

## SISTEMAS EMPOTRADOS

### 3º Grado en Ingeniería Informática

## PRÁCTICA 2

### Manejo de los puertos básicos de Entrada/Salida y Timers (I)

#### 2.1. Introducción

El CI LPC2378 es un microcontrolador y por tanto multiplexa los pines como entradas y salidas. Cada pin maneja de una a cuatro funciones. Los pines de A0 a A15 y D0 a D8 son exclusivos para uso del controlador externo de memoria (EMC).

A continuación se muestran los registros de configuración de los pines de I/O:

- (F)IOxDIR ( $x = [0,4]$ ), selecciona en cada bit si un pin es entrada o salida, con 0 es una entrada y con 1 es una salida.
- (F)IOxSET ( $x = [0,4]$ ): escribe uno a los bits seleccionados de un registro IO.
- (F)IOxCLR ( $x = [0,4]$ ): escribe cero a los bits seleccionados de un registro IO
- (F)IOxPIN ( $x = [0,4]$ ): permite escribir o leer a un registro IO.
- IOxMASK ( $x = [0,4]$ ): cuando se escriben unos a los bits se inhibe cualquier acción de escritura en los registros de entrada salida.
- PINSELx ( $x = [0,10]$ ): estos registros escogen una de cuatro funciones que tiene cada pin del microcontrolador.

#### 2.2. Objetivos

Los objetivos marcados en esta práctica son los siguientes:

- Que el alumnado estudie y aprenda a configurar los puertos básicos de entrada/salida (GPIO) del LPC2378.
- Estudio y programación de los temporizadores (*timers*) de que dispone dicho microcontrolador.

- Diseño de un programa de aplicación y su comprobación real con el osciloscopio.

### 2.3. Material utilizado

El material necesario para la realización de esta práctica es el siguiente:

- Ordenador personal con Windows XP (mínimo) con el software Keil  $\mu$  Vision 5 instalado y el *pack* correspondiente a nuestra placa.
- Placa de desarrollo Keil MCB2300.
- Adaptador USB–JTAG de la familia ULINK para depurar programas.
- Dos cables USB A–B conectados a dos puertos USB disponibles del ordenador.
- Osciloscopio.

### 2.4. Desarrollo de la práctica

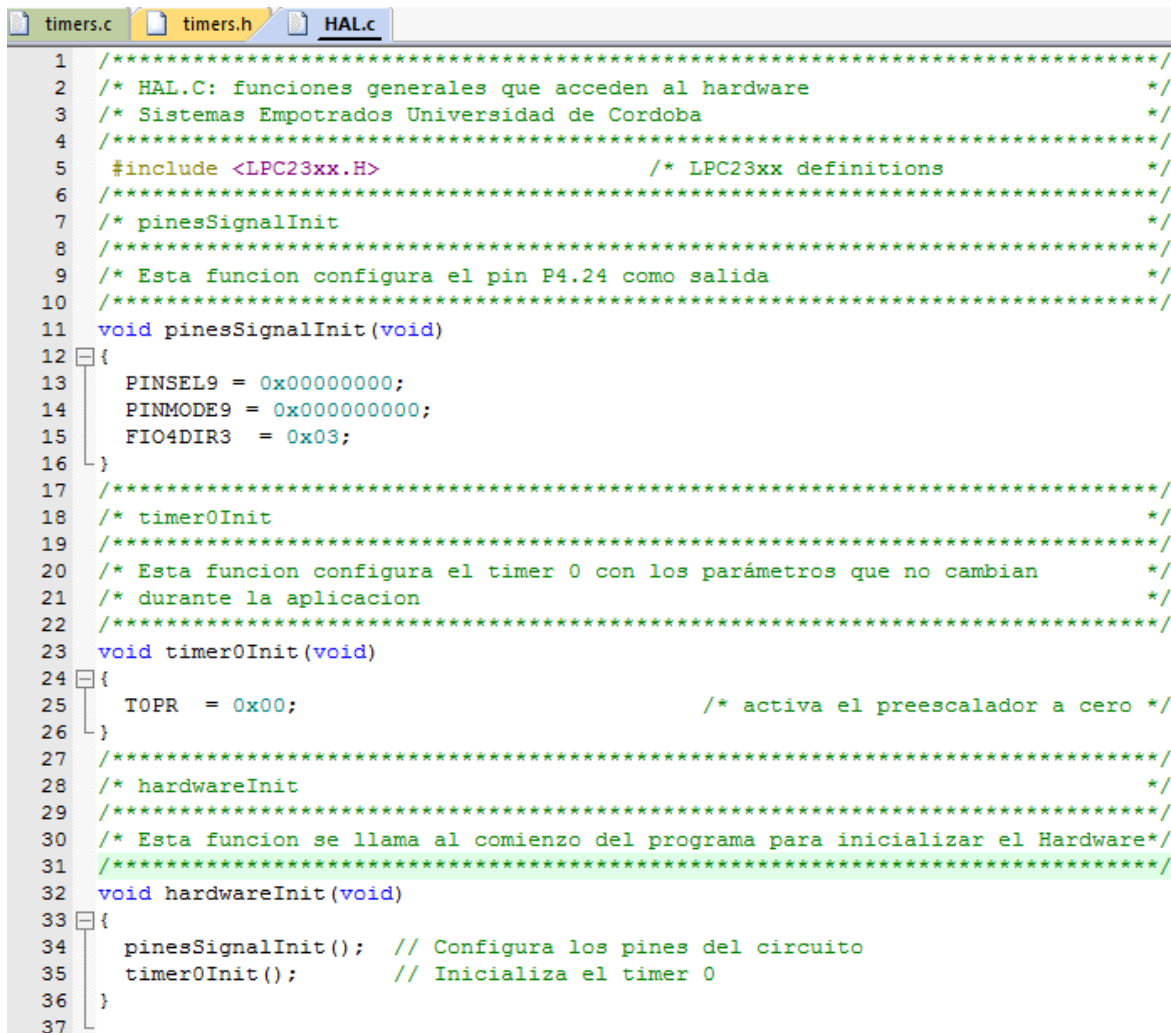
Queremos generar una señal digital por un puerto GPIO con distinta anchura a nivel bajo que a nivel alto. Tendrá una duración a baja de 500  $\mu$ s y a alta 700  $\mu$ s. Se comprobará con el osciloscopio que esta señal se genera correctamente.

