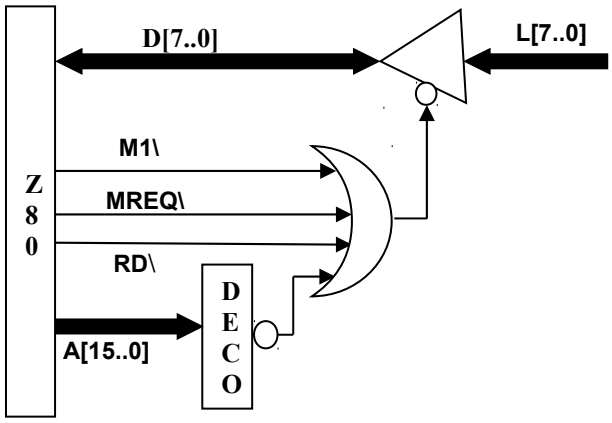
a) Utilice solo un lado de las hojas
b) Incluya un solo problema por hoja.
c) Sea prolijo.
d) Indique Nombre, CI y numere cada hoja.

Ejercicio 1 (18 pts.)

Un sistema con Z80 utiliza la lógica de la figura para realizar pruebas de laboratorio. Cuando el Z80 intenta leer un código de operación en la dirección de memoria indicada por el DECO, leerá el dato en la entrada L[7..0]. La entrada L[7..0] solo se modifica cuando el sistema no está energizado.



Se pide:

a) En función de los parámetros de tiempos del decodificador, la compuerta OR, el buffer triestado y el microprocesador Z80 escribir todas las condiciones necesarias para que solo se requiera insertar un UNICO tiempo de espera en el ciclo M1, para respetar el tiempo de setup de datos del Z80. Indicar para cada parámetro, cuando corresponda, si debe considerarse el valor máximo o el mínimo. Los períodos o semiperíodos de reloj T que aparezcan en las ecuaciones se deben anotar indicando a qué período corresponden dentro del ciclo (p. ej. T1, $T2_{LOW}$, etc.).
b) Determinar qué requerimientos se debe imponer al buffer para que el sistema funcione correctamente, si se está utilizando un Z84C0020 con un reloj de ciclo de trabajo 50% ($T_{HIGH}=T_{LOW}$) y frecuencia fclk= 20MHz.

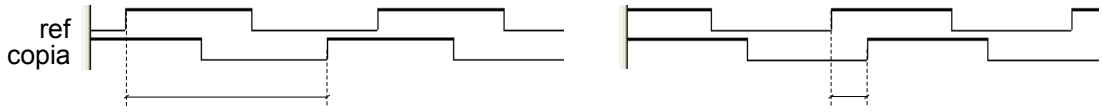
Datos:

Retardo Compuerta:	$t_{or(min)}$=0ns ; $t_{or(max)}$=10ns
Retardo Decodificador:	$t_{deco(min)}$ = 0ns; $t_{deco(max)}$ = 30ns
Retardo Buffer entrada a salida:	$t_{es\_bff(min)}$; $t_{es\_buff(max)}$ (datos no disponibles)
Retardo Buffer habilitación a salida:	$t_{oe\_bff(min)}$; $t_{oe\_buff(max)}$ (datos no disponibles)

Ejercicio 2 (18 pts.)

En un sistema con Z80 existente se desea agregar un mecanismo para determinar el desfasaje entre dos señales periódicas, **ref** y **copia**. Ambas señales son ondas cuadradas cuyas frecuencias presentan pequeñas variaciones alrededor de una frecuencia nominal fnom = 100 Hz. Para ello en cada flanco de subida de **copia** se debe generar una interrupción en la que se determinarán dos cosas:

- El retardo desde el flanco de subida de **ref** hasta el de **copia** que provocó la interrupción, medido en múltiplos de 100 us (se garantiza que ese retardo es siempre menor que 100 us x 255)
- El valor de **ref** (0 o 1) memorizado en el instante en que se produjo el flanco de subida de **copia.**



Se trabajará en modo 2 de interrupciones. Hay otros dispositivos que solicitan interrupción al Z80 pero la interrupción por flanco de subida de **copia** debe ser la de mayor prioridad. Se sugiere utilizar un canal de un CTC para medir el retardo y otro canal para generar las interrupciones con cada flanco de **copia**.

Se pide:

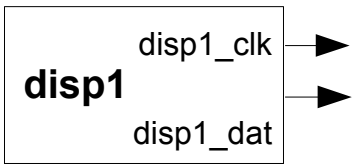
a) Conexión hardware del CTC y los puertos que sea necesario agregar, incluyendo su decodificación. Está disponible para esto el rango de direcciones 0x40 - 0x7F de E/S.
b) Inicialización del CTC y del sistema de interrupciones del Z80. Se debe asignar a los canales del CTC los vectores de interrupción 4 al 7 (del quinto al octavo lugar en la tabla de interrupciones).
c) Rutina de atención a la interrupción que determina el valor del retardo y el valor de ref, los almacena en los lugares de memoria reservados RETARDO y VALOR_REF y deja todo pronto para el siguiente período. Despreciar el tiempo de latencia de la interrupción respecto al retardo medido.

