

Informe de Laboratorio

Organización y Arquitectura de Computadoras

Autores: Federico Almada, Matías Didolichi

Índice

1. Introducción.....	3
1.1 General	3
1.2 Objetivos generales.....	3
1.3 Objetivos específicos	3
2. Descripción	4
2.1 Hardware.....	4
2.2 Software	4
2.3 Librerías utilizadas	4
2.4 Periféricos.....	5
3. Resultados.....	6
3.1 Proyecto utilizado.....	6
3.2 Modificaciones realizadas	6
3.3 Desarrollo del proyecto.....	7
4. Conclusiones	9
5. Bibliografía consultada.....	10

1. Introducción

1.1 General

El presente informe pretende hacer un seguimiento paso a paso del código implementado para la realización de nuestro proyecto de Laboratorio. Así como también, describir cuales fueron los periféricos utilizados para el mismo y su funcionamiento.

El proyecto realizado es una modificación de un proyecto base que reproduce tonos mediante un periférico de sonido. Para su realización, se han modificado funciones, arreglos y se han implementado librerías de la plataforma LPCXpresso.

El lenguaje de programación utilizado fue C estándar sobre el Microprocesador Cortex M3 del microcontrolador LPC1769.

1.2 Objetivos generales

Analizar el funcionamiento del proyecto base y el de las distintas librerías que ofrece el microprocesador para luego aplicar los objetivos específicos.

1.3 Objetivos específicos


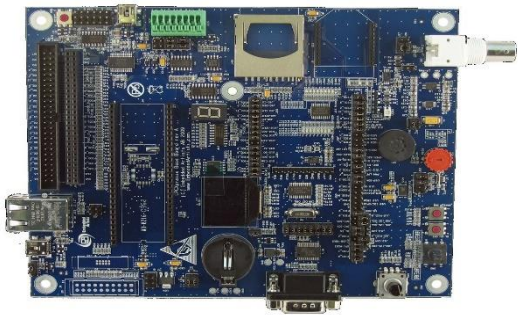
- Combinación de librerías
- Modificación del proyecto base
- Implementación de nuevas funciones

2. Descripción

2.1 Hardware

Se ha utilizado el kit LPCXpresso, que está compuesto de una placa con el debugger LPC-Link integrado, que permite la programación y depuración del firmware en tiempo real y el LPC1769 como target. Además, se utilizó una BaseBoard, la cual incluye periféricos como acelerómetros, sensores de luz, Speaker, Joystick, USB, Ethernet, entre otros.

Material de trabajo

LPC1769	
BaseBoard	

2.2 Software

En cuanto al software, se ha utilizado como entorno de desarrollo el LPCXpresso IDE, basado en Eclipse y de distribución gratuita.

2.3 Librerías utilizadas

Lib_MCU

- Contiene las funciones de inicialización de los módulos del Microcontrolador
- SPI, UART, I2C, Puertos I/O, ADC, DAC, CAN, USB, Ethernet, Timer, PWM

Lib_EaBaseBoard

- Contiene funciones de inicialización de los periféricos de la placa base
- Uart, sensor temperatura, rotary, rgb, pca9532, oled, sensor de iluminación, joystick, display 7 segmentos, memoria flash, memoria eeprom, acelerómetro, librería de caracteres para el oled.

2.4 Periféricos

Se han utilizado los siguientes dispositivos de la BaseBoard:

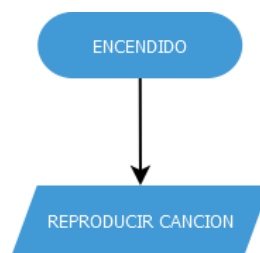
- **Joystick:** Circuito que utiliza un sensor de movimiento compuesto por una perilla que lee 4 direcciones (hacia arriba, hacia abajo, hacia el lado izquierdo, hacia el lado derecho) y un botón en el centro.
- **Speaker:** Dispositivo que reproduce tonos.

