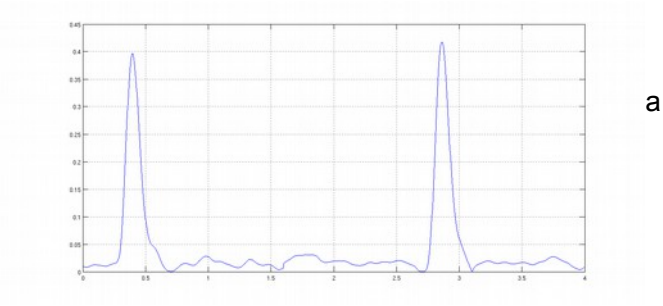
- Nombre y CI en cada hoja - Numere las hojas - Indique el total de hojas en la primera - Utilice solo un lado de las hojas	- Incluya un solo problema por hoja - **Sea prolijo** - **Aprobación:** mínimo UN problema

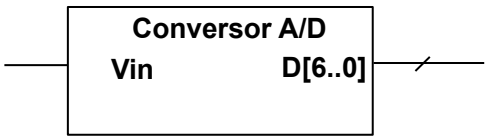
PROBLEMA 1

Se desea diseñar un sistema basado en un microprocesador z80 (fclk = 2MHz) que detecte picos en una señal como la de la figura. Cada vez que sucede un pico el sistema debe dar un pulso uno en un puerto de salida de 1 bit **(bandera)** mientras dure el pico y cuando la señal vuelve a cero debe dar en un puerto de salida de 8 bits **(amplitud)** el valor máximo recibido durante el pico.



a

La señal se adquiere a través de un conversor A/D de 7 bits como el de la figura.



Las lecturas del conversor se deben realizar cada 2ms. Para eso se cuenta con una señal **tic** periódica de de 500Hz.

En la rutina de atención a interrupciones se debe adquirir los datos a través del conversor A/D, procesar los picos y dar las salidas correspondientes en los puertos **bandera** y **amplitud**. Inicialmente se considera la señal en reposo, el comienzo del pico se detecta cuando la diferencia entre el dato actual y el dato previo es mayor al valor almacenado en la variable **umbral_1**. A partir de que se cumple esta condición la salida **bandera** se sube a '1' y se comienza a calcular el valor máximo de la señal. Se considera que la señal vuelve al reposo cuando el dato actual y el dato previo son ambos menores al valor almacenado en la variable **umbral_2**. Cuando se cumple esta segunda condición la salida **bandera** se debe volver a '0' y se debe dar el valor máximo del pico por la salida **amplitud**. A partir de este momento se vuelve a esperar un nuevo inicio de pico.

Luego de la inicialización el programa principal debe quedar en loop verificando si las variables **umbral_1** y **umbral_2** se desean modificar externamente. Cuando se quiere modificar las variables externamente se colocan los datos en las señales de 7 bits **dat_umbral_1** y **dat_umbral_2** respectivamente y se da un flanco de subida en la señal de 1 bit **req_umbral**. Luego de actualizadas las variables el sistema debe dar una confirmación con un pulso a '0' a través de la señal de 1 bit **ack_umbral**. Los datos en las señales **dat_umbral_1** y **dat_umbral_2** se mantienen estables entre el flanco de subida en **req_umbral** y el pulso en **ack_umbral**.

Se pide:

a) Hardware del sistema. Se cuenta con un chip de 32KB de RAM y uno de 32KB de ROM. Se debe reservar para futuras aplicaciones las tres cuartas partes más bajas (00h-BFh) del espacio de E/S.

b) Inicialización del sistema y reserva de memoria para las variables. Luego de un reset el sistema debe inicializar las variables umbral_1 y umbral_2 en 20 y 5 respectivamente.

c) Loop del programa principal para modificar externamente las variables **umbral_1** y **umbral_2**.

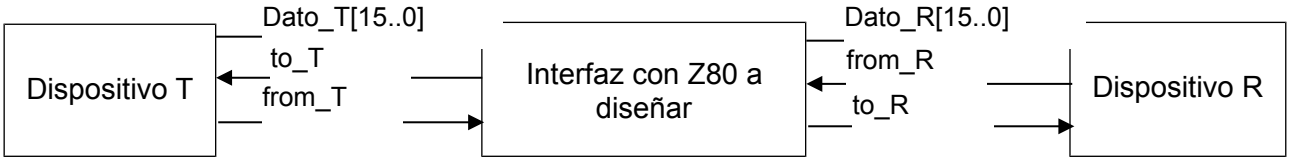
d) Rutina de atención a interrupciones que realice la detección de picos y maneje las salidas bandera y amplitud.

