Informe Puerto Paralelo

Diseño

Nosotros pensamos resolver el problema de esta manera:
La idea general es mostrar un char* de cualquier tamaño en el display, como el display posee sólo 7 segmentos, debemos partir ese char* en pedazos de tamaño 7, y combinar esos pedazos de manera que el char* pueda ser leído a través del display. Para ello desarrollaremos dos TADs:
Un TAD string, que se ocupara de manejar el char* a imprimir por el display, es decir, de partir ese char* adecuadamente, y otro TAD timer, que regulara la velocidad con el que ese char* es actualizado según el TAD string.
Siguiendo con esta línea de pensamiento, desarrollaremos una rutina de interrupción del puerto paralelo, la cual se encargará de tomar un caracter del char* obtenido a partir del TAD string, seleccionar un display y imprimir ese caracter en el dispay seleccionado, y en base al TAD timer, decidir si actualizamos el "pedazo" de char* a través del TAD string o dejarlo como está.

Corriendo el programa

Lo primero que hacemos es asegurarnos que el puerto paralelo este en dirección de salida (forward).Dado que vamos a cambiar la rutina de interrupción, primero salvamos la actual, y luego activamos la nuestra. Después desenmascaramos el pic ,para que las interrupciones no puedan ser ignoradas y activamos la petición de interrupciones del puerto paralelo para que cada interrupción ejecute nuestra rutina, pues de lo contrario las interrupciones del puerto paralelo serán ignoradas.
Ahora le pedimos al usuario la cadena de caracteres que desea imprimir en el display (como máximo 20 caracteres), con esta creamos nuestro TAD strin e iniciamos y creamos nuestro TAD timer, quien será manejado por nuestra rutina de interrupción.
A partir de aquí se empieza a mostrar por el display la cadena que el usuario ingreso, gracias a las interrupciones generadas por el display (y aquí es a donde participa nuestra rutina), y esto continúa hasta que el usuario presione una tecla.
Cuando esto ocurre desactivamos la petición de interrupción del puerto paralelo (por esto el display dejará de mostrar el mensaje tal como lo estaba haciendo), volvemos enmascarar el pic, restauramos el vector de interrupciones anteriormente salvado, ponemos el registro DATA en null para pisar lo que haya quedado y finalmente liberamos los recursos de memoria pedidos por nuestros TADs.

¿Que hace nuestra rutina de interrupción?

Lo primero que hace nuestra interrupción, es desactivar las interrupciones (con el objeto de que queden desactivadas durante la corrida de nuestra rutina). Luego tomamos un caracter de un char* que es un pedazo (de tamaño 7) de la cadena ingresada por el usuario, el cual va a ser impreso por un display, para esto obtenemos su código ASCII, y traducimos ese código según la tablita que nos dio la cátedra. A este carácter traducido lo copiamos en el registro DATA de nuestro puerto paralelo para que sea leído a través del puerto. Después, seleccionamos el display por el que el byte será impreso, para esto, activamos el registro CONTROL del display indicado. Finalmente, debemos actualizar la información a mostrar por el display, y es aquí donde interviene nuestro timer. Cada vez que nuestro timer llega al timeout, cambiamos el mensaje a mostrar por el display de la siguiente manera: si "c" es la cadena que se imprime en el display, entonces c[i] = c[i+1] para 0 <= i

<= 6, y c[7] = "el siguiente caracter a c[6] en la cadena ingresada por el usuario (la cual la tratamos como circular)" , y reiniciamos la cuenta del timer. Finalmente, avisamos al PIC que nuestra rutina de interrupción ha terminado, y activamos las interrupciones (dado que al inicio de la rutina la habíamos desactivado).

Problemas:

El problema que posee nuestro programa es que hasta el primer timeout el display imprime cualquier cosa, y luego comienza a andar bien salvo por una cosa: el último display no imprime el caracter que debería, sino que a veces no imprime nada, otras veces imprime cualquier cosa.
La posible solución al problema de que hasta el primer timeout imprime cualquier cosa es obtener la cadena que el usuario desea imprimir antes de activar nuestra rutina de interrupciones, pues de lo contrario, nuestra rutina lee basura de la memoria, y por consiguiente el display imprime basura, recordemos que a partir de cualquier timeout, se actualiza la información que mandamos a los display.