Proyecto 3: Garra

Ricardo Pellecer Orellana (pel19072)

27 de Noviembre 2020

1. Descripción

Este proyecto es un brazo con dos ejes de giro y una garra controlados a través del microcontrolador PIC 16F887 que cuenta con las siguientes funciones:

- Escoger si la garra será controlada de forma manual -con el joystick- o remota -por la computadora-. Esto se realizará a través de la interfaz realizada en Python con la librería de PyQt5 y con ayuda de la herramienta PyQt5 Designer. Esta interfaz cuenta con botones para seleccionar el modo de funcionamiento, posiciones de los servos y el usuario que se encuentra usando la garra. Estos datos se envían al PIC a través del módulo TTL.
- Funcionamiento Manual:

  Recibe la señal analógica de dos potenciómetros (o un joystick) y las convierte -a través del ADC- en señales digitales, para lo cual necesita multiplexar los canales analógicos que utiliza el PIC.

  Los señales analógicas convertidas a digitales se mapean a valores para generar señales cuadradas con pulsos modulables para poder controlar 2 servomotores -uno para cada eje de giro-.

  Generar una señal cuadrada con dos valores distintos de ancho de pulso que se pueden escoger a través de un botón para controlar la apertura de la garra.
- Remoto:

  Recibir la posición de los servomotores de los ejes de giro mandada por la interfaz -pueden cambiarse a través de sliders- y utilizarlos para crear las señales cuadradas de ancho de pulso modulable -emulan el funcionamiento del joystick-.

  Recibir la señal de la computadora si se quiere abrir o cerrar la garra y utilizar esta variable como el botón para seleccionar el ancho de pulso que controla la garra para ver si esta se cierra o se abre.
- Ambos modos de funcionamiento guardan en la EEPROM el usuario que está utilizando la maquinaria. Cada vez que hay un cambio de usuario, este se registra en la EEPROM.
- Este dato se lee de la EEPROM cada vez que hay un cambio de usuario y se envía a la computadora para registrar el último usuario que utilizó la garra. Este dato debe ser transformado a formato ASCII y mandado a través del puerto TX hacia la computadora. Esta recibe los datos a través de un módulo TTL. Nótese que es importante que esta conexión TX del PIC vaya hacia el RX del módulo.

2. Cálculos

Para este proyecto se utilizaron las interrupciones del Timer0, ADC, TX y del RX que ofrece el PIC16F887. Con el Timer0 se generó la señal cuadrada de ancho de pulso modulable. Estas señales debían de durar 0.5ms encendidas y 2.5ms apagadas o viceversa -según la selección realizada, ya sea 0° o 180°-. Para realizar los cálculos se tomó en cuenta que se utilizó el reloj interno por defecto del PIC (4MHz).

1. Timer0: El valor que se mueve al registro TMR0 está dado por la expresión $256 - \frac{T*10^6}{256}$, donde T es el tiempo deseado para la interrupción. En este caso, la interrupción deseada es de 0.5ms y 2.5ms, por lo que el valor obtenido es de 253 y 245 respectivamente. Este valor toma en cuenta un prescaler de 256.

