Desarrollo de un sistema USB® para el aprendizaje y la implementación en dispositivos programables PIC18®

Presentado por

Gabriela Alejandra Pardo Ortega

c.c. 1070984502

código: 58832

Director

Robin Alfonzo Blanco Cañon

Codirector

Andrés Mauricio Cifuentes

Universidad ECCI

Bogotá-Colombia

Junio – 2022

Índice

1. Objetivos 4
1.1 Objetivo General 4
1.2 Objetivos Específicos 4
2. Justificación 5
3. Introducción 6
4. Marco Teórico 7
4.1 Microcontrolador 7
4.1.1 Microcontrolador PIC18F4550 8
4.2 Compilador 8
4.2.1 MPLAB XC8® 9
4.3 Comunicación USB® 9
4.3.1 Características 9
4.3.2 Estructura 9
4.3.3 Funcionamiento 10
4.3.4 Historia 12
4.3.5 Versiones 12
4.3.6 Conectores 13
4.3.7 Clases 15
4.4 Software 17
4.4.1 Windows 17
4.4.2 Raspbian 17
4.4.3 MPLAB® X IDE 17
4.4.4 Proteus 17
4.4.5 Matlab 18
4.4.6 Python 18
4.5 Dispositivos y componentes electrónicos 18
4.5.1 Teclado matricial 18
4.5.2 Servomotor 18
4.5.3 Módulo Bluetooth HC-05 19

4.5.4 RTC (Real Time Clock) 19
4.5.5 LCD (Liquid Crystal Display) 20
4.5.6 DHT11 20
4.5.7 Relé de estado solido 21
4.5.8 Dimmer digital 21
4.5.9 Raspberry Pi 22
4.6 PWM (Pulse Width Modulation) 22
4.6.1 PWM en Corriente alterna 23
4.7 Lógica difusa 23
5. Metodología 24
5.1 Prueba y adaptación de programa base 24
5.2 Ejemplos por Módulo 26
5.2.1TIMER0 27
5.2.2 Módulo de entradas y salidas (I/O) 28
5.2.3 Módulo ADC (Conversor Análogo-Digital) 30
5.2.4 Módulo de Referencia 32
5.2.5 Módulo CMP (Comparador) 34
5.2.6 Módulo CCP 35
5.2.7 Módulo EUSART 38
5.2.8 Módulo MSSP 40
5.2.9 Módulo EEPROM 42
5.2.10 Programa de comunicación USB-CDC® por interrupción 44
5.3 Diseño de tarjeta PCB 46
5.4 Ejercicios de aplicación 46
5.3.1 Componentes electrónicos 46
5.3.2 Ejercicio de Instrumentación 51
5.3.3 Ejercicio de Control ON-OFF 56
5.3.4 Ejercicio de Control Difuso 62
6. Resultados 67
7. Conclusiones 71
8. Alcances 71
9. Referencias 72
10. Anexos 76

1. Objetivos

1.1 Objetivo General

Implementar un sistema de aprendizaje que permita comprender las características y el funcionamiento de la comunicación USB® haciendo uso del microcontrolador PIC18F4550 para mejorar el alcance profesional en las diferentes ramas de la ingeniería teniendo en cuenta sus diversas aplicaciones y las ventajas respecto a otros protocolos de comunicación similares.

1.2 Objetivos Específicos

- Explorar el protocolo de comunicación USB® mediante la programación del microcontrolador PIC18F4550 comprendiendo el funcionamiento del mismo y de las funciones utilizadas.
- Generar un programa base con el compilador XC8® en lenguaje C que permita un primer acercamiento al uso de este protocolo y a su vez permita la interacción con otros módulos del microcontrolador.
- Diseñar una tarjeta PCB que permita la implementación de ejemplos que utilicen la comunicación USB® del microcontrolador
- Realizar prácticas que muestren el uso de este protocolo en diferentes áreas relacionadas con la ingeniería y la interacción con diferentes sistemas operativos como lo son Windows y Linux.

2. Justificación

Debido al alto impacto que ha tenido el protocolo de comunicación USB® en la industria de la tecnología en los últimos años, es de gran importancia para los futuros ingenieros de carreras relacionadas a dicha temática tener un acercamiento a la implementación de este protocolo y de esta manera, ampliar su conocimiento del campo y obtener más oportunidades de empleo. Sin embargo, se presentan una serie de problemas al momento de la enseñanza de este.

Uno de los problemas más graves que se presenta al momento de querer llevar a cabo la implementación de este protocolo es la falta de información clara que se encuentra sobre este tema, ya que gran parte esta no es útil al momento de iniciar el aprendizaje de este protocolo ya sea por su complejidad, su ambigüedad, el software que utiliza o la falta de ejemplos para su uso.

Por otro lado, también se encuentra que gran parte de los dispositivos que permiten tener un acercamiento a este protocolo de comunicación tienen un costo bastante elevado.

Por lo tanto, se requiere de la recopilación de información para asegurar que más estudiantes tengan la posibilidad de un mayor acercamiento a estos temas y su implementación en un dispositivo que tenga un costo asequible para ellos.

3. Introducción

El protocolo de comunicación USB® (Universal Serial Bus) ha tenido una gran acogida por la sociedad desde su origen en la década de los 90, siendo este diseñado por un grupo de empresas importantes en el mundo de la tecnología como Apple, IBM, Intel, entre otras, con el fin de facilitar la conexión de periféricos con dispositivos como el computador [1]. Este protocolo llega a alcanzar altas velocidades, según la versión utilizada por el dispositivo puede obtener velocidades de entre 1,5Mbits/s y 40Gbits/s [2].

Entre sus características más importantes se encuentra la facilidad de conexión de hasta 127 periféricos, un conector universal que se utiliza en sus diferentes versiones además de su característica de "*Plug and Play*" que permite la reconexión de los dispositivos sin necesidad de apagar o reiniciar el dispositivo que controla la comunicación [3].

Entre sus versiones se encuentra la versión USB1.0® lanzada al mercado en 1996 que se utiliza especialmente en teclados y ratones ya que cuenta con velocidad idónea para el funcionamiento de estos [4], la versión USB1.1® que fue lanzada en 1998 permite una velocidad de comunicación mayor y corrige algunas de las fallas presentadas en su versión anterior sus usos en la actualidad son similares a los de USB1.0®, esta versión de igual manera fue sustituida por las versiones USB2.0® y USB3.0® que soportan velocidades de hasta 480Mbits/s y 15Gbits/s respectivamente [5]. La versión más actual es USB4.0® que alcanza velocidades de transmisión de hasta 40Gbits/s [6].

