|  |
| --- |
| MISE: Microcontroladores y Diseño con Microprocesadores.  Proyecto Explorer 16. G5 |

|  |
| --- |
| G5 |
| Iñigo Olaso  Xabier Olazar |
| Fecha: 10/12/2017 |



**ÍNDICE**

[1 Objetivo 3](#_Toc500697999)

[2 diagrama de flujo 4](#_Toc500698000)

[3 Interfaces 6](#_Toc500698001)

[3.1 Input: 6](#_Toc500698002)

[3.1.1 Pulsadores: 6](#_Toc500698003)

[**3.1.2** Comandos por UART: 6](#_Toc500698004)

[3.2 Output 8](#_Toc500698005)

[3.2.1 UART: 8](#_Toc500698006)

[3.2.2 Pantalla LCD: 8](#_Toc500698007)

# Objetivo

El objetivo de este documento es definir el flujo del programa así como las formas de interactuar con el mismo. Para ello en primer lugar analizaremos un pequeño diagrama de flujo del programa completo. Y en segundo lugar analizaremos las diferentes formas mediante las cuales se posibilita la interacción con el programa, dando lugar a la selección de la configuración de ciertos periféricos, tal y como se solicita en el enunciado del proyecto.

# diagrama de flujo

Se ha elaborado un diagrama de flujo donde se analiza el flujo secuencial de las instrucciones realizadas en el “main”. Es decir, las instrucciones ejecutadas en las interrupciones así como las casuísticas en las que estas son habilitadas no se han tenido en cuenta, ya que sino este documento sería muy extenso y la finalidad es dar unos indicios básicos de cómo se procede.



Figura 1 Diagrama de flujo.

## Ocupacion de la CPU

Para evaluar la ocupación de la CPU se ha decidido usar el Timer 1. Este es configurado durante la inicialización para contar 3ms, de esta forma establecemos un tiempo de ejecución por ciclo del while de 3ms, al final del bucle esperamos a que los 3ms finalicen e interpretamos el tiempo restante mediante instrucciones de suma. Para realizar las cuentas suponemos que esta instrucción equivale a 5 instrucciones de código ensamblador que habrá generado en compilador, multiplicamos el número de cuentas por el número de instrucción al tiempo necesario por la CPU para ejecutar cada instrucción (dependiendo del oscilador seleccionado), y se lo restamos a los 3 mseg. Evidentemente esta tarea hace que cada ciclo tarde más de 3 mseg de ejecución. Pero solo se trata de un método que nos permitirá observar la ocupación de la CPU al cambiar las configuraciones de la CPU.

A continuación, tenemos un ejemplo del código C usado para la implementacion:

# Interfaces

En este apartado se diferenciaran la interfaces input de la output ya que no todas son bidireccionales.

## Input:

Se ha desarrollado un programa que posibilite las configuraciones de los periféricos (osciladores, UART, CAD y Timers) solicitadas en funcionamiento mediante comandos por UART. Para interactuar con el programa es necesario respetar la estructura que definiremos más adelante. Durante la inicialización se exige la selección de osciladores mediante pulsadores, además el control de los leds también se realizará mediante pulsadores.

### Pulsadores:

Tenemos 4 botones:

| RD6 | RD7 | RA7 | RD13 |

Durante la inicialización deberemos tomar una de las siguientes elecciones:

1. Presionar RA7 == OSC=80MHz.
2. Presionar RA7+RD13 == OSC=8MHz.

Una vez inicializado los botones RD6, RD7 y RD13:

1. RD6: Desplaza un led encendido a través del array de leds.
2. RD7: Desplaza los dos leds de los laterales del array hasta el centro y enciendo todos los leds.
3. RD13: Apaga todos los leds.
4. RB3: El pulsador del joystick, enciende todos los leds.

### Comandos por UART:

A continuación definimos la estructura de cada comando. Cada trama está compuesta por 9 caracteres.

**CAD-X-Y-M:**

-X: Sus valores están comprendidos entre 1-2.

+ 1 el conversor 1.

+ 2 para seleccionar el conversor 2.

-Y: Sus valores están comprendidos entre 1-2.

+ 1 para seleccionar la transmisión por interrupción.

+ 2 para seleccionar la transmisión por DMA.

+ 3 para seleccionar la transmisión por DMA en modo Scatter/Gather.

-M: Sus valores están comprendidos entre 1-4. Esta opción solo es válida si has seleccionado el modo DMA. Si estas en modo interrupción simplemente no se tendrán en cuenta.

+ 1 Un bloque + PingPong .

+ 2 Continuo + PingPong.

+ 3 Un bloque.

+ 4 Continuo.

**TIM-X----:**

-X: Sus valores están comprendidos entre 1-3.

+ 1 para seleccionar el timer 6.

+ 2 para seleccionar el timer 8.

+ 3 para seleccionar el timer 2 y 3 (32 bits).

**UAR---Y-M:**

-Y: Sus valores están comprendidos entre 1-2.

+ 1 para seleccionar la transmisión por DMA.

+ 2 para seleccionar la transmisión por interrupción.

-M: Sus valores están comprendidos entre 1-4.

+ 1 Un bloque + PingPong .

+ 2 Continuo + PingPong.

+ 3 Un bloque.

+ 4 Continuo.

**OSC-X----:**

-X: Sus valores están comprendidos entre 1-2.

+ 1 para seleccionar el oscilador de 8MHz.

+ 2 para seleccionar el de 80MHZ.

## Output

En este apartado se analizaran dos de las tres interfaces outputs que forman parte de nuestro proyecto. La tercera seria el array de leds, que dada su simpleza y que ya se ha explicado anteriormente su funcionamiento no se ha visto necesario incluirlo otra vez.

### UART:

A continuación, podemos ver un ejemplo de los mensajes que veremos de Teraterm/Realterm/putty:

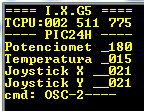


Figura 2 Realterm.

Pasaremos a la descripción de cada línea:

1. I.X.G5: Iñigo y Xabi Grupo 5.
2. TCPU: Tiempo de ocupación de la CPU expresado en 002 milisegundos 511 nanosegundos y 775 microsegundos.
3. Potenciómetro: se han expresado los valores recogidos de 0 a 180º de libertad.
4. En el caso del joytick se han expresado los datos obtenidos de 0 a 45º siendo 21 su estado de reposo.
5. El último comando expresa el último comando recibido por el microprocesador. En este caso el OSC-2----.

### Pantalla LCD:

A continuación, podemos ver un ejemplo de los mensajes que veremos desde el display:



Figura 3 Display

Pasaremos a la descripción de cada bloque:

1. G5: grupo 5.
2. T=15C: Es la temperatura recogida por el sensor. Para la conversión se ha tenido en cuenta el comportamiento descrito en su datasheet.
3. Pot=057: Ángulo al que se encuentra el potenciómetro teniendo en cuenta que gira 180º como máximo
4. Tiempo: 002 510: Es el tiempo de ocupación de la CPU expresado en en 002 milisegundos 510 nanosegundos.