3. Diagramas de Flujo

3.1. Diseño General

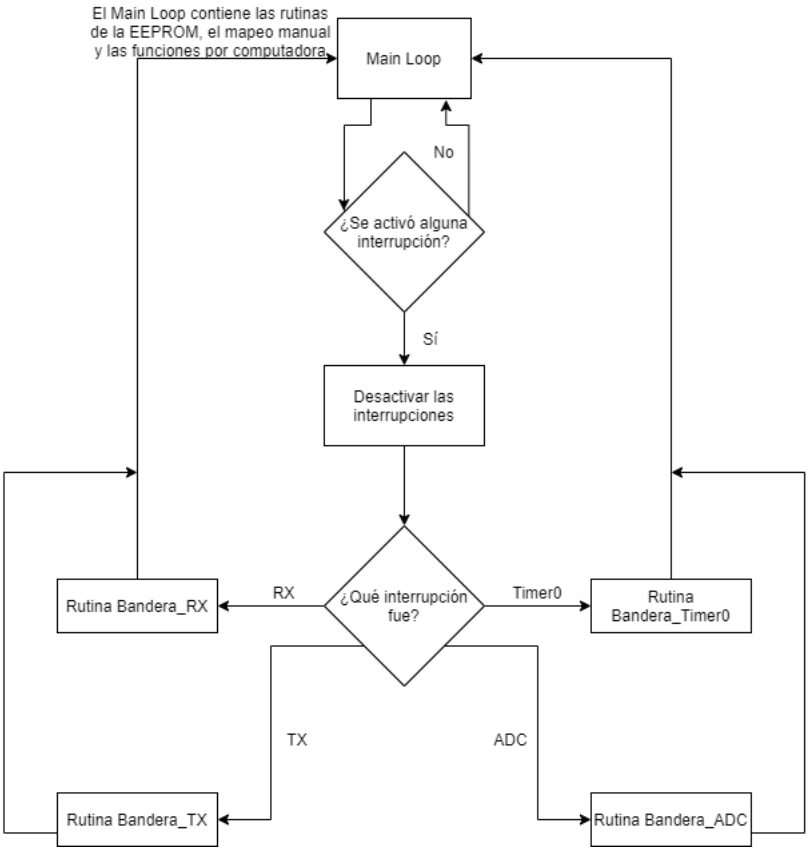

Figura 1: Diseño General

3.2. Módulo ADC

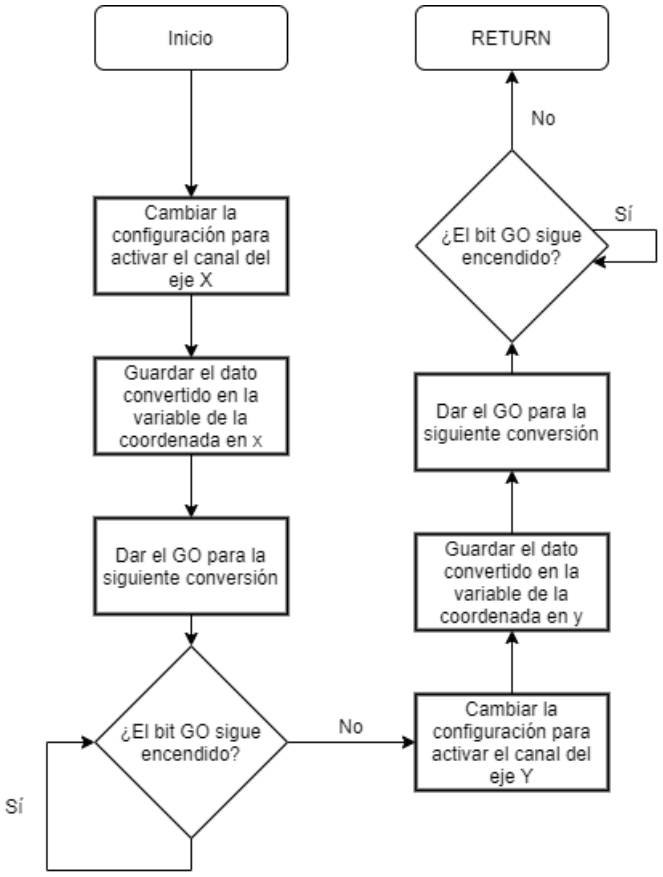

Figura 2: Conversión ADC

3.3. Módulo TX

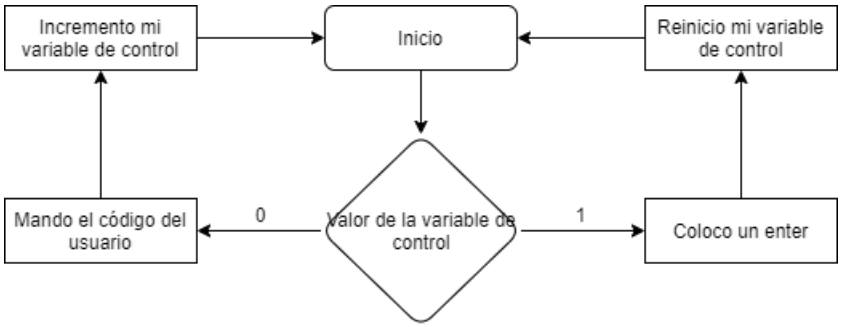

Figura 3: Envío de Datos