NOTA: fclk = 2,560 MHz frecuencia de reloj del sistema (Notar que 100 useg = 2^8 . 1 / fclk)

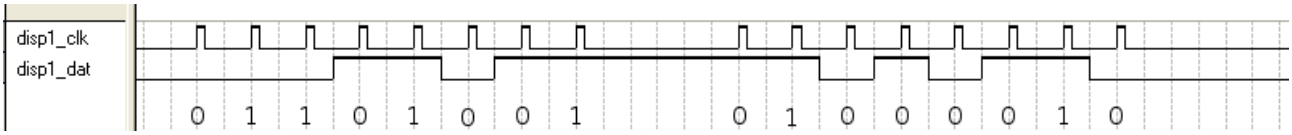
Ejercicio 3 (64 pts.)

Se desea utilizar un microprocesador ***Z80*** para conectar dos dispositivos, uno que transmite datos en formato serie (***disp1***) y otro que recibe en formato paralelo (***disp2***).

disp1 transmite palabras de 8 bits en formato serie a través de dos señales (***disp1_clk*** y ***disp1_dat***). Cada vez que el dispositivo da un flanco de subida en la señal ***disp1_clk*** significa que en la señal ***disp1_dat*** hay un bit para ser leído.

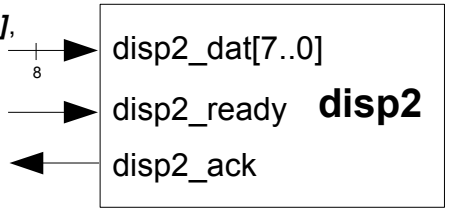


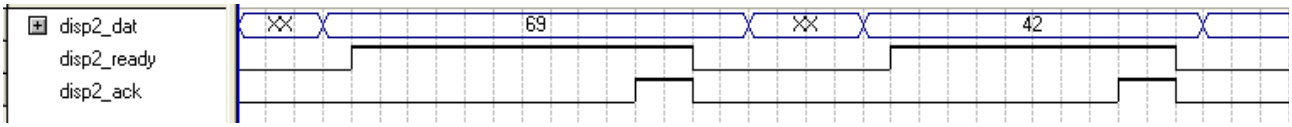
Los datos transmitidos por el dispositivo 1 son codificados de la siguiente manera: si el bit recibido es igual al anterior debe leerse como un 1 y si es diferente debe leerse como un 0. El bit más significativo de todas las palabras se asume 0 independientemente del valor del bit recibido y es el primero en ser transmitido, ver figura.



Para recibir los datos de ***disp1*** cada flanco de subida en ***disp1_clk*** generará una interrupción. La rutina de atención a interrupciones será la encargada de leer el bit recibido y decodificarlo. Además, cuando se complete una palabra decodificada de 8 bits la interrupción deberá guardarla en memoria en la dirección reservada ***palabra*** y avisar al programa principal seteando el valor **0xFF** en otra variable de memoria llamada ***hay_palabra***.

disp2 recibe los datos en formato paralelo a través de ***disp2_dat[7..0]***, cuando la señal ***disp2_ready*** toma valor '1' el dispositivo lee la palabra de ***disp2_dat[7..0]*** y confirma la lectura del dato llevando la señal ***disp2_ack*** a '1', esta señal mantendrá este valor hasta que ***disp2_ready*** vuelva a '0'.
Desde que se activa la señal ***disp2_ready*** hasta que se recibe la confirmación del dispositivo, las señales ***disp_ready*** y ***disp_dat[7..0]*** deben mantenerse estables, ver figura.





El programa principal mostrado en el recuadro es un loop infinito que invoca a la subrutina atiendo_otros, que realiza tareas no descritas aquí, y si hay una palabra la transmite invocando a la subrutina ***put_palabra***.
La subrutina ***put_palabra*** debe colocar la palabra en ***disp2_dat[7..0]*** y llevar la señal ***disp2_ready*** a '1' para indicarle a ***disp2*** que hay un dato pronto, cuando ***disp2*** responda con ***disp2_ack*** debe llevar ***disp2_ready*** a '0' y retornar.

Se puede suponer que cada vez que se completa una palabra recibida la anterior ya fue transmitida por el programa principal, y que habrá tiempo suficiente para inicializar el sistema antes que ***disp1*** comience a transmitir datos.

```
loop_ppal:
  call atiendo_otros

  ld a, (hay_palabra)
  cp 0
  jr z, loop_ppal

  ld a, (palabra)
  call put_palabra
  ld a, 0
  ld (hay_palabra), a
  jr loop_ppal
```

Se pide:

a) Hardware completo del sistema ***Z80***, ***disp1***, ***disp2***, incluyendo memorias para lo cual se cuenta con un chip de 32K de ROM y un chip de 32K de RAM.
b) Inicialización del ***Z80*** y reserva de variables de memoria.
c) Rutina de atención a interrupciones.
d) Subrutina ***put_palabra***.