PROBLEMA 2

Se desea conectar un dispositivo transmisor T, con un dispositivo receptor R. El dispositivo **T** transmite datos de 16 bits de acuerdo al siguiente protocolo: cuando T tiene un dato para transmitir, si **to_T** es 1, pone los datos en la salida **Dato_T**[15..0] y da un pulso a 0 en **from_T**. Si **to_T** es 0, espera a que vaya a 1 para poner los datos y dar el pulso en **from_T**. La señal **to_T** debe bajar a 0 como consecuencia del pulso en **from_T** y volver a 1 cuando la interfaz a diseñar lee el dato. Los datos en **Dato_T** están válidos solamente en el flanco de subida de **from_T**.

El dispositivo **R** recibe datos de 16 bits, de acuerdo al siguiente protocolo: cuando se quiere enviar hacia R se debe poner el dato en **Dato_R**[15..0] y un 1 en en la señal **to_R**, lo que provoca que R lea el dato y comience a procesarlo. Una vez que lo ha procesado da un pulso a 0 en la señal **from_R** para indicar que queda pronto para un nuevo dato. Este pulso debe provocar que **to_R** vuelva a 0.

Los datos son transmitidos en ráfagas, es decir, **T** transmite muchos datos consecutivos durante un tiempo, luego queda en reposo por un tiempo antes de volver a transmitir, y así sucesivamente. El dispositivo **R** es capaz de procesar, **en promedio**, más datos por segundo de los que envía el dispositivo **T**, pero no soporta las ráfagas. Se desea diseñar una interfaz basada en el procesador Z80 que permita interconectar los dispositivos T y R.



Los datos se almacenan temporalmente mediante una cola FIFO implementada en software. Para poner un dato en la cola se deberá invocar a la subrutina **PutFifo** que recibe como parámetro en el par de registros BC el dato a escribir. Para obtener el dato que está primero en la cola se deberá invocar a la subrutina **GetFifo.** Si la cola no está vacía, **GetFifo** saca el primer dato de la cola y lo devuelve en el par de registros BC con el flag Z prendido. Si en cambio no hay datos para leer en la cola la subrutina devuelve el flag Z apagado y el contenido del acumulador es desconocido. Se podrá suponer que la cola nunca se llena.

El sistema debe funcionar en Modo 2 de interrupciones, con rutinas de interrupción para ambos puertos que responderán a los pulsos en **from_R** y **from_T**. Se debe utilizar un único periférico PIO para implementar el byte bajo de **Dato_T** y **Dato_R** y los protocolos descritos arriba para las señales from_T, to_T, from_R y to_R. Se agregarán los puertos y lógica adicional que sea necesario para implementar el byte alto de dichas señales. La entrada debe tener prioridad frente a la salida.

Se debe implementar una variable de memoria **R_BUSY** para comunicación entre las 2 rutinas de atención a interrupción, que debe valer FFh si el dispositivo R está procesando un dato y 00h en caso contrario.

Se pide:

a) Diagrama del hardware. Se supondrá la memoria ya implementada (32K ROM y 32K RAM). Se debe mapear los puertos dentro del rango 80h-CFh.

b) Reserva de memoria e inicialización del sistema, que finaliza con las instrucciones del recuadro.

```
   call iniotros
Loop:
   call otros
   jp Loop
```

c) Rutina de atención a la interrupción del Dispositivo R. Si la memoria FIFO no está vacía, se debe enviar al dispositivo R el primer dato de la cola. En caso contrario, la rutina no debe enviar ningún dato y debe indicarlo haciendo **R_BUSY** = 00h.

d) Rutina de atención a la interrupción del Dispositivo T. Se debe leer el dato proveniente del Dispositivo T y si R está ocupado procesando datos (**R_BUSY** = FFh) se debe guardar el dato en la cola FIFO. En caso contrario se debe enviar directamente al dispositivo R y modificar la variable R_BUSY.

Nota: los pulsos en from_T y from_R tienen una duración de al menos un período de CK.