Existen varios dispositivos programables que permiten la comunicación USB® como lo son algunos microcontroladores, tarjetas STM32 núcleo, Arduino, entre otros. Sin embargo, estos dispositivos tienen costos elevados por lo cual no son la mejor opción para la introducción hacia el aprendizaje de este tema. Siendo así, una buena alternativa para empezar a conocer la programación de este protocolo, son los microcontroladores PIC® que tienen un costo asequible para los estudiantes. Estos tienen algunas características importantes como una alta velocidad de respuesta (normalmente usan una frecuencia de, 20MHz), su reducido tamaño, su versatilidad y compatibilidad con diversos Software que permiten la programación del mismo y la simulación de su funcionamiento [7].

Por lo tanto, en este proyecto se propone la exploración e implementación de la comunicación USB® utilizando el microcontrolador PIC18F4550 y el compilador XC8® de manera tal que se permita la comprensión de las librerías utilizadas y se experimente su fusión con otras funciones que tiene este microcontrolador además de recompilar y explicar los conceptos básicos de la comunicación USB®, teniendo en cuenta características fundamentales como su funcionamiento, velocidad, conexión, practicidad y aplicaciones en diferentes áreas de la ingeniería.

4. Marco Teórico

4.1 Microcontrolador

Es un dispositivo programable con la capacidad de realizar diversas tareas a alta velocidad, está compuesto por una CPU (Central Processing Unit), una memoria RAM y ROM donde almacena la información de su programación y la información recolectada por el mismo, además de puertos de entradas y salidas (I/O) que permiten la comunicación del microcontrolador con su entorno [8]. Véase Figura 1

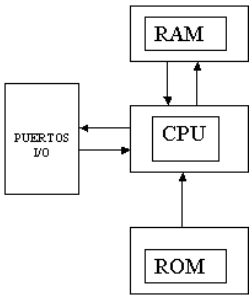


Figura 1. Estructura del microcontrolador. Tomado de https://www.ingmecafenix.com/

Este dispositivo puede tener una arquitectura Van Neumann en la cual se utiliza solo un bus de datos para transportar la información a través de este y cuenta con una única memoria de almacenamiento, o la arquitectura Harvard donde se presentan dos memorias, una para el almacenamiento de datos y la otra para el almacenamiento del programa lo que permite una mayor velocidad [9]. Véase Figura 2

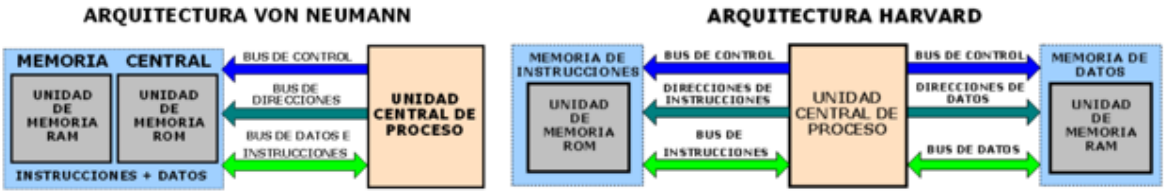


Figura 2. Arquitecturas de los microcontroladores. Tomado dehttps://compilandoconocimiento.com/

Los microcontroladores tienen una gran variedad de aplicaciones, se destacan especialmente en el campo de la robótica y electrónica donde uno de sus usos más comunes es en mini robots y casa inteligentes además de otros usos como vehículos automatizados y ciberseguridad [10].

Entre las empresas más destacadas de microcontroladores se encuentran Atmel Corporation, Freescale semiconductor, Texas Instruments, ZiLOG Inc. y Microchip Inc. las cuales fabrican microcontroladores con diferentes características y utilidades [11]. Entre estos se encuentran los microcontroladores PIC® (Fabricados por Microchip Inc.) los cuales cuentan con una serie de gamas diferentes que permiten diferentes funciones en el microcontrolador [12]:

- Gama Baja: Son PIC® de bajo consumo con recursos limitados, normalmente tienen entre 18 y 28 pines y los más comunes son la familia PIC16/17.

- Gama Media: Cuentan con entre 18 y 68 pines, a diferencia de la gama baja cuentan con módulos como ADC y TIMER además de algunas interrupciones. Entre estos se encuentra el PIC16X84 y sus variantes.
- Gama Alta: Cuentan con más funciones que las gamas anteriores entre las cuales se encuentra la posibilidad de comunicación en serie y paralelo, mayor velocidad y memoria, entre otras.

4.1.1 Microcontrolador PIC18F4550

Es un microcontrolador fabricado por Microchip®, funciona a 8 bits, tiene una arquitectura tipo Harvard y pertenece a la gama medio-alta de esta empresa. Además, tiene una muy buena capacidad de procesamiento y su consumo de energía no es muy elevado. Cuenta con 40 pines y seis puertos que permiten realizar diferentes operaciones [13]. Véase Figura 3

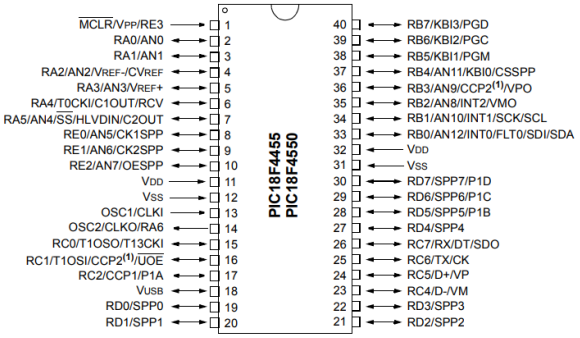


Figura 3. PIC18F4550. Tomada de PIC18F2455/2550/4455/4550 Data Sheet

Este microcontrolador cuenta con otras características como lo son una frecuencia de oscilación entre 31kHz y 8MHz haciendo uso de su oscilador interno o de hasta 48MHz haciendo uso de un oscilador externo, además permite el uso de PLL (Phase Lock Loop) para la implementación de frecuencias mayores.

Entre sus funciones se encuentran los módulos de comunicación USB®, cuatro Timer, tres interrupciones externas, dos módulos CCP (Capture/Compare/PWM), el módulo USART y MSSP que implementa diferentes tipos de comunicación serial con otros dispositivos, un conversor análogo digital, entre otras funciones que hacen de este un dispositivo de gran utilidad [14].

4.2 Compilador

Un compilador es un traductor que transforma un programa de un lenguaje de programación a otro, usualmente un lenguaje de maquina o un lenguaje intermedio. Este ejecuta ciertas etapas para cumplir con su función, iniciando con una revisión del programa fuente verificando las diferentes sintaxis que deben ser cumplidas por este realizando un análisis léxico y semántico. Eventualmente realiza una etapa de síntesis donde se genera y optimiza el código en el lenguaje final de la programación. En algunos casos este puede tener más faces debido a la inclusión de un lenguaje intermedio a la hora de realizar la traducción [15].

