Prácticas
Sistemas Empotrados I
Curso 2013-2014

Dr. D. José Luis Villarroel Salcedo

Dr. D. Enrique Torres Moreno

ÍNDICE

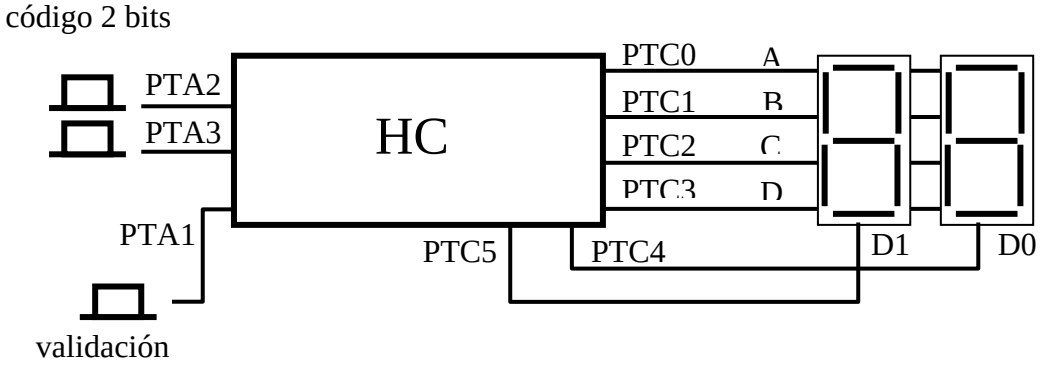
Practica 1. Entradas salidas paralelo. Entorno de desarrollo

1 Objetivos

- Manejo del entorno de programación
- Programación E/S paralelo
- Manejo depurador

2 Descripción del sistema controlado

El sistema controlado se compone de 3 pulsadores de entrada conectados al PTA y de dos *displays* de 7 segmentos conectados al PTC:



Al presionar el pulsador de validación, el valor codificado por los 2 pulsadores se muestra por el *display* D0. El valor contenido en el *display* D0 pasa al D1. Inicialmente, D0 contiene el valor 0 y D1 el valor 1.

3 Trabajo a realizar

3.1 Conexión de los periféricos al microcontrolador

- Verificar y entender las conexiones realizadas en la Placa de Adaptación. PTA debe estar conectado a los pulsadores y PTC a los visualizadores de siete segmentos.
- Verificar el correcto funcionamiento de los pulsadores que se vayan a utilizar.

3.2 Programación del la iluminación de los displays

- Implementación de una función para la iluminación del display, cuyo prototipo es:

```
void display(unsigned char queDisplay, unsigned char valor)
```

- Implementación de la iluminación alternativa de los 2 *displays* con los valores 0 y 1 respectivamente

3.3 Lectura del valor introducido a través de los pulsadores

Es importante resaltar que los pulsadores conectados al puerto PTA están negados, por lo tanto: pulsado → valor 0; sin pulsar → valor 1. De esta forma al presionar un pulsador se genera un flanco de subida.

- Configurar el PTA:
  - PTA1..PTA3 como entradas con *pullup*
  - Habilitación de la interrupción en PTA1 como flanco de bajada
- Programar una rutina de interrupción que lea el valor de los pulsadores (recuérdese que están negados) y lo deposite en una variable global. La rutina de interrupción deberá ejecutarse cuando se produzca en un flanco de bajada en PTA1.
- Verificación del funcionamiento con el *debugger*

3.4 Aplicación completa

A partir de lo programado en los puntos anteriores montar la aplicación completa de acuerdo a las especificaciones.

3.5 Mapa de memoria

A partir de la información contenida en el *.map dibujar el mapa de memoria de vuestro programa, identificando las distintas secciones del programa, su ubicación en la memoria y el empleo de memoria, tanto FLASH como RAM

3.6 Puntos opcionales

Almacenar el valor de los 5 últimos códigos introducidos en el sistema y visualizar únicamente los más antiguos.