3.4. Módulo RX

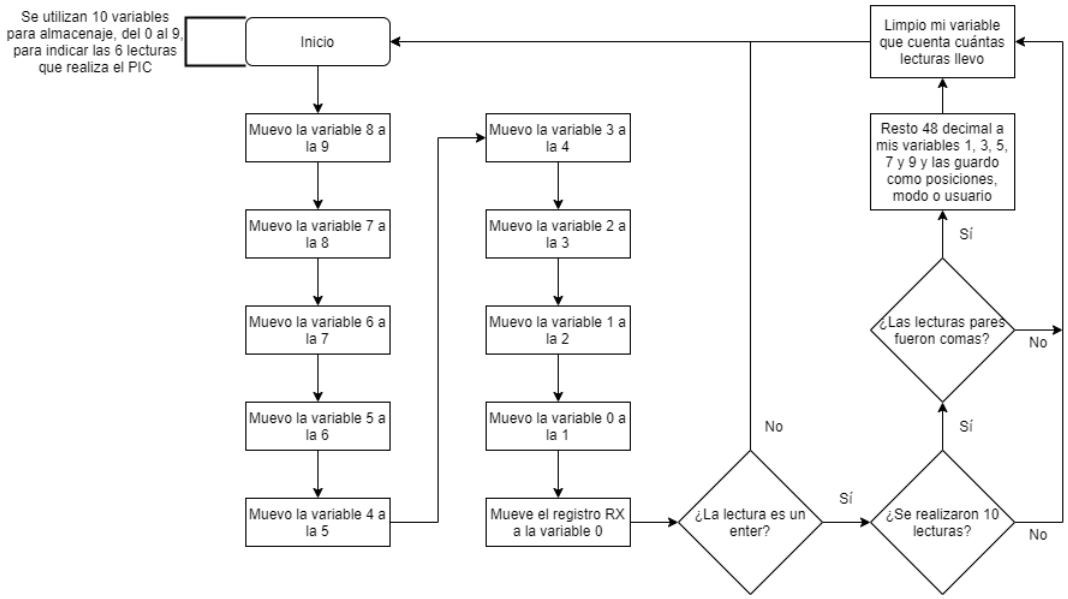

Figura 4: Recepción de Datos

3.5. Módulo EEPROM

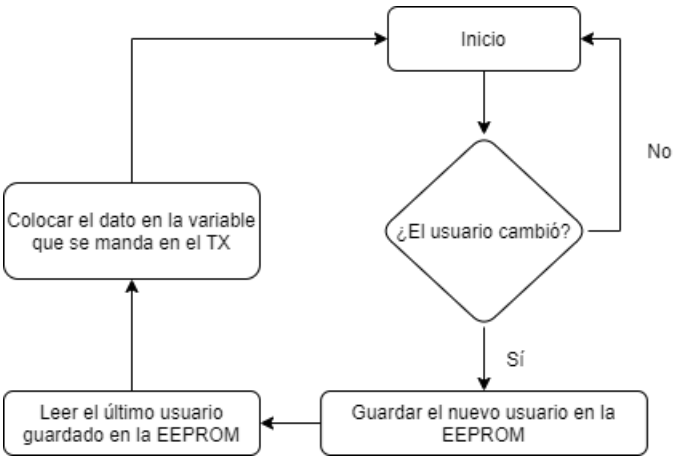

Figura 5: Almacenaje de Datos

4. Esquemático

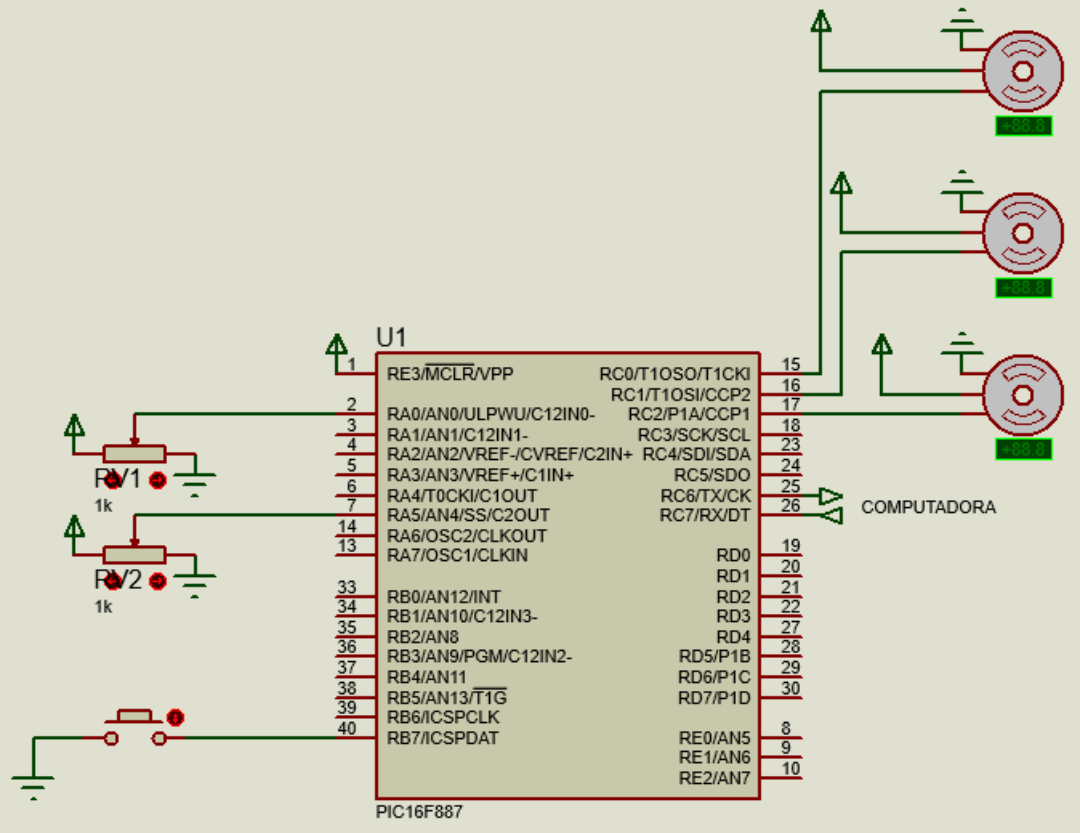

Figura 6: Esquemático (utilizando Proteus)

