Universidad del Valle de Guatemala

Digital 2 - Laboratorio

Kurt Kellner

Pablo Rene Arellano Estrada

151379

**MINI-PROYECTO 2**

**-I2C con MPU9250-**

**ARCHIVOS Y LIBRERIAS .C Y .H:**

|  |  |  |
| --- | --- | --- |
| **PIC** | **ARDUINO** | **ESP32** |
| PIC\_MPU\_6050.C | Serial\_Arduino.cc | Adaruitio\_07\_Digital\_out.cc  Config.h |
| Configuracion\_Header\_File.H |  | Adaruitio\_07\_Publish.cc  Config.h |
| MPU6050\_res\_define.H |  |  |
| I2C\_Master\_File.C LCD\_SPI.H  I2C\_Source\_File.C |  |  |
| USART\_Header\_File.H  USART\_Source\_FileI.C |  |  |

**OBSERVACIONES:**

* No se pudo encontrar el manual del sensor MPU9250, solo del 6500, por lo que se usaron los registros encontrados en este manual, y en la pagina de [KHALED MAGDY](https://deepbluembedded.com/author/admin/).
* La comunicación serial simultanea entre el PIC y el ESP32 no fue posible, por lo que se transmitieron datos del PIC al ESP32 con UART, pero para los leds se conectaron pines del ESP32 al PIC para efectos de demostración usando el regulador de voltaje.
* Adaruit no permitió enviar y recibir datos de manera razonable con los botones y los medidores, por lo que se hicieron en archivos separados y en momento separados para evitar los 30 cambios por minuto en los Feeds.

**PSEUDOCODIGO-MAESTRO:**

1. Se llama a las librerias correspondeintes:
   * xc.h
   * Variables de ancho definido stdint.h
   * Tipos de variables, macros, entradas y salidas stdio.h
   * Libreria de comunicacion SPI
   * Libreria de comunicacion USART
   * Libreria de pantalla LCD
2. Puerto PORTCbits.RC0 = 0 para SS de potenciometro;

* Se envia dato por SPI\_Enviar ()
* Se recibe dato enviado de esclavo SPI\_Recibir()
* Puerto PORTCbits.RC0 = 1 para deseleccionar potenciometero;

1. Puerto PORTCbits.RC1 = 0 para SS de botones;

* Se envia dato por SPI\_Enviar ()
* Se recibe dato enviado de esclavo SPI\_Recibir()
* Puerto PORTCbits.RC1 = 1 para deseleccionar botones

1. Puerto PORTCbits.RC2 = 0 para SS de semaforo;

* Se envia dato por SPI\_Enviar ()
* Se recibe dato enviado de esclavo SPI\_Recibir()
* Puerto PORTCbits.RC2 = 1 para deseleccionar semaforo

1. En Lcd se coloca el cursor en (0,1)

* Se escribe ("ADC: BTN: TEMP:")

1. En Lcd se coloca el cursor en (2,1)

* Dato\_pot de PIC1
* pot = voltaje \* 4 para conversion
* unidad = pot / 100 para unidad
* x1 = pot % 100 y x2 = x1 / 10 para milésima
* y1 = x1 % 10 y y2 = y1 / 1 para centesima
* Escribir en LCD (unidad)
* Escribir en LCD (punto)
* Escribir en LCD (milesima)
* Escribir en LCD (centesima)

1. En Terminal Virtual escribir:

* Escribir en Terminal (ADC:)
* Escribir en Terminal (unidad)
* Escribir en Terminal (punto)
* Escribir en Terminal (milesima)
* Escribir en Terminal (centesima)

1. En Lcd se coloca el cursor en (2,7)

* Dato\_botones de PIC2
* Si dato es < 10:
  + Escribir en LCD (cero)
  + Escribir en LCD (dato\_botones)
* Si dato > 10:
  + w1 = dato\_botones/10
  + w2 = dato\_botones%10
  + Escribir en LCD (w1)
  + Escribir en LCD (w2)

1. En Terminal Virtual escribir:

* Escribir en Terminal (CONT:)
* Si dato es < 10:
  + Escribir en Terminal (cero)
  + Escribir en Terminal (dato\_botones)
* Si dato es > 10:
  + Escribir en Terminal (w1)
  + Escribir en Terminal (w2)

