Universidad del Valle de Guatemala — Natalia de León Bercián

Electrónica Digital 2 — carne: 18193

Mayo 2021 — sección: 20

Laboratorio # 6

TivaC - TivaWare

Link del video:

https://youtu.be/ScYC6Lj5EgY

Link del GitHub:

https://github.com/nataliadlb/LABS_REPOSITORIO.git

Parte 2

Periféricos	Registro necesario
Timers	RCGCWTIMER
UARTs	RCGCUART
USB	RCGCUSB
PWM	RCGCPWM
ADC	RCGCADC

Parte 3

Función para delay

```
//Funciones obtenidas de https://gist.github.com/ctring/7f12d812fb594eecc493

void delayMs(uint32_t ui32Ms) {
    SysCtlDelay(ui32Ms * (SysCtlClockGet() / 3 / 1000));
}

void delayUs(uint32_t ui32Us) {
    SysCtlDelay(ui32Us * (SysCtlClockGet() / 3 / 1000000));
}

* El parámetro es la cantidad de ms o us que se quiere para el delay.
```

Código

```
//**************************************************************************
// Laboratorio # 6
// Natalia de León Bericán
// carne: 18193
// Digital 2
```

```c
// Sección 20
// Mayo 2021
//
//*****************************************************************************

#include <stdint.h>
#include <stdbool.h>
#include "inc/tm4c123gh6pm.h"
#include "inc/hw_memmap.h"
#include "driverlib/debug.h"
#include "driverlib/gpio.h"
#include "driverlib/sysctl.h"
#include "inc/hw_types.h"
#include "driverlib/systick.h"

#define LED_ROJO GPIO_PIN_1
#define LED_VERDE GPIO_PIN_3
#define LED_AMARILLO GPIO_PIN_1|GPIO_PIN_3

//*****************************************************************************
//
// VARIABLES
//
//*****************************************************************************
uint8_t i;
uint8_t semaforo = 0;
uint32_t value = 0;

//*****************************************************************************
//
// PROTOTIPOS DE FUNCIONES
//
//*****************************************************************************
void delayMs(uint32_t ui32Ms);


//*****************************************************************************
//
// The error routine that is called if the driver library encounters an error.
//
//*****************************************************************************
#ifdef DEBUG
void
__error__(char *pcFilename, uint32_t ui32Line)
{
    while(1);
}
#endif

//*****************************************************************************
//
// CONFIGURACION
//
//*****************************************************************************
int
```

```
main(void){
    //Config de reloj
    SysCtlClockSet(SYSCTL_SYSDIV_5 | SYSCTL_USE_PLL | SYSCTL_XTAL_16MHZ |
SYSCTL_OSC_MAIN);

    // Enable the GPIO port that is used for the on-board LED.
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);

    // Check if the peripheral access is enabled.
    while(!SysCtlPeripheralReady(SYSCTL_PERIPH_GPIOF)){
    }

    // Enable the GPIO pins. OUTPUTS/INPUTS
    GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, LED_ROJO); //RED LED
    GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_2); //BLUE LED
    GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_3); //GREEN LED
    GPIOPinTypeGPIOInput(GPIO_PORTF_BASE, GPIO_PIN_4); //PUSH COMO ENTRADA
    GPIOPadConfigSet(GPIO_PORTF_BASE, GPIO_PIN_4, GPIO_STRENGTH_2MA,
GPIO_PIN_TYPE_STD_WPU); //weak pull up


//*****************************************************************************
    // Loop forever.

//*****************************************************************************

    while(1){

        if ( !GPIOPinRead(GPIO_PORTF_BASE, GPIO_PIN_4) ){//Si se presiona el
boton, se activa una variable
            semaforo = 1;
        }


        if(semaforo == 1 && (GPIOPinRead(GPIO_PORTF_BASE, GPIO_PIN_4))){
//Debouncing: solo hasta que la variable se encienda y el boton ya no este
presionado

            // GREEN LED ON
            GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3,
LED_VERDE);
            delayMs(700);


            for (i = 0; i < 4; i++){//Parpadeo
                // GREEN LED ON
                GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3,
LED_VERDE);
                delayMs(300);

                // GREEN LED OFF
                GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3,
0x0);
                delayMs(300);
            }
```

```
            // YELLOW LED ON
            GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3,
LED_AMARILLO);
            delayMs(1500);

            // YELLOW LED OFF
            GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3,
0x0);
            delayMs(1000);

            // RED LED ON
            GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3,
LED_ROJO);

            semaforo = 0;

        }

    }
}


//*****************************************************************************
//
// FUNCIONES
//
//*****************************************************************************
void delayMs(uint32_t ui32Ms) {  //Funcion obtenida de
https://gist.github.com/ctring/7f12d812fb594eecc493
    SysCtlDelay(ui32Ms * (SysCtlClockGet() / 3 / 1000));
}
```