5. Código

5.1. Assembly

```
;*******************************************************************************
;                                                                              *
;    Filename:     Code -> code.asm        *
;    Date:                   03/11/2020                                        *
;    File Version:           v.1                                               *
;    Author:                 Ricardo Pellecer Orellana                         *
;    Company:                UVG                                               *
;    Description:            PROYECTO FINAL                                    *
;                                                                              *
;*******************************************************************************

#include "p16f887.inc"

; CONFIG1
; __config 0xE0D4
 __CONFIG _CONFIG1, _FOSC_INTRC_NOCLKOUT & _WDTE_OFF & _PWRTE_OFF & _MCLRE_OFF & _CP_OFF & _CPD_OFF & _BOREN_OFF & _IESO_OFF & _FCMEN_OFF & _LVP_OFF
; CONFIG2
; __config 0xFFFF
 __CONFIG _CONFIG2, _BOR4V_BOR40V & _WRT_OFF

; TODO PLACE VARIABLE DEFINITIONS GO HERE


GPR_VAR UDATA
    W_TEMP RES     1 ; PARA GUARDAR INFO MIENTRAS SE EJECUTA LA INTERRUPCIÓN
    STATUS_TEMP RES     1
    VAR_ADCX RES     1
    VAR_ADCY RES     1
    FLAG_ANTIREBOTE RES     1
    FLAG_ADC RES     1
    TX_FLAG RES     1
    ALTO RES     1
    BAJO RES     1
    TOGGLE RES     1
    RXB0 RES     1
    RXB1 RES     1
    RXB2 RES     1
    RXB3 RES     1
    RXB4 RES     1
    RXB5 RES     1
    RXB6 RES     1
    RXB7 RES     1
    RXB8 RES     1
    RXB9 RES     1
    SERVO_GARRA RES     1
    SERVO_EJE1 RES     1
    SERVO_EJE2 RES     1
    SERVO_FUN RES     1
    USUARIO RES     1
    LAST_USER RES     1
    USER_FLAG RES     1
    CUENTARX RES     1
    DIVISION RES     1
    CONT1 RES     1
    MODO RES     1


;*******************************************************************************
; RESET VECTOR
;*******************************************************************************

RES_VECT    CODE    0x0000 ; processor reset vector
    GOTO    START ; go to beginning of program

;*******************************************************************************
; ISR VECTOR
;*******************************************************************************

ISR_VECTOR  CODE    0x0004

PUSH:     ; PUSHEA LOS DATOS DE STATUS Y W A UNA VARIABLE TEMPORAL EN CASO SE VEAN AFECTADOS EN LA INTERRUPCIÓN
    BCF     INTCON, GIE     ; DESACTIVA INTERRUPCIONES PARA EVITAR INTERRUPCIONES MIENTRAS SE ESTÁ EN EL ISR
    MOVWF   W_TEMP
    SWAPF   STATUS, W
    MOVWF   STATUS_TEMP

ISR:
    BTFSC   PIR1, ADIF     ; CÓDIGO PARA SABER DE PARTE DE QUIÉN ES LA INTERRUPCIÓN
    CALL    BANDERA_ADC
    BTFSC   INTCON, TOIF
    CALL    BANDERA_TIMERO
    BTFSC   PIR1, RCIF
    CALL    BANDERA_RX
    BTFSC   PIR1, TXIF    ; CÓDIGO PARA SABER DE PARTE DE QUÉ TIMER SE REALIZÓ LA INTERRUPCIÓN
    CALL    BANDERA_TX

POP:     ; POPEA LOS DATOS DE UNA VARIABLE TEMPORAL A STATUS Y W PARA RECUPERAR CUALQUIER DATO PERDIDO EN LA INTERRUPCIÓN
    SWAPF   STATUS_TEMP, W
    MOVWF   STATUS
    SWAPF   W_TEMP, F
    SWAPF   W_TEMP, W
RETFIE     ; INCLUYE LA REACTIVACION DEL GIE

;*******************************************************************************
; TABLA DE CONVERSIONES
;*******************************************************************************
; SE USARA ESTA TABLA PARA REALIZAR LAS CONVERSIONES A ASCII
; Y ENVIAR LOS DATOS EN EL FORMATO DESEADO
CONVERSIONES:
    ANDLW   b'00001111'
    ADDWF   PCL, F
    RETLW   .48 ;0
    RETLW   .49 ;1
    RETLW   .50 ;2
    RETLW   .51 ;3
    RETLW   .52 ;4
    RETLW   .53 ;5
    RETLW   .54 ;6
    RETLW   .55 ;7
```

```
    RETLW   .56 ;8
    RETLW   .57 ;9
    RETLW   .65 ;A
    RETLW   .66 ;B
    RETLW   .67 ;C
    RETLW   .68 ;D
    RETLW   .69 ;E
    RETLW   .70 ;F

;*********************************************************************************
; MAIN PROGRAM
;*********************************************************************************
MAIN_PROG   CODE     0x0100                 ; let linker place main program

START
SETUP:
    CALL    CONFIGURACION_BASE     ; EXPLICACIONES EN LA SECCIÓN DE CONFIGURACIONES
    CALL    CONFIGURACION_PWM
    CALL    CONFIGURACION_TIMER0
    CALL    CONFIGURACION_TIMER2
    CALL    CONFIGURACION_INTERRUPCION
    CALL    CONFIGURACION_TX_9600
    CALL    CONFIGURACION_RX
    CALL    CONFIGURACION_ADC

;*********************************************************************************
; MAIN LOOP
;*********************************************************************************
LOOP:
    BANKSEL TRISA
    BSF     PIE1, TXIE
    BANKSEL PORTA
    CALL    EEPROM_ESCRITURA
    MOVLW   .9
    SUBWF   SERVO_FUN, W
    BTFSC   STATUS, Z
    GOTO    AUTOMATIC
    MANUAL:
BTFSS   PORTB, RB7 ; REVISA SI EL BOTÓN DE CAMBIO DE ESTADO SE HA PRESIONADO
CALL    ANTIR ; INDICA QUE YA SE PRESIONÓ
BTFSC   PORTB, RB7 ; NO EJECUTA LA INSTRUCCIÓN SI SIGUE PRESIONADO EL BOTÓN
CALL    MODO_FUNCIONAMIENTO ; SE EJECUTA EL CAMBIO DE ESTADO
CALL    MAPPEO
GOTO    LOOP
    AUTOMATIC:
MOVLW .9
SUBWF   SERVO_GARRA, W
BTFSC   STATUS, Z
GOTO    AUTOMATIC_HIGH
AUTOMATIC_LOW:
    MOVLW .253
    MOVWF BAJO
    MOVLW .245
    MOVWF ALTO
    CALL CONVERSION_COMPU
    GOTO LOOP
AUTOMATIC_HIGH:
    MOVLW .245
    MOVWF BAJO
    MOVLW .253
    MOVWF ALTO
    CALL CONVERSION_COMPU
    GOTO LOOP