3. Resultados

3.1 Proyecto utilizado

Utilizamos el proyecto de ejemplo Speaker_tone, el cual consiste en reproducir un conjunto de tonos, es decir, un sonido.

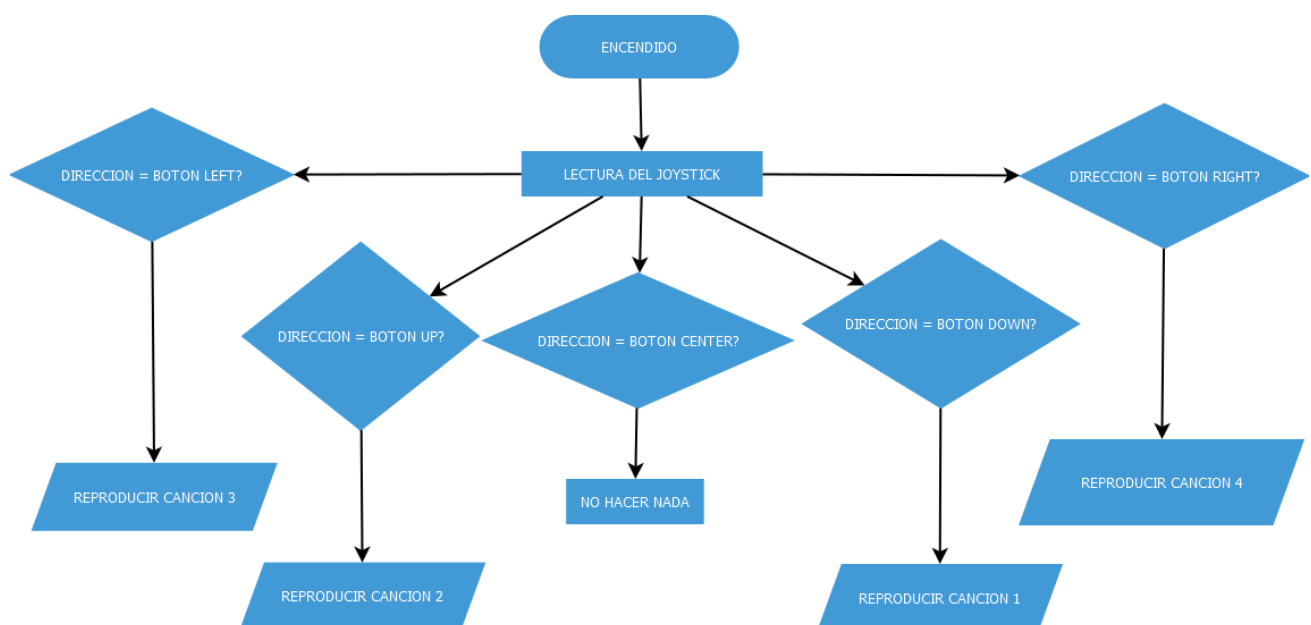
Diagrama del flujo del programa original



3.2 Modificaciones realizadas

- Implementación de la librería del Joystick
- Implementación de nuevos arreglos
- Implementación de funciones del Joystick y condicionales

Diagrama de flujo del programa con sus modificaciones correspondientes



3.3 Desarrollo del proyecto

A continuación, se detalla paso a paso a través de capturas de pantalla todos y cada uno de los procedimientos que fueron necesarios para llevar a cabo el proyecto.

Paso 1:

Implementamos la librería del Joystick.

```
#include "joystick.h"
```

Paso 2:

El proyecto base reproducía solo un arreglo infinitamente, nosotros implementamos 4 arreglos que serán llamados cuando se detecte la lectura de un botón del Joystick (Se detallará en el paso 4).

```
56 static uint8_t * song1 = (uint8_t*)"C2,C2,G2,G2,A2,A2,G4,F2,F2,E2,E2,D2,D2,C4,G2,G2,F2,F2,E2,E2,D4,G2,G2,F2,F2,E2,E2,D4,C2,C2,G2,G2,A2,  
57 static uint8_t * song2 = (uint8_t*)"E2,E2,F2,G2,G2,F2,E2,D2,C2,C2,D2,E2,E4,D4,E2,E2,F2,G2,G2,F2,E2,D2,C2,C2,D2,E2,D3,C1,C2,";  
58 static uint8_t * song3 = (uint8_t*)"E2,E2,E2,D2,E2,G2,B3,G1,E2,E2,G2,E2,E4,D2,D2,F2,F2,A2,A2,F2,F2,G3,G1,E2,E2,G2,E2,E4,D4,E2,E2,E2,D2,  
59 static uint8_t * song4 = (uint8_t*)"A1,B2,c2,A3,A1,A2,B2,A2,F2,G4,G2,A2,G2,E2,F4,F2,G2,F2,D2,E4,E2,F2,E2,C2,E4,E2,F2,A2,c2,d4.c1,B2,F2,
```

Paso 3:

Luego, en la función Main() hemos declarado una variable "joy" de tipo entero con el valor 0 y también llamamos a la función de inicialización del Joystick.

De forma tal que, cuando se encienda el microcontrolador, inicie el periférico Joystick e la BaseBoard.

```
167     uint8_t joy = 0; //inicializo la variable joy con 0  
168     joystick_init(); //inicialo el Joystick
```

Paso 4:

A continuación, cuando se ejecute el bucle while(1), la BaseBoard va a iniciar el sensor de lectura del Joystick, y cuando este detecte que se ha presionado un botón, al valor lo guardará en la variable "joy" declarada anteriormente.

```
while (1)  
{  
    joy = joystick_read(); // Guardo la funcion de lectura del joystick en joy
```

Paso 5:

Finalmente, hemos puesto 5 condiciones para realizar distintas operaciones dependiendo del valor que la variable "joy" posea.

Por ejemplo, si el botón hacia arriba es presionado, se lee uno de los arreglos declarados y la función PlaySong() va a reproducirlo por el Speaker.

```
167     uint8_t joy = 0; //inicializo la variable joy con 0
168     joystick_init(); //inicialo el Joystick
169
170     while (1)
171     {
172         joy = joystick_read(); // Guardo la funcion de lectura del joystick en joy
173
174         if((joy & JOYSTICK_CENTER) !=0){
175             continue;
176         }
177         if((joy & JOYSTICK_DOWN) !=0){
178             playSong((uint8_t*)song1);
179             Timer0_Wait(3000);
180         }
181         if((joy & JOYSTICK_UP) !=0){
182             playSong((uint8_t*)song2);
183             Timer0_Wait(3000);
184         }
185         if((joy & JOYSTICK_LEFT) !=0){
186             playSong((uint8_t*)song3);
187             Timer0_Wait(3000);
188         }
189         if((joy & JOYSTICK_RIGHT) !=0){
190             playSong((uint8_t*)song4);
191             Timer0_Wait(3000);
192         }
193     }
```

Al finalizar la reproducción del sonido, la función Timer0_Wait(3000) bloqueará la condición y la función joystick_read() estará a la espera de una nueva lectura.

4. Conclusiones

Gracias a las clases del Laboratorio, hemos podido comprender mejor cómo funciona una BaseBoard, analizándola de manera directa, e investigando cada una de las partes que componen su arquitectura.

Así como también, durante el desarrollo del proyecto pudimos verificar y comprobar cómo es que viajan los datos ejecutando los distintos proyectos de prueba en el IDE LPCXpresso al microcontrolador, utilizando el lenguaje C estándar.

Además, hemos podido aplicar los conocimientos teóricos vistos en clase, los cuales nos han servido mucho para la realización del proyecto.

Y por último y no menos importante, creemos que, gracias a las clases del laboratorio se solidificaron los conocimientos teóricos vistos durante la cursada.

5. Bibliografía consultada

[1] NXP Semiconductors N.V, "32-bit Arm Cortex®-M3 microcontroller - LPC1769_68_67_66_65_64_63" - 4 May 2018.