, el programa principal debe procesar las últimas 200 muestras guardadas en memoria y buscar los picos que allí se encuentren. Cada vez que se detecta un pico, se debe escribir en la salida **pico[8]** el valor de dicha muestra y se debe generar un pulso en alto de corta duración en la salida **NUE**. En caso de no encontrar ninguno, se debe de escribir 0x00 en **pico** y dar el pulso en **NUE**. Se asegura que al momento de generarse el **start**, habrá al menos 200 muestras en la cola.

Al ser una cola circular, debe de tenerse especial cuidado al incrementar o decrementrar punteros cuando se encuentran cerca de los extremos de la misma.

Se pide:

a) Todo el hardware incluyendo memorias 32kB de ROM y 32kB de RAM

b) Rutina de atención a la interrupción.

c) Inicialización del sistema que incluya todo lo necesario para el funcionamiento correcto del sistema. Incluir las directivas necesarias para determinar la ubicación en memoria de todo el código, variables, tablas, cola circular y constantes usadas.

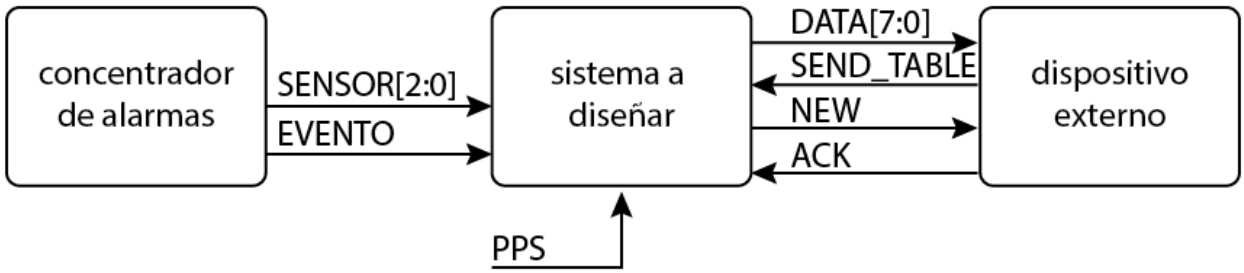
d) Programa principal

PROBLEMA 2

Se desea implementar un sistema basado en un **T80** que registre el instante en que se producen alarmas reportadas por un concentrador.

El concentrador de alarmas genera un pulso a cero en su salida **EVENTO** cuando uno de sus 8 sensores se activa y al mismo tiempo despliega el número del sensor involucrado (de 000b a 111b) en la señal **SENSOR[2..0]**. El valor en **SENSOR**[] solo es válido durante el flanco de bajada de **EVENTO**. Los eventos se producen con una distancia mayor a 1 segundo.

Se debe implementar un contador de 16 bits para llevar la cuenta de los segundos transcurridos desde el reset, esta cuenta será la que se utilizará para registrar el instante de la alarma. Se cuenta con una señal externa **PPS**, la cual consiste en un pulso a nivel alto cada 1 segundo.



El sistema a diseñar deberá ser interrumpido por la señal **EVENTO**, leer cuál fue el sensor involucrado y registrar la hora en una tabla con la estructura:

Dirección memoria	Contenido
0x9000	Alarma sensor 0, Hora byte bajo
0x9001	Alarma sensor 0, Hora byte alto
0x9002	Alarma sensor 1, Hora byte bajo
0x9003	Alarma sensor 1, Hora byte alto
0x9004	Alarma sensor 2, Hora byte bajo
0x9005	Alarma sensor 2, Hora byte alto
0x9006	Alarma sensor 3, Hora byte bajo
0x9007	Alarma sensor 3, Hora byte alto
…..	…..

Una vez inicializado todo el sistema, el programa principal deberá quedarse en loop esperando que la señal **SEND_TABLE** pase a 1. Cuando esto suceda, se enviarán hacia un dispositivo externo uno a uno los bytes por el puerto **DATA[7:0]** hasta terminar el contenido, luego de lo cual volverá a esperar por la señal **SEND_TABLE.** Esta señal pasa a 0 inmediatamente que el dispositivo externo recibe el primer byte.

Para controlar el flujo de datos se cuenta con las señales **NEW** y **ACK.** Cuando el dispositivo externo esta listo para recibir un nuevo byte, pone a 1 su señal **ACK**. La presencia de un nuevo dato debe ser indicada con un flanco de subida en **NEW**, en ese instante el dispositivo bajará **ACK** inmediatamente.

Se pide:

a) Hardware de puertos de entrada y salida y controladores de interrupción. No se debe hacer las memorias (dispone de 32K de ROM y 32K de RAM). Existen otros puertos ya implementados y solo se dispone de las direcciones 0x80 a 0xFF.
b) Reserva de variables en memoria y software a ejecutar luego de un reset para inicializar el sistema, el cual terminará en JP MAIN_LOOP, dirección donde se implementará el bucle del programa principal. Toda la tabla debe de comenzar inicializada en 0.
c) Loop de programa principal que envía la tabla.
d) Rutinas de atención a las interrupciones generadas por las señales PPS y evento.