Se programarán los timers por *polling* (sondeo). El final de la cuenta del *timer* se detectará muestreando los bits de control .

Crear un nuevo proyecto (practica2) en una nueva carpeta. Al igual que en la práctica anterior, copiar en esta carpeta los siguientes ficheros:

- LPC2300.s
- retarget.c
- serial.c
- HAL.c : para definir las funciones generales que acceden al hardware. En el caso de esta práctica al puerto de entrada/salida y al timer 0.)
- timers.h : definiciones necesarias para esta práctica y algunas funciones.

En el fichero HAL.c vamos a definir las funciones generales que acceden al hardware. En esta práctica solamente vamos a necesitar configurar un pin del puerto 4 (el pin P4.24) como salida, y el timer 0.



```

1  /*****
2  /* HAL.C: funciones generales que acceden al hardware
3  /* Sistemas Empotrados Universidad de Cordoba
4  /*****
5  #include <LPC23xx.H>          /* LPC23xx definitions
6  /*****
7  /* pinesSignalInit
8  /*****
9  /* Esta funcion configura el pin P4.24 como salida
10 /*****
11 void pinesSignalInit(void)
12 {
13     PINSEL9 = 0x00000000;
14     PINMODE9 = 0x00000000;
15     FIO4DIR3 = 0x03;
16 }
17 /*****
18 /* timer0Init
19 /*****
20 /* Esta funcion configura el timer 0 con los parámetros que no cambian
21 /* durante la aplicacion
22 /*****
23 void timer0Init(void)
24 {
25     TOPR = 0x00;          /* activa el preescalador a cero */
26 }
27 /*****
28 /* hardwareInit
29 /*****
30 /* Esta funcion se llama al comienzo del programa para inicializar el Hardware*/
31 /*****
32 void hardwareInit(void)
33 {
34     pinesSignalInit();    // Configura los pines del circuito
35     timer0Init();        // Inicializa el timer 0
36 }
37
  
```

**Figura 2-1 Fichero HAL.c para acceder al hardware**

PINSEL9 = 0X00000000      selecciona el pin P4.24 como GPIO (bits 17 y 16)

PINMODE9 = 0X00000000      lo habilita como salida resistencia pull-up

FIO4DIR3 = 0x03      selecciona los pines P4.24 y P4.25

TOPR = 0x00      activa el preescalador del timer 0 a un valor cero.