;*********************************************************************************
; SUBRUTINAS DE INTERRUPCION
;*********************************************************************************
BANDERA_TX: ; MANDO DATOS --> UN NUMERO Y UN ENTER
    MOVLW   .1
    SUBWF   TX_FLAG, W
    BTFSC   STATUS, Z
    GOTO    ENTER
    DIGITO1:
MOVFW LAST_USER
CALL CONVERSIONES
MOVWF TXREG
INCF TX_FLAG
GOTO LIMPIEZA
    ENTER:
MOVLW .10
MOVWF TXREG
CLRF TX_FLAG
    LIMPIEZA:
BANKSEL TRISA
BCF PIE1, TXIE
BANKSEL PORTA
    RETURN

BANDERA_ADC:
    BTFSC   FLAG_ADC, 0
    GOTO    ADCY
    ADCX:
MOVFW   ADRESH     ; MANDA LA CODIFICACION DIGITAL DE MI SEÑAL ANALOGICA AL PUERTO B
MOVWF   VAR_ADCX
CALL CONFIGURACION_ADCY
BCF PIR1, ADIF
BSF ADCON0, 1
BSF FLAG_ADC, 0
    RETURN
    ADCY:     ; FUNGIONA CON EL SERVO DERECHO
MOVFW   ADRESH     ; MANDA LA CODIFICACION DIGITAL DE MI SEÑAL ANALOGICA AL PUERTO B
MOVWF   VAR_ADCY
CALL CONFIGURACION_ADCX
BCF PIR1, ADIF
BSF ADCON0, 1
BCF FLAG_ADC, 0
    RETURN

BANDERA_TIMER0:     ; PERMITE ABRIR O CERRAR LA GARRA CARGANDOLE VALORES DE CONTROL ENTRE 0.5ms Y 2.5ms
    BTFSC   TOGGLE, 0
    GOTO    LOW_OPEN
    HIGH_OPEN:
MOVFW   BAJO
MOVWF   TMR0
BSF     PORTC, 0
BSF     TOGGLE, 0
```

```
BCF     INTCON, TOIF
    RETURN
    LOW_OPEN:
MOVFW   ALTO
MOVWF   TMRO
BCF     PORTC, 0
BCF     TOGGLE, 0
BCF     INTCON, TOIF
    RETURN

BANDERA_RX:
    INCF    CUENTARX,1     ; CORRO TODOS LOS DATOS RECIBIDOS UN BYTE
    MOVFW   RXB8
    MOVWF   RXB9

    MOVFW   RXB7
    MOVWF   RXB8

    MOVFW   RXB6
    MOVWF   RXB7

    MOVFW   RXB5
    MOVWF   RXB6

    MOVFW   RXB4
    MOVWF   RXB5

    MOVFW   RXB3
    MOVWF   RXB4

    MOVFW   RXB2
    MOVWF   RXB3

    MOVFW   RXB1
    MOVWF   RXB2

    MOVFW   RXB0
    MOVWF   RXB1

    MOVFW   RCREG
    MOVWF   RXB0

    XORLW   .10 ; REVISO SI EL ULTIMO BYTE LEIDO ES UN ENTER PARA INDICAR QUE YA LEI TODOS LOS DATOS
    BTFSC   STATUS,Z
    GOTO    VERIFICACION
    RETURN

    VERIFICACION:
MOVLW   .10 ; VERIFICO QUE SE HAYAN HECHO 10 LECTURAS
SUBWF   CUENTARX,W
BTFSS   STATUS,Z
GOTO    ERRONEO

MOVLW .44 ; VERIFICO QUE LAS COMAS ESTEN EN DONDE DEBERIAN
SUBWF RXB2,W
BTFSS STATUS,Z
GOTO ERRONEO

MOVLW .44
SUBWF RXB4,W
BTFSS STATUS,Z
GOTO ERRONEO

MOVLW .44
SUBWF RXB6,W
BTFSS STATUS,Z
GOTO ERRONEO

MOVLW .44
SUBWF RXB8,W
BTFSS STATUS,Z
GOTO ERRONEO

MOVLW   .48 ; GUARDO CADA DATO EN DONDE CORRESPONDE
SUBWF   RXB1,W ; LES RESTO 48 PARA PASAR DE ASCII A DECIMAL
MOVWF   USUARIO

MOVLW   .48
SUBWF   RXB3,W
MOVWF   SERVO_EJE1

MOVLW   .48
SUBWF   RXB5,W
MOVWF   SERVO_EJE2

MOVLW   .48
SUBWF   RXB7,W
MOVWF   SERVO_GARRA

MOVLW   .48
SUBWF   RXB9,W
MOVWF   SERVO_FUN
CLRF CUENTARX; REINICIO LA CUENTA PARA VOLVER A EMPEZAR
RETURN

    ERRONEO:
CLRF    CUENTARX
RETURN

;*********************************************************************************
; RUTINA EEPROM
;*********************************************************************************
EEPROM_ESCRITURA:
    MOVFW   USUARIO     ; ESTA RESTA FUNCIONA COMO UN ANTIREBOTE
    SUBWF   USER_FLAG, W    ; --> SOLO GUARDA EN LA EEPROM CUANDO EL USUARIO CAMBIA
    BTFSS   STATUS, Z
    GOTO    ACCION
RETURN
    ACCION:
    MOVFW   USUARIO
    MOVWF   USER_FLAG
    BANKSEL EEADR
    MOVLW   .0
    MOVWF   EEADR
    BANKSEL PORTA
    MOVFW   USUARIO
```

```
    BANKSEL EEDAT
    MOVWF   EEDAT
    BANKSEL EECON1
    BCF     EECON1, EEPGD
    BSF     EECON1, WREN

    BCF     INTCON, GIE
    MOVLW   0x55
    MOVWF   EECON2
    MOVLW   0xAA
    MOVWF   EECON2
    BSF     EECON1, WR
    BSF     INTCON, GIE

    BCF     EECON1, WREN
    BANKSEL PORTA
    CALL    EEPROM_LECTURA
RETURN

;*********************************************************************************
; RUTINA DE LECTURA DE LA EEPROM
;*********************************************************************************
EEPROM_LECTURA:     ; LEE EL ULTIMO USUARIO USADO
    BCF     INTCON, GIE
    MOVLW   .0
    BANKSEL EEADR
    MOVWF   EEADR
    BANKSEL EECON1
    BCF     EECON1, EEPGD
    BSF     EECON1, RD
    BANKSEL EEDATA
    MOVFW   EEDATA
    BANKSEL PORTA
    MOVWF   LAST_USER
    MOVWF   PORTD     ; COLOCA EL ULTIMO USUARIO EN EL PUERTO D
    BSF     INTCON, GIE
RETURN

;*********************************************************************************
; RUTINA DE SELECCIÓN DE MODOS DE FUNCIONAMIENTO
;*********************************************************************************
MODO_FUNCIONAMIENTO:
    MOVFW   FLAG_ANTIREBOTE
    SUBLW   .1 ; REVISA QUE SÍ HAYA PASADO POR EL ANTIREBOTE
    BTFSS   STATUS, Z ; SI NO PASÓ POR EL ANTIREBOTE, SIGNIFICA QUE NO SE PRESIONÓ EL BOTÓN Y NO EJECUTA LA INSTRUCCIÓN
RETURN
    CLRF    FLAG_ANTIREBOTE ; LIMPIA LA BANDERA PARA QUE SE PUEDA VOLVER A PRESIONAR EL BOTÓN SIN REBOTES
    MOVFW   MODO
    SUBLW   .1
    BTFSC   STATUS, Z
    GOTO    REINICIOA ; SIRVE PARA QUE NO SE PASE DEL MODO 8
    CONTEO:
MOVLW .253
MOVWF BAJO
MOVLW .245
MOVWF ALTO
     INCF    MODO
    RETURN
    REINICIOA:
MOVLW .245
MOVWF BAJO
MOVLW .253
MOVWF ALTO
CLRF MODO
    RETURN

;*********************************************************************************
; RUTINA DE ANTIREBOTE
;*********************************************************************************
ANTIR:
    BSF   FLAG_ANTIREBOTE, 0 ; YA QUE SE USAN PULL UPS, ESTE MASKING ME PERMITE VER QUÉ VALOR SE COLOCÓ EN CERO, ES DECIR, SE PRESIONÓ
RETURN

;*********************************************************************************