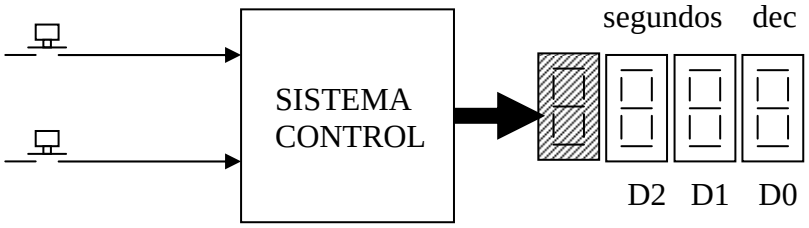
Practica 2. Manejo del tiempo. Un cronómetro

1 Objetivos

- Manejo de visualizadores de 7 segmentos
- Problemática del tiempo
- Programación de un módulo reloj de sistema basado en el TBM
- Programación de un cronómetro basado en el módulo reloj

2 Descripción del sistema

Se trata fundamentalmente del desarrollo de un cronómetro. Se deberá disponer de dos pulsadores: uno de RESET o puesta a cero del cronómetro y otro START/STOP de puesta en marcha o parada de la medición. El sistema deberá representar el tiempo en tres visualizadores de 7 segmentos. Los de la izquierda representarán los segundos y el de la derecha las décimas de segundo.



Para la realización del sistema se debe emplear la Placa de Adaptación, la tarjeta de Visualizadores de 7 Segmentos y la tarjeta de Pulsadores-Leds. A continuación se incluye una tabla con las conexiones:

HC08	**Señal tarjeta**	**Observaciones**
PTC0	Bit A código BCD	Entrada a conversor BCD – 7 segmentos
PTC1	Bit B código BCD	Entrada a conversor BCD – 7 segmentos
PTC2	Bit C código BCD	Entrada a conversor BCD – 7 segmentos
PTC3	Bit D código BCD	Entrada a conversor BCD – 7 segmentos
PTC4	Activación D0	Lógica negada
PTC5	Activación D1	Lógica negada
PTC6	Activación D2	Lógica negada
PTA1	P1	Pulsado → señal baja
PTA2	P2	Pulsado → señal baja

3 Trabajo a realizar

3.1 Trabajo previo de preparación

- Estudiar el funcionamiento e inicialización de los puertos PTA y PTC.
- Estudiar el funcionamiento e inicialización del módulo MTIM.
- Estudiar el código de base de la práctica.

3.2 Conexión de los periféricos al microcontrolador

- Verificar y entender las conexiones realizadas en la Placa de Adaptación. PTA debe estar conectado a los pulsadores, el pulsador de STAR/STOP al PTA1 y el pulsador de RESET al PTA2 (módulo KBI). PTC debe estar conectado a los visualizadores de siete segmentos.
- Verificar el correcto funcionamiento de los pulsadores que se vayan a utilizar.

3.3 Programación de un módulo de reloj

- Programar un módulo de reloj basado en el MTIM con un *tick* de 1 ms.
- ¿Cuál es el rango?
- Puesta a punto del módulo utilizando los *displays* (utilizar la función que se da ya programada) y visualizando el tiempo en segundos y décimas de segundo.
- Conseguir que todos los visualizadores luzcan por igual.

3.4 Pulsadores de control y aplicación

Programar la aplicación CRONOMETRO utilizando la capacidad del módulo de teclado KBI para interrumpir.

4 Estructura del módulo reloj

clock.h

```
#ifndef clock_h
#define clock_h

void Reset_Clock (void) ;
void Start_Clock (void) ;
void Stop_Clock (void) ;
unsigned int Get_Time (void) ;
void delay_until(unsigned int T);

#endif
```

clock.c

```
#include "derivative.h"
#include "clock.h"


/* Variables del modulo */

volatile static unsigned int tick_counter ;


/* Implementacion servicios */
void interrupt 6 Tick (void) {…}

void Reset_Clock (void) {

  asm  sei ;

  /* Inicializacion del periferico */
  /* Tick de 1 ms, 1 kHz */
  …

  /* Inicializacion de variables */
  …

  asm cli ;
  return;
}

void Start_Clock (void) {…}

void Stop_Clock (void) {…}

unsigned int Get_Time (void) {…}

void delay_until(unsigned int T){
  while(!(T==tick_counter))
    asm WAIT ;
}
```