4.2.1 MPLAB XC8®

Es un compilador libre que optimiza el estándar ISO C90/C99 utilizando el lenguaje C, este se puede usar para la programación de microcontroladores PIC® de 8 bits y AVR®. Este utiliza la interfaz común de C (CCI) teniendo en cuenta el estándar ANSI [16].

4.3 Comunicación USB®

El protocolo de comunicación USB® (Universal Serial Bus) es uno de los estándares de comunicación más utilizados comercialmente en la actualidad ya que en este momento la mayoría de periféricos o dispositivos que se conectan al computador como lo son cámaras, impresoras, escáneres, entre otros utilizan este tipo de comunicación [17].

Para realizar la comunicación entre dos dispositivos inicialmente se debe definir un dispositivo *Host*, normalmente es el computador, y un dispositivo *Device*. El *Host* será encargado de dirigir la comunicación, por lo tanto, debe tener toda la información de los dispositivos *Device* que se conecten con él [18].

4.3.1 Características

El protocolo de comunicación USB® se caracteriza por su facilidad de uso ya que se es muy versátil y se utiliza en una gran cantidad de dispositivos además de su característica de *Plug and Play* que permite la desconexión en caliente, es decir, que se puede conectar o desconectar en cualquier momento y forma sencilla soportando a su vez hasta 127 dispositivos conectados. Otra ventaja que presenta es la alimentación ya que se tiene un suministro de energía para los dispositivos de hasta 500mA [19]. De igual manera se caracteriza por las altas velocidades de transferencia de datos que puede alcanzar que pueden llegar a ser de hasta 40Gbits/s según la versión que se utilice [2].

4.3.2 Estructura

La estructura de este protocolo consiste en un *Host* (elemento que actúa como elemento principal o controlador del sistema USB®) el cual pude tener dos o tres puertos (zócalo a través del cual se accede a la red USB®) donde se pueden conectar ya sea un *Hub* (dispositivo que expande la cantidad de puertos disponibles) o un *Device* (dispositivos que van a interactuar con el *Host*) y finalmente al *Hub* irán conectados los *Device* que se vayan a utilizar [20]. Véase Figura 4

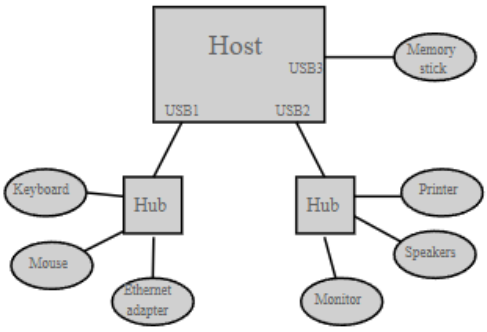


Figura 4. Estructura típica de una red USB®. Tomada de https://www.electronics-notes.com/

Otra manera en la que se puede usar el USB® es con un OTG (On-The-Go) un dispositivo que permite a otros aparatos con un comportamiento regular de *Device,* como tabletas y smartphones, trabajar como *Host* teniendo algunas limitaciones frente a las funciones comunes de este [21].

La comunicación USB® está basada en transferencia por *Data Pipes* que son los conductos por los cuales viaja la información para ello se tienen dos tipos de *Pipes* los de mensaje que se encargan de llevar la información de Host al Device y viceversa y los de corriente que van solo en dirección al Host y se encargan de las transferencias de información isócronas, de interrupción o masivas que vienen de los *Endpoints* (Búfer de almacenamiento) del *Device* [20]. *Véase* Figura 5

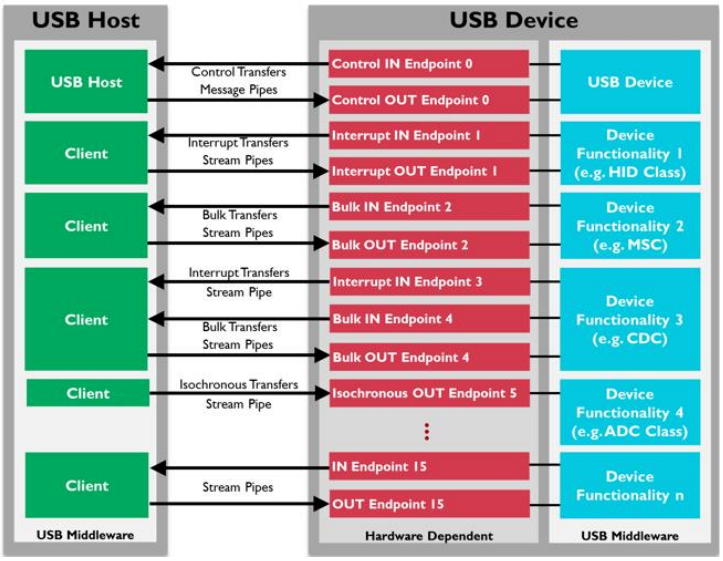


Figura 5. Estructura de Transferencia de datos por USB®. Tomada de www.keil.com

4.3.3 Funcionamiento

La comunicación USB® inicia en el momento que el *Host* identifica la presencia del *Device* donde se comienza la etapa de "Enumeración", allí el *Host* se encarga de reconocer el tipo de dispositivo que ha detectado y define la velocidad de comunicación. Esta etapa está dada por nueve pasos: Conexión entre el *Device* y el *Host*, Detección del *Device*, Identificación de la velocidad, Obtención de los descriptores (información codificada del dispositivo) del *Device*, Resetear el *Device* y Asignarle una dirección, Obtener los descriptores de configuración, Obtener los descriptores de interfaz, Leer los *Drivers* (archivos del *Host* con la información del *Device*) y finalmente Uso del *Device* [22]. Véase Figura 6.

No.	Time	Sync	In	Address	Endpoint	CRC5	Eop
8	575.968975ms	0X01	0X96	0X00	0X0	0X08	Eop

No.	Time	Sync	Data1	Data	Data	Data	Data	Data	Data	Data	Data	CRC16	Eop
9	576.02702ms	0X01	0XD2	0X12	0X01	0X10	0X01	0X00	0X00	0X00	0X08	0X88EE	Eop

No.	Time	Sync	ACK	Eop
10	576.095345ms	0X01	0X4B	Eop

No.	Time	Sync	In	Address	Endpoint	CRC5	Eop
11	576.110595ms	0X01	0X96	0X00	0X0	0X08	Eop

No.	Time	Sync	NAK	Eop
12	576.161845ms	0X01	0X5A	Eop

No.	Time	Sync	In	Address	Endpoint	CRC5	Eop
13	576.177845ms	0X01	0X96	0X00	0X0	0X08	Eop

No.	Time	Sync	NAK	Eop
14	576.229005ms	0X01	0X5A	Eop

Figura 6. Enumeración USB®. Tomada de https://www.perytech.com/

Una vez identificado el *Device* se puede iniciar con la transferencia de datos, para esto el *Host* inicializa la comunicación teniendo en cuenta los *Frames* (intervalos de tiempo controlados por el *Host*) que se determinan por la velocidad específica del *Device*. El inicio de la transacción (evento de comunicación) está dado por el inicio de *Frame* (SOF) seguido por la transacción, la cual está dividida en tres partes: Handshake, Data y Token, véase Figura 7, donde el Token (enviado por el Host) describe el tipo de transacción, Data es la información (DataOUT va hacia el Device, DataIN va hacia el Host) y Handshake el estado de la solicitud [23].Véase Tabla 1.