; RUTINA DE CONVERSION COMPU
;*********************************************************************************
CONVERSION_COMPU: ; EN LA COMPU MANDO DATOS DE 0 A 8, POR LO QUE NECESITO MULTIPLICAR POR 16 Y SUMAR 32 PARA EL MAPPEO
    SWAPF   SERVO_EJE1, 0 ; USAR UN SWAPF ES COMO MULTIPLICAR POR 16, PUESTO QUE CORRE LOS BITS 4 VECES
    ADDLW   .32
    MOVWF   CCPR1L
    SWAPF   SERVO_EJE2, 0
    ADDLW   .32
    MOVWF   CCPR2L
RETURN

;*********************************************************************************
; RUTINA DE CONVERSION ADC
;*********************************************************************************
MAPPEO: ; CON EL ADC SE MANDAN DATOS DE 0 A 255, POR LO QUE NECESITO DIVIDIR DENTRO DE 2 Y SUMAR 32 PARA EL MAPPEO
    RRF     VAR_ADCX, 0
    ANDLW   b'01111111'; USAR ROTATE RIGHT SEGUIDO DE UN AND ES COMO DIVIDIR DENTRO DE 2
    ADDLW   .32
    MOVWF   CCPR1L
    RRF     VAR_ADCY, 0
    ANDLW   b'01111111'
    ADDLW   .32
    MOVWF   CCPR2L
RETURN

;*********************************************************************************
; CONFIGURACIONES
;*********************************************************************************
CONFIGURACION_BASE:
    BANKSEL PORTA
    CLRF    PORTA ; LIMPIA LOS PUERTOS PARA EVITAR QUE TENGAN CUALQUIER VALOR INICIAL DISTINTO DE 0
    CLRF    PORTB
    CLRF    PORTC
    CLRF    PORTD
    CLRF    PORTE

    BANKSEL ANSEL
    CLRF    ANSEL
    BSF     ANSEL, 0 ; POT EN X
    BSF     ANSEL, 5 ; POT EN Y
    CLRF    ANSELH ; BORRA EL CONTROL DE ENTRADAS ANALÓGICAS
```

```
    BANKSEL TRISA
    MOVLW   b'00100001'; POT Y TRANSISTORES
    MOVWF   TRISA

    MOVLW   b'10000000'; PUSHES EN EL PUERTO B PARA INCLUIR LOS PULL UPS POR SOFTWARE - TIENE 4 SALIDAS PUES SE CONECTARON AHÍ LOS TRANSISTORES (NPN 3904)
    MOVWF   TRISB
    MOVWF   WPUB ; PARA PULL UPS
    BCF     OPTION_REG, 7

    CLRF    TRISC ; SIN USAR
    CLRF    TRISD ; PARA DISPLAY
    CLRF    TRISE

    BANKSEL PORTA
    CLRF    FLAG_ANTIREBOTE
    CLRF    FLAG_ADC
    CLRF    TX_FLAG
    CLRF    VAR_ADCX
    CLRF    VAR_ADCY
    CLRF    MODO
    CLRF    ALTO
    CLRF    BAJO
    CLRF    TOGGLE
    CLRF    SERVO_GARRA
    CLRF    SERVO_EJE1
    CLRF    SERVO_EJE2
    CLRF    SERVO_FUN
    CLRF    CUENTARX
    CLRF    RXB0
    CLRF    RXB1
    CLRF    RXB2
    CLRF    RXB3
    CLRF    RXB4
    CLRF    RXB5
    CLRF    RXB6
    CLRF    RXB7
    CLRF    RXB8
    CLRF    RXB9
    CLRF    USUARIO
    CLRF    LAST_USER
    CLRF    USER_FLAG
    CLRF    DIVISION
RETURN

CONFIGURACION_PWM:
    SERVO1:
    BANKSEL CCP1CON
    BCF     CCP1CON, 7
    BCF     CCP1CON, 6 ; 6 Y 7 PARA SINGLE OUTPUT
    BCF     CCP1CON, 5
    BCF     CCP1CON, 4 ; BITS MENOS SIGNIFICATIVOS PARA EL ANCHO DE PULSO
    BSF     CCP1CON, 3
    BSF     CCP1CON, 2
    BCF     CCP1CON, 1
    BCF     CCP1CON, 0
    SERVO2:
    BANKSEL CCP2CON
    BCF     CCP2CON, 5
    BCF     CCP2CON, 4 ; BITS MENOS SIGNIFICATIVOS PARA EL ANCHO DE PULSO
    BSF     CCP2CON, 3
    BSF     CCP2CON, 2
    BSF     CCP2CON, 1
    BSF     CCP2CON, 0
RETURN

CONFIGURACION_TIMERO:
    BANKSEL TRISA
    CLRWDT ; CONFIGURACIÓN PARA EL FUNCIONAMIENTO DEL TIMERO
    MOVLW   b'01010111'; PRESCALER DE 1:256 PARA PODER GENERAR INTERRUPCIONES DE 0.5ms
    MOVWF   OPTION_REG
    BANKSEL PORTA
RETURN

CONFIGURACION_TIMER2:
    BANKSEL PORTA
    MOVLW   b'11111111'; PRESCALER Y POSTSCALER DE 16 CADA UNO Y TIMER 2 ACTIVADO
    MOVWF   T2CON
RETURN

CONFIGURACION_INTERRUPCION:
    BANKSEL TRISA
    BSF     PIE1, ADIE
    BSF     PIE1, RCIE ; HABILITA INTERRUPCION DE RECEPCION SERIAL CON RX
    BSF     PIE1, TXIE ; HABILITA INTERRUPCION DE RECEPCION SERIAL CON TX
    BSF     INTCON, PEIE ; INTERRUPCIONES PERIFÉRICAS -RC-
    BSF     INTCON, TOIE

    MOVLW   .187 ; TECHO PARA TIMER2 - PARA QUE EL PWM FUNCIONE CON 3ms
    MOVWF   PR2 ; PARA PULSO DE 0° --> CCPR1L = 0x20 ^ CCP1CON<5:4> = b'00
; PARA PULSO DE 180° --> CCP1L = 0x9D ^ CCP1CON<5:4> = b'00
; FUNCION PARA CONVERSION DE DATOS ADC PARA METER EN EL CCPR1L:
; CCP1RL = 32 + ADC/2

    BANKSEL PORTA
    BSF     INTCON, GIE ; HABILITA LAS INTERRUPCIONES
    BCF     INTCON, TOIF ; PARA ASEGURARSE DE QUE NO TENGA OVERFLOW AL INICIO
RETURN

CONFIGURACION_TX_9600:
    BANKSEL TRISA
    BCF     TXSTA, TX9
    BCF     TXSTA, SYNC     ; PARA LOGRAR UN BAUD DE 9600 CON UN FOSC DE 4MHz
    BSF     TXSTA, BRGH     ; PARA LOGRAR UN BAUD DE 9600 CON UN FOSC DE 4MHz

    BANKSEL ANSEL
    BCF     BAUDCTL, BRG16  ; PARA LOGRAR UN BAUD DE 9600 CON UN FOSC DE 4MHz

    BANKSEL TRISA
    MOVLW   .25
    MOVWF   SPBRG     ; PARA LOGRAR UN BAUD DE 9600 CON UN FOSC DE 4MHz
    CLRF    SPBRGH     ; PARA LOGRAR UN BAUD DE 9600 CON UN FOSC DE 4MHz
    BSF     TXSTA, TXEN
    BANKSEL PORTA
RETURN

CONFIGURACION_RX:
```

```
    BANKSEL PORTA
    BSF     RCSTA, SPEN
    BCF     RCSTA, RX9
    BSF     RCSTA, CREN
RETURN

CONFIGURACION_ADC:
    BANKSEL ADCON1
    CLRF    ADCON1 ; VDD Y VSS COMO REFERENCIA / JUSTIFICADO A LA IZQUIERDA

    BANKSEL ADCON0
    BSF     ADCON0, 0
    BSF     ADCON0, 1
    BCF     ADCON0, 2
    BCF     ADCON0, 3
    BCF     ADCON0, 5
    BCF     ADCON0, 6
    BSF     ADCON0, 7
RETURN

CONFIGURACION_ADCX:
    BANKSEL ADCON0
    BCF     ADCON0, 4
RETURN

CONFIGURACION_ADCY:
    BANKSEL ADCON0
    BSF     ADCON0, 4
RETURN
;*******************************************************************************

    END
```