1. En Lcd se coloca el cursor en (2,12)

* Dato\_semaforo de PIC3
* decena = dato\_botones/10
* unidad = dato\_botones%10
* Escribir en LCD (decena)
* Escribir en LCD (unidad)
* Escribir en LCD (grados C)

1. En Terminal Virtual escribir:

* Escribir en Terminal (TEMP:)
* Escribir en Terminal (decena)
* Escribir en Terminal (unidad)
* Escribir en Terminal (grados Centigrados)

1. Mandar datos a terminar virtual:
   * Escribir valor sobre TXREG
   * Dar tiempo a que termine de escribir el valor en TXREG

**SLAVE\_POT:**

1. Se llama a las librerias correspondeintes:
   * xc.h
   * Variables de ancho definido stdint.h
   * Tipos de variables, macros, entradas y salidas stdio.h
   * Libreria de comunicacion SPI
   * Libreria de conversion ADC
2. Congifura interrupcion:

* Si la bandera se levanta (TMR0IF == 1):
  + Llamar a la funcion de conversion
  + Colocar TMR0 = 100 para reiniciar

1. Si se levanta bandera interrupcion de SPI (SSPIF == 1)

* Se recibe dato de maestro = SPI\_Recibir()
* Se envia conversion de potensiometro SPI\_Enviar (ADRESH)

1. Se configura puertos;
   * TRISAbits.TRISA0 = 1 como entrada
   * ANSELbits.ANS0 = 1 como Analogico para potenciometero
   * TRISAbits.TRISA5 = 1 como entrada
   * ANSELbits.ANS5 = 0 como Digital
   * PORTAbits.RA5 = 1 para apagar comunicacion al inicio (SS)
   * TRISCbits.TRISC3 = 1 como entrada de reloj Maestro
   * TRISCbits.TRISC4 = 1 como entrada de dato Maestro
   * TRISCbits.TRISC5 = 0 como salida de dato Esclavo a Maestro
2. Otras configuraciones:

* Se habilitan interrupciones globales con GIE = 1.
* Se habilita interrupciones de Timer 0 con T0IE = 1.
* Se configura oscilador a 4MHz con IRCF.
* Se coloca el Timer0 con temporizador como T0CS = 0.
* El Timer0 se coloca con Prescaler con PSA = 0.
* Se coloca el Prescaler en 8
* Se inicia el Timer0 en 100.

1. Configuracion de librerias

* Se inicia en Canal 0
* Se inicia en modo Esclavo con CKP = 0 y CKE = 1

1. Se llama a Funcion de conversion

* Se coloca el GO\_DONE = 1 para iniciar conversion
* Se da un Delay de 10ms
* Si ya se hizo la conversion y GO\_DONE = 0:
  + Se coloca el GO\_DONE = 1 para seguir conversion
  + Se envia el registro ADRESH a PORD para mostrar con leds

**SLAVE\_PUSH:**

1. Se llama a las librerias correspondeintes:
   * xc.h
   * Variables de ancho definido stdint.h
   * Tipos de variables, macros, entradas y salidas stdio.h
   * Libreria de comunicacion SPI
   * Libreria de conversion ADC
2. Congifura interrupcion:

* Si la bandera se levanta (RBIF == 1):
  + Para evitar Mismatch se escribe en Puerto B
  + Se llama a funcion incrementar
  + Se llama a funcion decrementar
  + La bandera RBIF se apaga.

1. Si se levanta bandera interrupcion de SPI (SSPIF == 1)

* Se recibe dato de maestro = SPI\_Recibir()
* Se envia valor del puerto (PORTD)
* Se apaga SSPIF.