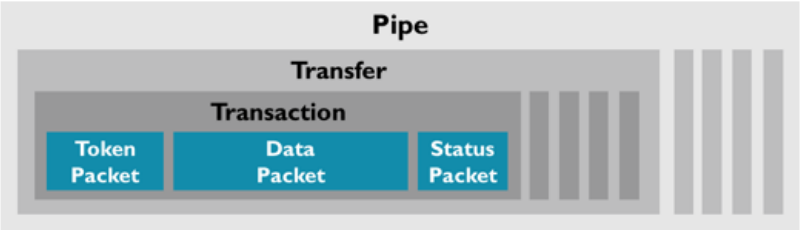


Figura 7. Transacción USB®. Tomada de PIC18F2455/2550/4455/4550 Data Sheet

Token Packets		
OUT	0001b	Inicio de una transacción de salida
IN	1001b	Inicio de una transacción de entrada
SET UP	0101b	Señales de inicio de una transacción de control
SOF	1101b	Paquete Start of Frame
Data Packets		
DATA0	0010b	Limpiar el bit de palanca en el paquete de datos
DATA1	1011b	Establecer el bit de palanca en el paquete de datos
Handshake		
ACK	0110b	Transacción exitosa
NACK	1010b	Transacción fallida
STALL	1110b	Respuesta retrasada, reintento necesario

Tabla 1. Transacciones USB®. Tomada de microchipdeveloper.com

4.3.3.1 Descriptores

Son estructuras de datos con un formato definido que proporcionan la información del *Device* para una correcta configuración por parte del *Host.* Existen cuatro tipos

de descriptores:

- **Device Descriptors (Descriptores de dispositivo):** Contienen la información del dispositivo, entre estos la identificación del producto (PID) y la identificación del vendedor (VID).
- **Configuration Descriptors (Descriptores de configuración):** Contiene información sobre sobre los requisitos de alimentación del dispositivo y la cantidad de interfaces que puede admitir.
- **Interface Descriptors (Descriptores de interfaz):** Definen la cantidad de *Endpoints* y *Pipes* que admite el dispositivo.
- **Endpoints Descriptors (Descriptores de Endpoints):** Especifica el tipo de transferencia, la dirección, el intervalo de sondeo y el tamaño máximo de paquete para cada *Endpoint.*

4.3.4 Historia

El estándar USB® nace en el año 1994 con la unión de un grupo de empresas llamado *USB® Implementers Forum* (USB-IF) entre las que se encuentran Apple, Compaq, DEC, Hewlett-Packad, IBM, Intel, Microsoft, NEC y Nortel que buscaban diseñar un estándar que permitiera unificar la conexión de dispositivos o periféricos y así sustituir todas las conexiones que existían en ese momento por una que permitiera optimizar los costos y esfuerzos en la fabricación de los mismos [1].

A mediados de la década de los 90 se lanzó al mercado la primera versión de este estándar USB1.0®, un par de años después, en 1998, se lanzó USB1.1® seguida por su versión USB2.0® en el año 2000, USB3.0 en el año 2008 y USB3.1® en el año 2013 [24].

En 2019 se presentó una nueva versión USB3.2® que prometía más velocidad de transferencia que las anteriores y en este mismo año se presentó la versión USB4.0® [25] que salió al mercado en 2021 [26].

4.3.5 Versiones

A lo largo de los años han salido al público varias versiones del protocolo USB® como se muestra en la Tabla 2, estas han sido mejoradas en cuanto a la velocidad y transferencia de datos. Además, tienen la característica de retrocompatibilidad [27].

Versión USB®	**Fecha de lanzamiento**	**Nombre**	**Velocidad de transferencia**
USB1.0®	Enero 1996	Full Speed	12Mbits/s
USB1.1®	Agosto 1998	Full Speed	12Mbits/s
USB2.0®	Abril 2000	High Speed	480Mbits/s
USB3.0®	Noviembre 2008	SuperSpeed	5Gbits/s
USB3.1®	Julio 2013	SuperSpeed+	10Gbits/s
USB3.2®	Septiembre 2017	SuperSpeed+	20Gbits/s
USB4.0®	2021		40Gbits/s

Tabla 2. Versiones USB®. Tomada de https://www.cmd-ltd.com/

4.3.6 Conectores

4.3.6.1 Tipo A

Es el primer conector USB® implementado con la versión USB1.0®, véase Figura 8, inicialmente fue implementado con una línea blanca, sin embargo, a partir de la versión USB3.0® se empezó a utilizar una línea azul para diferenciarlo de las versiones anteriores. Está compuesto por cuatro líneas, dos para la alimentación (Cables rojo y negro) y dos para el transporte de datos (Cables verde y rojo) [28]

Figura 8. Conector tipo A. Tomada de PcComponentes.com

4.3.6.2 Tipo B

Este conector tiene una forma cuadrada con dos de las esquinas redondeadas y sus pines están ubicados dos arriba y dos abajo, véase Figura 9, se utiliza en las versiones USB2.0® y USB3.0®, sus principales aplicaciones se encuentran en impresoras y escáneres [29].

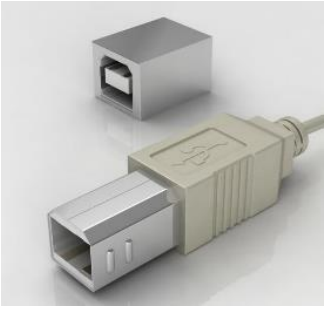


Figura 9. Conector tipo B. Tomada de https://www.turbosquid.com/

4.3.6.3 Mini USB®

Este conector es más pequeño que el estándar (tipo A), véase Figura 10, y tiene un pin extra conocido como el identificador ID el cual está diseñado para una mejora en el puerto, sin embargo, este normalmente no se utiliza. Se usa especialmente con la versión USB2.0® [30].

Figura 10. Conector Mini USB®. Tomada de https://culturacion.com/

4.3.6.4 Micro USB®

Este conector se caracteriza por su tamaño pequeño, véase Figura 11, sus usos más comunes actualmente son algunos smartphones y se utilizan con las versiones USB2.0® y USB3.0®, en esta última versión tienen un pin extra como una extensión a un lateral del conector estándar [31].