5.2. Python

5.2.1. PyQt Designer

```
from PyQt5 import QtCore, QtGui, QtWidgets


class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(800, 600)
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.pushButton = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton.setGeometry(QtCore.QRect(40, 140, 101, 28))
        self.pushButton.setObjectName("pushButton")
        self.pushButton_2 = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton_2.setGeometry(QtCore.QRect(40, 170, 101, 28))
        self.pushButton_2.setObjectName("pushButton_2")
        self.label_presentacion = QtWidgets.QLabel(self.centralwidget)
        self.label_presentacion.setGeometry(QtCore.QRect(110, 30, 600, 60))
        self.label_presentacion.setAlignment(QtCore.Qt.AlignCenter)
        font = QtGui.QFont()
        font.setPointSize(14)
        self.label_presentacion.setFont(font)
        self.label_presentacion.setObjectName("label_presentacion")
        self.label = QtWidgets.QLabel(self.centralwidget)
        self.label.setGeometry(QtCore.QRect(10, 99, 161, 51))
        self.label.setAlignment(QtCore.Qt.AlignCenter)
        self.label.setObjectName("label")
        self.label_2 = QtWidgets.QLabel(self.centralwidget)
        self.label_2.setGeometry(QtCore.QRect(400, 110, 71, 16))
        self.label_2.setObjectName("label_2")
        self.label_3 = QtWidgets.QLabel(self.centralwidget)
        self.label_3.setGeometry(QtCore.QRect(230, 150, 55, 16))
        self.label_3.setObjectName("label_3")
        self.pushButton_3 = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton_3.setGeometry(QtCore.QRect(340, 140, 93, 28))
        self.pushButton_3.setObjectName("pushButton_3")
        self.pushButton_4 = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton_4.setGeometry(QtCore.QRect(440, 140, 93, 28))
        self.pushButton_4.setObjectName("pushButton_4")
        self.label_4 = QtWidgets.QLabel(self.centralwidget)
        self.label_4.setGeometry(QtCore.QRect(190, 190, 121, 41))
        self.label_4.setObjectName("label_4")
        self.horizontalSlider = QtWidgets.QSlider(self.centralwidget)
        self.horizontalSlider.setGeometry(QtCore.QRect(350, 200, 160, 22))
        self.horizontalSlider.setMaximum(8)
        self.horizontalSlider.setValue(4)
        self.horizontalSlider.setOrientation(QtCore.Qt.Horizontal)
        self.horizontalSlider.setObjectName("horizontalSlider")
        self.label_5 = QtWidgets.QLabel(self.centralwidget)
        self.label_5.setGeometry(QtCore.QRect(190, 270, 131, 41))
        self.label_5.setObjectName("label_5")
        self.verticalSlider = QtWidgets.QSlider(self.centralwidget)
        self.verticalSlider.setGeometry(QtCore.QRect(420, 230, 22, 160))
        self.verticalSlider.setMaximum(8)
        self.verticalSlider.setValue(4)
        self.verticalSlider.setOrientation(QtCore.Qt.Vertical)
        self.verticalSlider.setObjectName("verticalSlider")
        self.label_6 = QtWidgets.QLabel(self.centralwidget)
        self.label_6.setGeometry(QtCore.QRect(580, 100, 160, 41))
        self.label_6.setObjectName("label_6")
        self.pushButton_5 = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton_5.setGeometry(QtCore.QRect(600, 140, 93, 28))
        self.pushButton_5.setObjectName("pushButton_5")
        self.pushButton_6 = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton_6.setGeometry(QtCore.QRect(600, 170, 93, 28))
        self.pushButton_6.setObjectName("pushButton_6")
        self.label_7 = QtWidgets.QLabel(self.centralwidget)
        self.label_7.setGeometry(QtCore.QRect(580, 200, 160, 41))
        self.label_7.setObjectName("label_7")
        self.label_8 = QtWidgets.QLabel(self.centralwidget)
        self.label_8.setGeometry(QtCore.QRect(10, 500, 200, 100))
        self.label_8.setObjectName("label_8")
        MainWindow.setCentralWidget(self.centralwidget)
        self.menubar = QtWidgets.QMenuBar(MainWindow)
        self.menubar.setGeometry(QtCore.QRect(0, 0, 800, 26))
        self.menubar.setObjectName("menubar")
        MainWindow.setMenuBar(self.menubar)
```

```
        self.statusbar = QtWidgets.QStatusBar(MainWindow)
        self.statusbar.setObjectName("statusbar")
        MainWindow.setStatusBar(self.statusbar)

        self.retranslateUi(MainWindow)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)

    def retranslateUi(self, MainWindow):
        _translate = QtCore.QCoreApplication.translate
        MainWindow.setWindowTitle(_translate("MainWindow", "PROYECTO 2"))
        self.label_presentacion.setText(_translate("MainWindow", "BIENVENIDO A LA PLATAFORMA DE LA GARRA MECÁNICA\n"))
        self.pushButton.setText(_translate("MainWindow", "MANUAL"))
        self.pushButton_2.setText(_translate("MainWindow", "COMPUTADORA"))
        self.label.setText(_translate("MainWindow", "SELECCIÓN DE CONTROL"))
        self.label_2.setText(_translate("MainWindow", "CONTROLES"))
        self.label_3.setText(_translate("MainWindow", "GARRA"))
        self.pushButton_3.setText(_translate("MainWindow", "ABRIR"))
        self.pushButton_4.setText(_translate("MainWindow", "CERRAR"))
        self.label_4.setText(_translate("MainWindow", "PRIMER EJE DE GIRO"))
        self.label_5.setText(_translate("MainWindow", "SEGUNDO EJE DE GIRO"))
        self.label_6.setText(_translate("MainWindow", "SELECCIONAR USUARIO"))
        self.pushButton_5.setText(_translate("MainWindow", "USUARIO 1"))
        self.pushButton_6.setText(_translate("MainWindow", "USUARIO 2"))
        self.label_7.setText(_translate("MainWindow", "ÚLTIMO USUARIO ACTIVO:\n"))
        self.label_8.setText(_translate("MainWindow", "CREADO POR:\nRICARDO PELLECER ORELLANA\nCARNÉ 19072"))


if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    MainWindow = QtWidgets.QMainWindow()
    ui = Ui_MainWindow()
    ui.setupUi(MainWindow)
    MainWindow.show()
    sys.exit(app.exec_())
```