El fichero timers.h lo utilizaremos para incorporar las definiciones necesarias.

```

timers.c  timers.h  HAL.c
1  /*****
2  /* timers.h: definiciones para la practica de timers y algunas funciones */
3  /* Sistemas Empotrados Universidad de Cordoba */
4  /*****
5
6  #define PULSO0_LOW 5 //duracion del pulso a nivel bajo
7  #define PULSO0_HIGH 7 //duracion del pulso a nivel alto
8  #define SIGNAL0_PIN_HIGH FIO4SET3 = 0x01; // Pin señal 0 a alto P4.24
9  #define SIGNAL0_PIN_LOW FIO4CLR3 = 0x01; // Pin señal 0 a bajo P4.24
10 #define FALSE (unsigned int)0x00000000
11 #define TRUE (unsigned int)0x00000001
12
13 /*****
14 /* delayT0Unlocked */
15 /*****
16 /* Esta funcion arranca el timer 0 y programa el registro match0 */
17 /*****
18 void delayT0Unlocked(unsigned int delayInDecimaMiliseg)
19 {
20     TOTCR = 0x02; /* reset timer */
21     TOMR0 = delayInDecimaMiliseg * (12000000 / 10000 - 1);
22     TOMCR = 0x07; /* timer on match */
23     TOTCR = 0x05; /* inicia timer y para cuando se llegue al final de cuenta*/
24 }
25
  
```

Figura 2-2 Fichero timers.h para las definiciones.

FIO4SET3 = 0x01      coloca el pin P4.24 a nivel alto

FIO4CLR3 = 0x01      coloca el pin P4.24 a nivel bajo

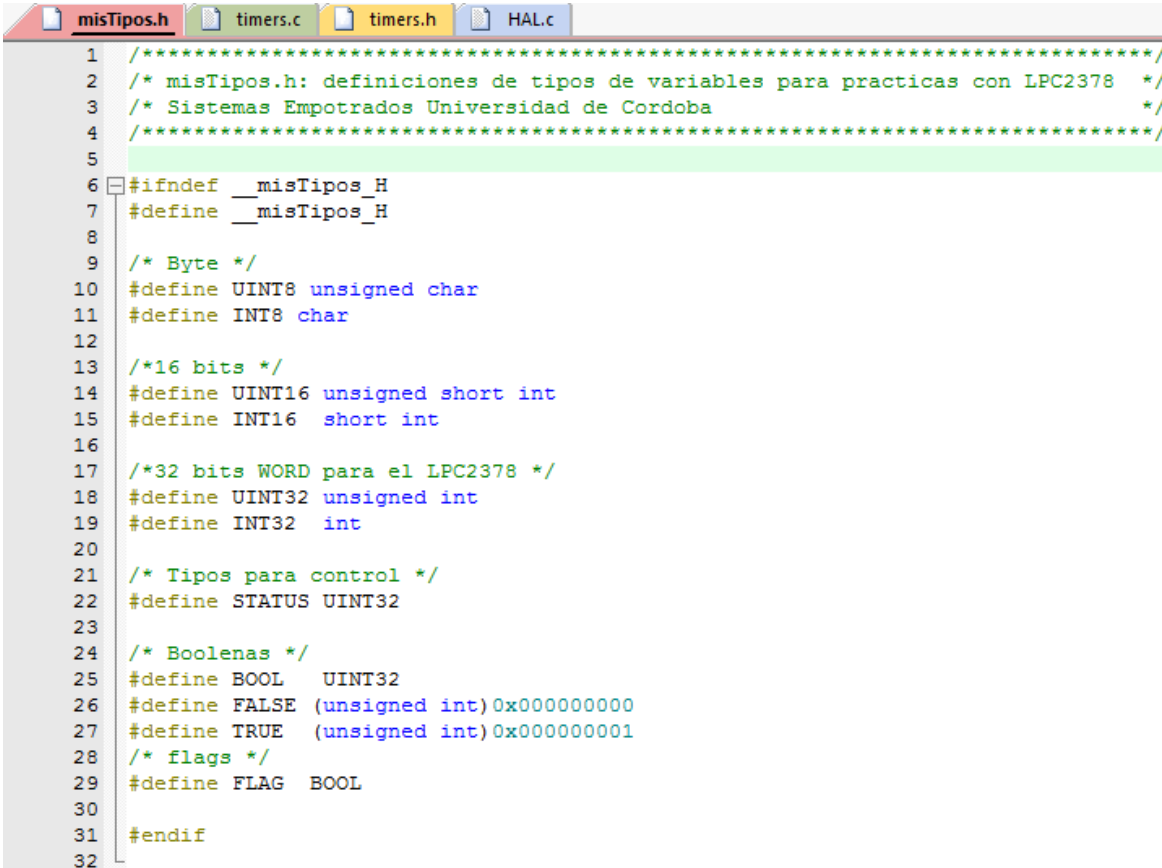
TOTCR = 0x02      Registro de control del timer 0. Resetea el contador