Figura 11. Conector Micro USB®. Tomada de www.turbosquid.com

4.3.6.5 Tipo C

Es un conector pequeño y ovalado, véase Figura 12, es conocido también como conector reversible ya que puede conectarse en ambos sentidos a diferencia de las versiones anteriores, por lo tanto, desde su salida al mercado se ha intentado convertir en el conector estándar para USB®. Se usa en las versiones USB3.1® en adelante y tiene ofrece mayor velocidad en transferencia de datos e intensidad en la corriente eléctrica [32].

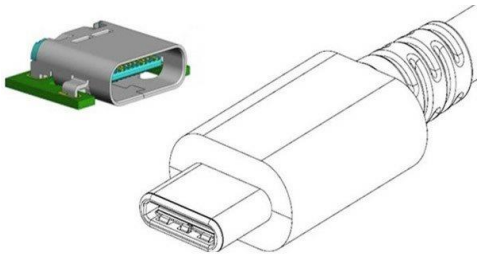


Figura 12. Conector tipo C. Tomada de https://blog.depau.es/

4.3.7 Clases

Las clases del protocolo USB® se utilizan para definir la función que cumplen los dispositivos de tal manera que se pueda identificar con mayor facilidad el *Driver* que se requiere para realizar la comunicación de manera correcta. La información que identifica el tipo de clase a utilizar está contenida en tres bytes (la clase base, la subclase y el protocolo), véase Tabla 3. Esta información se puede encontrar en el descriptor del dispositivo o en la interfaz del descriptor dependiendo de la clase utilizada [33].

Clase base	Subclase	Protocolo	Uso del descriptor	Descripción
00h	00h	00h	Dispositivo	Use class information in the Interface Descriptors
01h	Xxh	xxh	Interfaz	Audio
02h	Xxh	xxh	Ambos	Communications and CDC Control
03h	Xxh	xxh	Interfaz	HID (Human Interface Device)
05h	Xxh	xxh	Interfaz	Physical
06h	01h	01h	Interfaz	Image
07h	Xxh	xxh	Interfaz	Printer
08h	Xxh	xxh	Interfaz	Mass Storage
09h	00h	00h/01h/02h	Dispositivo	Hub
0Ah	Xxh	xxh	Interfaz	CDC-Data
0Bh	Xxh	xxh	Interfaz	Smart Card
0Dh	00h	00h	Interfaz	Content Security
0Eh	Xxh	xxh	Interfaz	Video
0Fh	Xxh	xxh	Interfaz	Personal Healthcare
10h	00h/01h/02h	00h	Interfaz	Audio/Video Devices
11h	00h	00h	Dispositivo	Billboard Device Class
12h	00h	00h	Interfaz	USB® Type-C Bridge Class
3Ch	00h	00h	Interfaz	I3C Device Class
DCh	00h/01h/02h/03h/ 4h/05h/06h/07h/08h	00h/01h	Ambos	Diagnostic Device
E0h	01h/02h	01h/02h/03h/04h	Interfaz	Wireless Controller
EFh	00h/01h/ 02h/04h	00h/01h/02h/03h/ 04h/05h/06h/07h	Ambos	Miscellaneous
FEh	01h/02h/03h	00h/01h	Interfaz	Application Specific
FFh	Xxh	xxh	Ambos	Vendor Specific

Tabla 3. Clases USB®. Tomada de www.usb.org

Entre las clases más destacadas se encuentran:

4.3.7.1 Communication Device Class (CDC)

Se identifica con el byte 02h y está dada por un elemento de administración y uno de notificación; el elemento administrativo es encargado de la configuración y funcionamiento del dispositivo y consiste en un *endpoint* 0, mientras el elemento de notificación es el encargado del transporte de datos hacia el *Host* y tiene un *endpoint* de interrupción [34]. Esta interfaz se genera mediante la emulación de puertos virtuales COM usando el modelo de control abstracto con las siguientes especificaciones [35].

- Descriptor de dispositivo estándar
- Descriptor de configuración estándar
- Descriptor de asociación de interfaz
- Descriptor funcional de encabezado de CDC
- Descriptor funcional de la unión de los CDC
- Descriptor funcional de gestión de llamadas
- Descriptor funcional de gestión de control abstracto
- Descriptor de interfaz estándar para la interfaz de comunicación de clase CDC
- Descriptor de punto final estándar para el punto final de interrupción IN
- Descriptor de interfaz estándar para la interfaz de datos de clase CDC
- Descriptores de punto final estándar para puntos finales Bulk IN y Bulk OUT

4.3.7.2 Human Interface Device (HID)

Es una de las clases más comunes, se identifica con el byte 03h, esta clase se utiliza especialmente para dispositivos que tienen una interacción directa con las personas para controlar los sistemas de cómputo como lo son los teclados, punteros y pedales [36]. Estos dispositivos cumplen con características como su intercambio de datos en informes (*reports*) que son solicitados por el *Host* como transferencias de control o interrupción además requiere de un *endpoint* de entrada para realizar su comunicación con el *Host*, de igual manera hace uso de los siguientes descriptores en el *Device* para identificarse a la hora de hacer la comunicación [37]:

- Standard Device Descriptor
- Standard Configuration Descriptor
- Standard Interface Descriptor for the HID Class
- Class-Specific HID Descriptor
- Standard Endpoint Descriptor for Interrupt IN endpoint
- Class-Specific Report Descriptor

4.3.7.3 Mass Storage Class (MSD)

Se encuentra en dispositivos de almacenamiento como los pendrives y discos duros. Esta clase permite la transferencia de inmensas cantidades de información entre los dispositivos que se realiza la comunicación [38]. Para la exitosa transferencia de datos se requiere de tres etapas [39]:

- Transporte de comandos: El *Host* envía un comando en *estructura Command Block Wrapper (CBW)*
- Transporte de datos: La información es enviada hacia o desde el *Host*
- Transporte de estado: El *Device* envía el estado en estructura *Command Status Wrapper (CSW)*

4.4 Software

4.4.1 Windows

Es un conjunto de softwares, servidores y dispositivos móviles desarrollado por Microsoft con diferentes arquitecturas como lo son x86, x86-64 y ARM. Este permite en gran medida una mejor interacción entre el usuario y el Hardware utilizado ya que es el encargado de administrar los recursos del ordenador, además de ofrecer una interfaz sencilla e intuitiva y herramientas que permiten comodidad a la hora de manejar el equipo [40].

4.4.2 Raspbian

También conocido como Raspberry Pi OS, es el sistema operativo usado por las Raspberry Pi, este está basado en Linux y se utiliza normalmente con procesadores de arquitectura ARM. Algunas versiones traen programas previamente instalados y tienen un escritorio tipo PIXEL (Pi Improved X-Window Environment, Lightweight) que se basa en LXDE y es muy ligero [41].