1. Se configura puertos;
   * TRISAbits.TRISA5 = 1 como entrada
   * ANSELbits.ANS5 = 0 como Digital
   * PORTAbits.RA5 = 1 para apagar comunicacion al inicio (SS)
   * TRISCbits.TRISC3 = 1 como entrada de reloj Maestro
   * TRISCbits.TRISC4 = 1 como entrada de dato Maestro
   * TRISCbits.TRISC5 = 0 como salida de dato Esclavo a Maestro
   * TRISBbits.TRISB2 = 1 como salida para boton 1 y se apagan
   * TRISBbits.TRISB3 = 1 como salida para boton 2 y se apagan
2. Otras configuraciones:

* Se habilitan interrupciones globales con GIE = 1.
* Se habilita interrupciones del Puerto B con PEIE = 1.
* Se configura oscilador a 4MHz con IRCF.
* Se coloca el Timer0 con temporizador como T0CS = 0.
* El Timer0 se coloca con Prescaler con PSA = 0.
* Se coloca el Prescaler en 8
* Se inicia el Timer0 en 100.

1. Configuracion de librerias

* Se inicia en Canal 0
* Se inicia en modo Esclavo con CKP = 0 y CKE = 1

1. Se llama a Funcion Incremenar con Antirebote (Igual para Decrementar pero con PORTD - -)

* Se verifica si hay un cambio en RB2:
  + Se hace un delay con un ciclo For de conteo i = 6
  + Se indica que el boton ha dejado de ser presionado
  + Si el delay ha terminado y se indica que el boton se presiono exitosamente
    - Se incrementa PORTD
    - Se inidica que boton ya no ha sido presionado nuevamente
  + El delay creado y usado anteriormente de boton presionado se reinicia a 0.
* Se realiza un nuevo conteo de seguridad para ver si esta suelto
* Se indica que el boton esta suelto
  + Presionado = 0 para que pueda ser presionado otra vez
  + El delay creado y usado anteriormente de boton suelto se reinicia a 0.

**SLAVE\_SEMAFORO:**

1. Se llama a las librerias correspondeintes:
   * xc.h
   * Variables de ancho definido stdint.h
   * Tipos de variables, macros, entradas y salidas stdio.h
   * Libreria de comunicacion SPI
   * Libreria de conversion ADC
   * Se definen los colores del semaforo
     1. LED\_rojo
     2. LED\_amarillo
     3. LED\_verde
2. Congifura interrupcion:

* Si la bandera se levanta (TMR0IF == 1)
  + Llamar a la funcion de conversion
  + Se apaga la bandera TMR0IF = 0
  + Colocar TMR0 = 100 para reiniciar

1. Si se levanta bandera interrupcion de SPI (SSPIF == 1)

* Se recibe dato de maestro = SPI\_Recibir()
* Se envia conversion de potensiometro SPI\_Enviar (ADRESH)
* Se apaga SSPIF.

1. Se configura puertos;
   * TRISAbits.TRISA0 = 1 como entrada
   * ANSELbits.ANS0 = 1 como Analogico para LM31
   * TRISAbits.TRISA5 = 1 como entrada
   * ANSELbits.ANS5 = 0 como Digital
   * PORTAbits.RA5 = 1 para apagar comunicacion al inicio (SS)
   * TRISCbits.TRISC3 = 1 como entrada de reloj Maestro
   * TRISCbits.TRISC4 = 1 como entrada de dato Maestro
   * TRISCbits.TRISC5 = 0 como salida de dato Esclavo a Maestro
2. Otras configuraciones:

* Se habilitan interrupciones globales con GIE = 1.
* Se habilita interrupciones de Timer 0 con T0IE = 1.
* Se configura oscilador a 4MHz con IRCF.
* Se coloca el Timer0 con temporizador como T0CS = 0.
* El Timer0 se coloca con Prescaler con PSA = 0.
* Se coloca el Prescaler en 8
* Se inicia el Timer0 en 100.

1. Configuracion de librerias

* Se inicia en Canal 0
* Se inicia en modo Esclavo con CKP = 0 y CKE = 1

1. Se llama a Funcion de conversion

* Se coloca el GO\_DONE = 1 para iniciar conversion
* Se da un Delay de 10ms
* Si ya se hizo la conversion y GO\_DONE = 0:
  + Se coloca el GO\_DONE = 1 para seguir conversion
  + Se envia el registro ADRESH y se llama a funcion Semaforo

1. Funcion Semaforo:

* Si es a menor a 25 grados centrigrados en LM31:
  + LED\_verde = 1
* Si es a entre 25 y 36 grados centigrados en LM31:
  + LED\_amarillo = 1
* Si es a mayor a 37 grados centrigrados en LM31:
  + LED\_rojo = 1