5.2.2. Interfaz de Control

```
from Design_Union import *
from PyQt5 import QtWidgets
import threading
import serial
import time
import sys
usuario = 0
toggle = 0
garra = 0
value_horizontal = 0
value_vertical = 0
class SKETCH (QtWidgets.QMainWindow, Ui_MainWindow):
    def __init__ (self):
        super().__init__()
        self.setupUi(self)
        self.pushButton.clicked.connect(self.presionado1)
        self.pushButton_2.clicked.connect(self.presionado2)
        self.pushButton_3.clicked.connect(self.cerrar)
        self.pushButton_4.clicked.connect(self.abrir)
        self.pushButton_5.clicked.connect(self.us1)
        self.pushButton_6.clicked.connect(self.us2)
        self.horizontalSlider.valueChanged.connect(self.obtener_valor_horizontal)
        self.verticalSlider.valueChanged.connect(self.obtener_valor_vertical)
        instruccion = threading.Thread(daemon=True,target=controles)
        instruccion.start()

    def presionado1(self):
        global toggle
        toggle = 0

    def presionado2(self):
        global toggle
        toggle = 1

    def abrir(self):
        global garra
        garra = 0

    def cerrar(self):
        global garra
        garra = 1

    def obtener_valor_horizontal(self):
        global value_horizontal
        value_horizontal = self.horizontalSlider.value()

    def obtener_valor_vertical(self):
        global value_vertical
        value_vertical = self.verticalSlider.value()

    def us1(self):
        global usuario
        usuario = 0

    def us2(self):
        global usuario
        usuario = 1

    def last(self, ultimo_usuario):
        global usuario
        self.label_7.setText("ÚLTIMO USUARIO ACTIVO:\n          USUARIO " + str(int(ultimo_usuario)+1))

    def actualizacion(self):
        self.update()

def controles():
    global ventanamain, toggle, garra, value_vertical, value_horizontal, usuario
    ser = serial.Serial(port='COM4',baudrate=9600, parity=serial.PARITY_NONE, stopbits=serial.STOPBITS_ONE,bytesize=serial.EIGHTBITS, timeout=0)
    while (1) :
        ser.flushOutput()
        if (toggle == 1):
            # Mando 9 para control manual
            ser.write(bytes.fromhex('39'))
            #print(ser.read())
            # Coma
            ser.write(bytes.fromhex('2C'))
```

```
            #print(ser.read())
            # Posicion primer servo
            if (garra == 1):
                # 00 para posición mínima
                ser.write(bytes.fromhex('30'))
                #print(ser.read())
            else:
                # 9 para posición máxima
                ser.write(bytes.fromhex('39'))
                #print(ser.read())
            # Coma
            ser.write(bytes.fromhex('2C'))
            #print(ser.read())
            # Posicion segundo servo
            ser.write(bytes.fromhex(hex(ord(str(value_vertical)))[2:]))
            #print(ser.read())
            # Coma
            ser.write(bytes.fromhex('2C'))
            #print(ser.read())
            # Posicion tercer servo
            ser.write(bytes.fromhex(hex(ord(str(value_horizontal)))[2:]))
            #print(ser.read())
            # Coma
            ser.write(bytes.fromhex('2C'))
            #print(ser.read())
            # Usuario actual
            ser.write(bytes.fromhex(hex(ord(str(usuario)))[2:]))
            #print(ser.read())
            # Enter
            ser.write(bytes.fromhex('0A'))
            #print(ser.read())
            try:
                ser.flushInput()
                time.sleep(.3)
                ser.readline()
                final = str(ser.readline())
                ventanamain.last(final[2])
                print(str(int(final[2])+1))
            except:
                pass
            ventanamain.actualizacion()

        else:
            try :
                # Mando 00 para control manual
                ser.write(bytes.fromhex('30'))
                #print(ser.read())
                # Coma
                ser.write(bytes.fromhex('2C'))
                #print(ser.read())
                # Posicion primer servo
                ser.write(bytes.fromhex('30'))
                #print(ser.read())
                # Coma
                ser.write(bytes.fromhex('2C'))
                #print(ser.read())
                # Posicion segundo servo
                ser.write(bytes.fromhex('30'))
                #print(ser.read())
                # Coma
                ser.write(bytes.fromhex('2C'))
                #print(ser.read())
                # Posicion tercer servo
                ser.write(bytes.fromhex('30'))
                #print(ser.read())
                # Coma
                ser.write(bytes.fromhex('2C'))
                #print(ser.read())
                # Usuario actual
                ser.write(bytes.fromhex(hex(ord(str(usuario)))[2:]))
                #print(ser.read())
                # Enter
                ser.write(bytes.fromhex('0A'))
                #print(ser.read())
                try:
                    ser.flushInput()
                    time.sleep(.3)
                    ser.readline()
                    final = str(ser.readline())
                    #final[0] = int(final[0][2:], 16)
                    #final = ['adios']
                    ventanamain.last(final[2])
                    print(str(int(final[2])+1))
                except:
                    pass
                ventanamain.actualizacion()
            except:
                pass

aplication = QtWidgets.QApplication([])
ventanamain=SKETCH()
ventanamain.show()
aplication.exec_()
```

La explicación del funcionamiento se encuentra más a detalle en el siguiente video.