4.4.3 MPLAB® X IDE

Es un software gratuito que permite configurar, desarrollar, depurar y calificar diseños de los microcontroladores que ofrece la marca Microchip [42], cuenta con una gestión completa de proyectos, una ventana de observación configurable y un editor con funciones que incluyen finalización de código y navegación por hipervínculos [43].

4.4.4 Proteus

Es un software que permite la simulación de circuitos electrónicos además de la elaboración de esquemas y placas electrónicas. Este tiene cuatro herramientas diferentes para el desarrollo de los circuitos: ISIS que permite la elaboración de esquemas eléctricos, ARES que permite la elaboración de placas de circuito impreso, PROSPICE que sirve para realizar simulaciones y VSM (Virtual System Modelling) que simula microcontroladores como PIC®, AVR®, 8051, HC11, ARM/LPC200 y BASIC STAMP [44].

4.4.5 Matlab

Matlab es un software utilizado especialmente para realizar cálculos con vectores y matrices, para la representación gráfica de dos y tres dimensiones. Además, permite la simulación, modelado y análisis de sistemas de control [45]. Este sistema cuenta con un lenguaje de programación propio, el cuales interpretado y puede ejecutarse a través de un entorno interactivo por medio de un archivo script, además de su entorno para la creación de interfaces de usuario GUI, entre otros complementos que permiten realizar tareas de comunicación, visualización, representación, análisis de datos, entre otras [46].

4.4.6 Python

Python es un lenguaje de programación interpretado de código abierto que sirve para trabajar con grandes volúmenes de datos, es usado por grades empresas alrededor del mundo para construir aplicaciones web, analizar datos, automatizar operaciones y crear aplicaciones empresariales fiables y escalables. Es un lenguaje multiparadigma utilizado en campos aparentemente complejos como diseño de aplicaciones o la inteligencia artificial, entre otros [47].

4.5 Dispositivos y componentes electrónicos

4.5.1 Teclado matricial

Es un teclado básico con 16 teclas con un formato de matriz 4x4 que permite al microcontrolador escanear los pines de salida para identificar cual es presionado. Viene etiquetado con los símbolos 1, 2, 3, A, 4, 5, 6, B, 7, 8, 9, C, 0, *, # y D y tiene un formato telefónico [48].

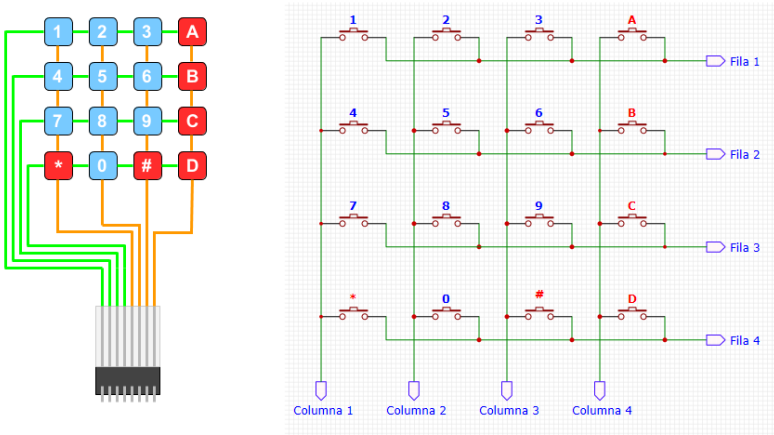


Figura 13. Teclado matricial. Tomada de http://arduparatodos.blogspot.com/

4.5.2 Servomotor

Un servomotor es un motor eléctrico al cual se le puede controlar la velocidad y posición en el eje de giro, su movimiento no supera los 180°, normalmente utilizan corriente continua (DC) con un voltaje entre 4 y 8 voltios, pero existen también de

corriente alterna (AC). El servomotor está compuesto por un motor, un sistema de engranajes que regula su torque y velocidad, un sistema de control y un potenciómetro que se encargan de dar exactitud al posicionamiento del motor por medio de señales eléctrica. Sus aplicaciones se encuentran en diferentes ramas de la ingeniería como la robótica [49].

Figura 14. Servomotor. Tomada de http://panamahitek.com/

4.5.3 Módulo Bluetooth HC-05

Es un dispositivo que permite la conexión de un Arduino, Microcontrolador y dispositivos similares con un dispositivo con conexión bluetooth como un smartphone haciendo uso de un puerto serial USART. Su alimentación es de 3.3 voltios, aunque permite también la conexión con 6 voltios. Este dispositivo se configura por medio de comandos AT que permiten variar el nombre, la velocidad de transmisión y la clave de acceso. Además, se puede utilizar en modo Maestro o Esclavo según sean las necesidades del programador [50].

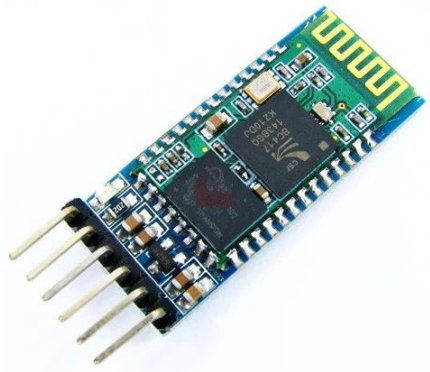


Figura 15. Módulo Bluetooth. Tomada de www.geekfactory.mx

4.5.4 RTC (Real Time Clock)

Es un reloj computarizado que contabiliza el tiempo (horas, minutos, segundos, días, meses y años), utilizan un oscilador de cristal lo que les da independencia de las señales de reloj, tienen un sistema de temporización que les permite la exactitud en su sincronización. Se usan normalmente en dispositivos como computadores personales, sistemas integrados y servidores [51].

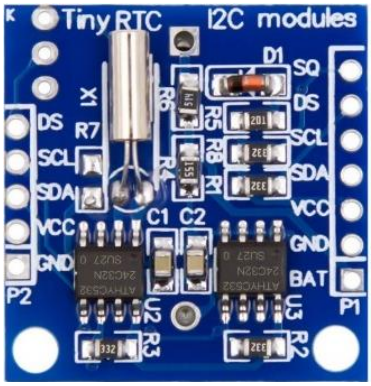


Figura 16. Módulo RTC DS1307. Tomada de https://gleanntronics.ie/

4.5.5 LCD (Liquid Crystal Display)

Es una pantalla que usa una sustancia liquida atrapada entre dos placas de vidrio que consigue mostrar determinados símbolos según la iluminación que se genera por medio de corrientes eléctricas inducidas. Estas pueden ser de un solo color o utilizar células RGB para la implementación de diferentes tonos, además vienen de diversos tamaños, resoluciones y permiten la modulación del brillo trasero [52].