**LIBRERIA SPI:**

1. Para PIC Maestro:
   * Establecer SS en bits para POT, BOTONES Y SEMAFORO
   * Establecer bits de entrada y salida
   * Establecer bit de CLK

* Con bit de registro SSPCON se elige:
  + SPI Master Mode Fosc/4
  + SPI Master Mode Fosc/16
  + SPI Master Mode Fosc/64
  + SPI Master Mode TMR2
* Elegir forma de transmission con CKP y CKE
  + Low Level en en borde de subida
  + High Level en borde de caida
  + Low Level en borde de caida
  + High Level en borde de subida
* Se muestra a la mitad del dato con SMP
* Se habilita puertos para comunicacion serial con SSPEN

1. Para PIC Esclavo:
   * Establecer SS para comunicar con maestro
   * Establecer bits de entrada y salida
   * Establecer bit de CLK

* Con bit de registro SSPCON se elige:
  + Slave Mode SS enable
  + Slave Mode SS disable
* Elegir forma de transmission con CKP y CKE igual que el maestro
  + Low Level en en borde de subida
  + High Level en borde de caida
  + Low Level en borde de caida
  + High Level en borde de subida
* Se muestra el dato a la mitad con SMP
* Se habilita puertos para comunicacion serial con SSPEN

1. Se establece funcion para envair datos:
   * Se escribe en el buffer SSPBUF dato de SSPSR
2. Se establece funcion para recibir datos:
   * Se envia dato al buffer que recibe de SSPSR.

**LIBRERIA LCD:**

1. Escribir sobre el Puerto de 8 bits;
   * Escribir sobre Puerto en Pic
2. Enviar a LCD;

* Elegir commando a LCD
* Se mueve dato a Puerto
* Cambiar E de 1 a 0 para indicar trasmision.

1. Inicializar LCD
   * Inicializar Puerto de 8 bits
   * Inicializar RS = 0 y RW=0.
   * Enviar a LCD 00111000 para dos lineas de codigo y 5x11 caracteres
   * Enviar a LCD 00001100 como activado y encendido
   * Enviar a LCD 00000001 para borrar la visualizacion anterior
   * Enviar a LCD 00000110 para incremento y desplazamiento desactivado
2. Limpiar LCD;
   * Enviar a LCD 00000001 para limpiar pantalla
3. Colocar posicion de cursor:

* Si se escribe en fila 1:
  + Se establece variable de columna 0x80
  + Se manda posicion a LCD
* Si se escribe en fila 2:
  + Se establece variable de columna 0XC0
  + Se manda posicion a LCD

1. Escribir cadena de caracteres o String:

* Escribir una lista a[i]
* Hasta que este vacio escribir cadena de caracteres

1. Mover a la izquierda:
   * Mandar 0001\_1000 como comando a la LCD
2. Mover a la derecha:
   * Mandar 0001\_1100 como comando a la LCD
3. Escribir un solo caracter:
   * Se coloca RS como dato
   * Se escribe un byte al Puerto de 8 bits
   * Se mueve E de 1 a 0 para que se transmitan los datos

**LIBRERIA USART:**

1. Seleccionar el Baud Rate en 9600:

* SPBRG = (4MHz/(16\*9600))-1 = 25

1. Configurar Registro TXSTA;

* TX9 = 0 ya que es solo para 8 bits
* TXEN = 1 para permitir transmision
* SYNC = 0 para comunicacion Asincrona
* BRGH = 1 para alta velocidad

1. Configurar Registro RCSTA:

* SPEN = 1 para permitir comunicacion serial
* RX9 = 0 para 8 bits
* CREN = 1 para permitir recibir