TOMR0 = delayInDecimalMiliseg \* (12000000/10000–1) Coloca el registro match en el valor deseado. El reloj del LPC2378 es 12 MHz.

TOMCR = 0x07      Registro de control del match. Genera una interrupción cuando coincida la cuenta del timer con el valor del registro match.

TOTCR = 0x05      Habilita la cuenta

En la programación de sistemas empotrados, es muy útil disponer de un fichero donde vamos añadiendo los tipos de variables que vamos a utilizar durante todo el diseño. Este fichero sería el `misTipos.h` que podría ser algo así:



```
1  /*****  
2  /* misTipos.h: definiciones de tipos de variables para practicas con LPC2378  */  
3  /* Sistemas Empotrados Universidad de Cordoba  */  
4  *****/  
5  
6  #ifndef __misTipos_H  
7  #define __misTipos_H  
8  
9  /* Byte */  
10 #define UINT8 unsigned char  
11 #define INT8 char  
12  
13 /*16 bits */  
14 #define UINT16 unsigned short int  
15 #define INT16 short int  
16  
17 /*32 bits WORD para el LPC2378 */  
18 #define UINT32 unsigned int  
19 #define INT32 int  
20  
21 /* Tipos para control */  
22 #define STATUS UINT32  
23  
24 /* Booleanas */  
25 #define BOOL UINT32  
26 #define FALSE (unsigned int)0x00000000  
27 #define TRUE (unsigned int)0x00000001  
28 /* flags */  
29 #define FLAG BOOL  
30  
31 #endif  
32
```

**Figura 2-3 Fichero para la definición de tipos de variables.**

Finalmente el programa principal para obtener una señal digital no periódica quedaría, por ejemplo, de la manera siguiente:

T0IR: Registro de interrupción del timer 0. Se lee para ver cuál de las interrupciones posibles están pendientes y se escribe en él para borrar esta interrupción. En este caso el canal 1 en modo de captura.

C:\Users\Carlos Diego\Asignaturas Actuales\Sistemas Empeotrados\Curso 2015-2016\Practicas\PracticasResueltas\Practica2\_G

File Edit View Project Flash Debug Peripherals Tools SVCS Window Help

LPC2378Timers

Project: timersP2

- LPC2378Timers
  - Startup Code
    - LPC2300.s
  - System Calls
    - Retarget.c
      - rt\_misc.h
      - stddef.h
      - stdint.h
      - stdio.h
    - Serial.c
      - LPC23xx.H
      - misTipos.h
  - Source Code
    - timers.c
    - LPC23xx.H
    - stdio.h
    - timers.h
    - HAL.c
    - LPC23xx.H

```

1  /*****
2  /* timers.C: Programa principal
3  /*****
4  /* Sistemas empotrados. Universidad de Córdoba */
5  /*****
6
7  #include <stdio.h>
8  #include <LPC23xx.H> /* LPC23xx definitions */
9  #include "timers.h"
10
11 extern void hardwareInit(void);
12
13 /* Timer0
14 int main (void)
15 {
16     hardwareInit();
17     while(1)
18     {
19         SIGNAL0_PIN_LOW;
20         delayT0Unlocked(PULS00_LOW);
21         while(!(TOIR & 0x00000001));
22         TOIR = 1;
23         SIGNAL0_PIN_HIGH;
24         delayT0Unlocked(PULS00_HIGH);
25         while(!(TOIR & 0x00000001));
26         TOIR = 1;
27     }
28 }
29
  
```

**Figura 2-4 Programa principal de la aplicación.**

Después de haber compilado, enlazado, simulado y depurado el programa, comprobaremos con el osciloscopio, que la señal se genera correctamente, observando y midiendo sus características.

El pin escogido es el P4.24, tenemos que irnos al esquemático de la placa MCB2300 y ver que este se corresponde con el pin 127 y localizarlo en la placa para conectar el osciloscopio.