Figura 17. LCD 16x2. Tomada de www.orientdisplay.com

4.5.6 DHT11

Es un sensor usado medir temperatura en un rango de 0°C a 50° C y humedad en un rango de 20% a 90% con una resolución de 16 bits. Este tiene un NTC con el cual toma la medida de la temperatura y un microcontrolador con 8 bits de salida para los datos medidos con una exactitud entre ±1°C y ±1%. Algunas de sus aplicaciones son medir la temperatura y la humedad, climatizadores automáticos y monitoreo del entorno [53].

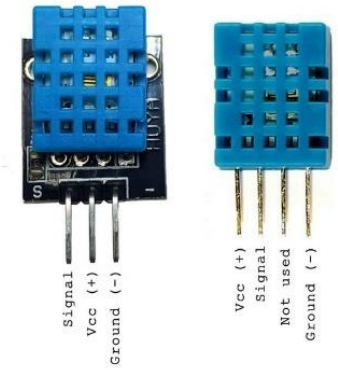


Figura 18. Sensor DHT11. Tomada de https://surtrtech.com/

4.5.7 Relé de estado solido

Un relé es un interruptor que permite el paso de corriente cuando está cerrado y lo interrumpe cuando está abierto y se acciona eléctricamente. Está compuesto por una bobina que genera un campo electromagnético que a su vez hace que su contacto interno cambie de estado. Por su parte un relé de estado sólido cumple la misma función con la diferencia que no tiene partes móviles por lo que para cambiar de estado utiliza materiales semiconductores que le permiten una velocidad de conmutación mayor, además, protegen al circuito de estática, no tienen contacto ni rebote y tienen una mayor durabilidad. [54]

Figura 19. Relé de estado sólido. Tomada de https://electrojoan.com/

4.5.8 Dimmer digital

Es un dispositivo usualmente usado para regular la intensidad de la luz con bombillos que utilizan corriente alterna (AC). Estos regulan el voltaje RMS recibido por el bombillo para de esta manera modular su brillo. La mayoría incluyen un encendido “paso por cero” de la tensión mediante la cual se referencia el corte de la señal de salida obteniendo una onda de entre 60 voltios y 200 voltios según la intensidad de luz que se desee [55].

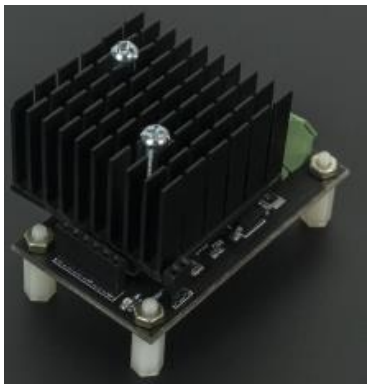


Figura 20. Dimmer digital. Tomada de www.vistronica.com

4.5.9 Raspberry Pi

Es un computador de pequeñas dimensiones que se adapta al uso de periféricos de entrada y salida como un mouse, teclado y pantalla para interactuar con el usuario [56]. Sirve especialmente para la implementación de prototipos, utiliza sistemas operativos GNU/Linux como Raspbian y tiene una arquitectura de procesador ARM distinta a la de los ordenadores de mesa ya que puede realizar tareas con menor consumo de energía [57]. Su versión 4.0, una de las más recientes, cuenta con un procesador Broadcom BCM2711 y una arquitectura de 64 bits además de una velocidad de procesamiento de 1.5GHz. Por otro lado, cuenta con puertos USB® para las versiones 2.0 y 3.0 y un controlador Ethernet nativo; su memoria RAM puede estar entre 1GB, 2GB y 4GB según la necesidad del usuario. Finalmente, su alimentación se puede dar utilizando el puerto USB® tipo C, el PoE HAT o mediante sus conectores GPIO [58].

Figura 21. Raspberry Pi. Tomada de https://hipertextual.com/

4.6 PWM (Pulse Width Modulation)

La modulación por ancho de pulso (PWM) sirve para generar cantidades intermedias de potencia eléctrica entre la potencia máxima y el apagado. Consiste en apagar y encender rápidamente un interruptor que permite el paso de la energía. Para esto, se aplica la potencia máxima durante cortos periodos de tiempo determinando el tiempo de encendido por medio del Duty-Cycle (ciclo de trabajo) [59], véase Figura 22.

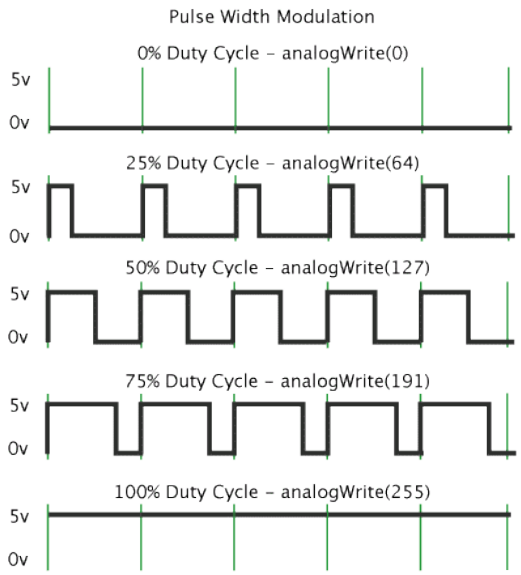


Figura 22. Función del PWM. Tomada de http://blog.electricbricks.com/

4.6.1 PWM en Corriente alterna

Para generar un PWM en corriente alterna se requiere un procedimiento similar al anterior utilizando como interruptor un dispositivo que permita el manejo de señal alterna como un tiristor, un MOSFET o un Triac teniendo en cuenta el paso de la señal por cero (Zero cross) y el momento de disparo de la señal PWM por corriente directa [60], véase Figura 23.

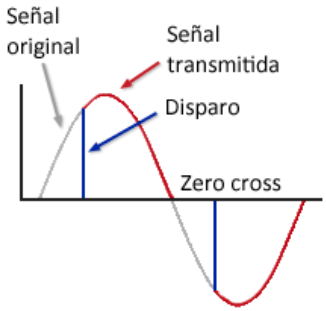


Figura 23. Señal de PWM en AC. Tomada de vizcainostore.com

4.7 Lógica difusa

La lógica difusa parte de la imposibilidad de definir con total claridad una afirmación como verdadera o falsa teniendo en cuenta que según el punto de vista puede variar su estado y considerando los puntos intermedios que se puedan llegar a presentar en dicha afirmación. Por ende, los conjuntos a evaluar se toman de manera lingüística donde determinados elementos pueden pertenecer a más de un conjunto a la vez, de manera tal que dichos elementos tienen un rango de pertenencia al conjunto definido por un valor numérico entre 0 y 1 donde “0” significa que no pertenece al conjunto en absoluto y “1” significa una total pertenencia [61].