1. RCIF = 0 para iniciar bandera en cero
2. TXIF = 0 para iniciar transmision apagada

**LIBRERIA ADC:**

1. Seleccionar canal con ADCON0

* Canal 0
* Canal 1
* Canal 2
* ---
* Canal 13

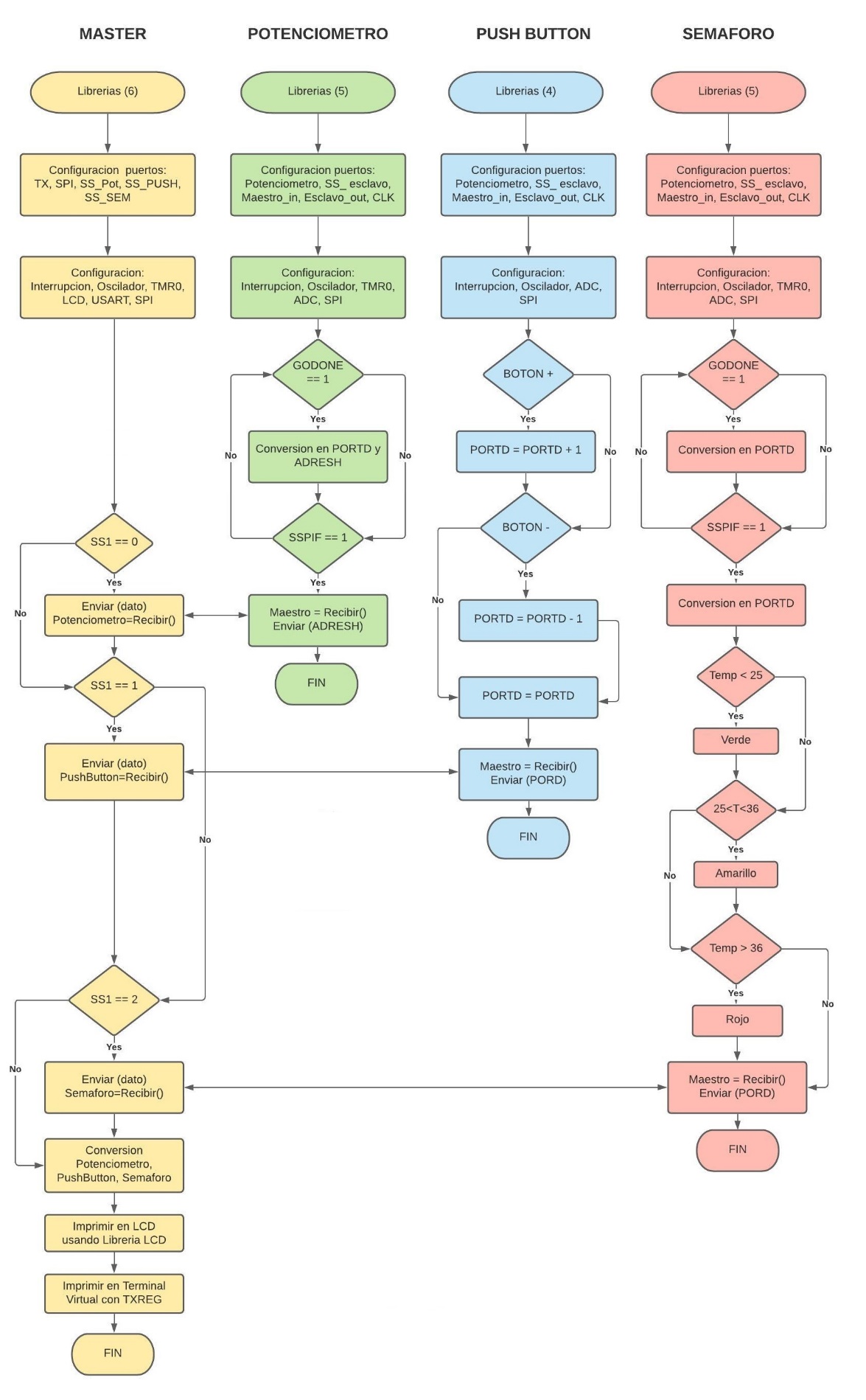
1. Configurar ADCON0;

* ADCS1 = 1 para frecuencia de oscilacion / 32
* GO\_DONE = 0 para iniciar con la conversion apagada
* ADON = 1 para habilitar conversion

1. Configurar ADCON1

* ADFM = 0 para justificar a la izquierda
* Colocar VCFG1 = 0 para colocar tierra iguala a cero
* Colocar voltaje de referencia:
  + VCFG0 = 0 para Vref+ en 5V
  + VCFG0 = 1 para Vref- en 0Vo en voltaje externo.

**DIAGRAMA DE FLUJO:**

****