5. Metodología

Para iniciar este proyecto se plantean una serie de pasos a seguir, véase Figura 24, con el fin implementar del protocolo de comunicación USB-CDC® con el microcontrolador PIC18F4550 usando el compilador XC8® utilizando el software MPLAB® X, para acceder a los códigos de los programas presentados a continuación ingrese al siguiente link: https://gitlab.com/GABI2603/usb-cdc.git.

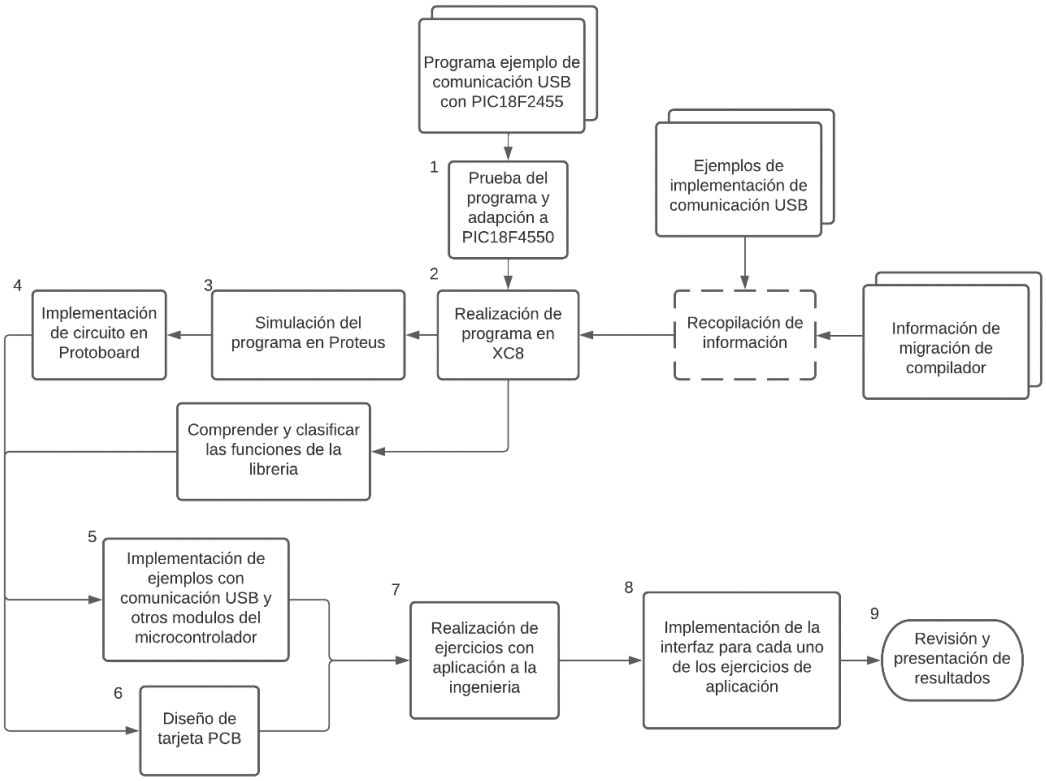


Figura 24. Etapas del proyecto. Fuente propia

5.1 Prueba y adaptación de programa base

Inicialmente se utiliza el programa propuesto en el foro TODOPIC [62] en el cual, se presenta un ejemplo de comunicación USB-CDC® utilizando el microcontrolador PIC18F2455 y el compilador C18®.

Teniendo en cuenta el objetivo de implementar dicho programa en el compilador XC8® se realiza la búsqueda de otros ejemplos que permitan la adaptación de este. Entre estos se encuentra un ejemplo con dicho compilador utilizando el PIC16F1455 y el complemento de MPLAB® X, MPLAB® Code Configurator (MCC) [63]. De esta manera al explorar los dos ejemplos se realiza un primer programa con la estructura mostrada en la Figura 25, el cual realiza la comunicación USB-CDC® entre el microcontrolador y el computador, siendo este la base del proyecto en general.

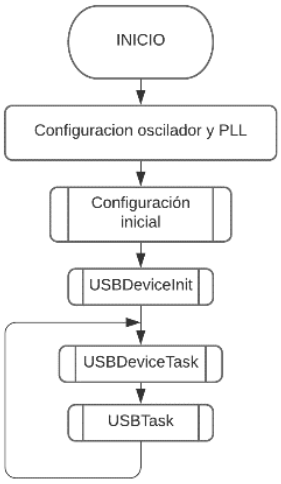


Figura 25. Programa base. Fuente propia

Este programa utiliza las librerías generadas por el MPLAB® Code Configurator, véase anexo 1. para los microcontroladores PIC18FxxKxx donde se destacan las funciones USBDeviceInit y USBDeviceTask las cuales se encargan de la inicialización de la comunicación y evitar la desconexión del dispositivo respectivamente. Por otra parte, se tiene en cuenta que la frecuencia de trabajo del microcontrolador para realizar dicha comunicación debe ser de 48MHz, por lo tanto, para este programa se utiliza el módulo PLL (Phase Locked Loop) del microcontrolador que permite elevar la frecuencia proporcionada por el cristal externo de 20MHz. De igual manera, Se tienen las funciones de configuración inicial con la cual se configuran los módulos a usar y la función USBTask presentada en la Figura 26, que es la encargada de realizar la comunicación de ejemplo.

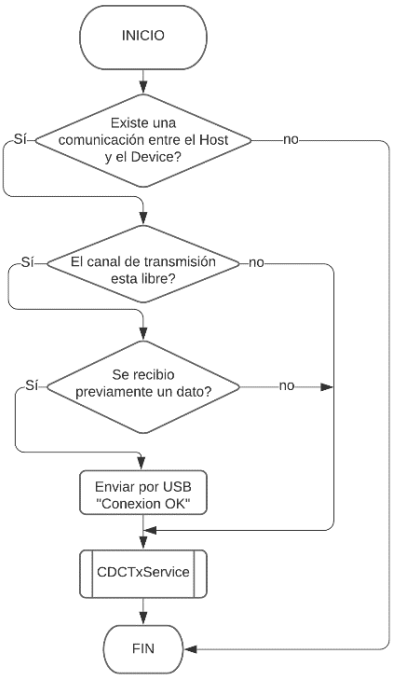


Figura 26. Función USBTask. Fuente propia

En la función USBTask se tienen en cuenta dos funciones importantes getsUSBUSART y putrsUSBUSART que son las encargadas de enviar y recibir los datos de la comunicación.

Adicionalmente, en las librerías utilizadas se tienen una serie de funciones que permiten realizar otras acciones en la comunicación USB® como cambiar la velocidad de trabajo o incluso permitir el uso de otras clases como USB-HID®. Véase anexo 2.

Para la implementación de este primer programa se debe realizar el montaje de la Figura 27, donde se aprecia la unión del microcontrolador con el conector USB, además de otros requerimientos como el cristal, la alimentación y el condensador del pin Vbus que se requiere para realizar la comunicación.

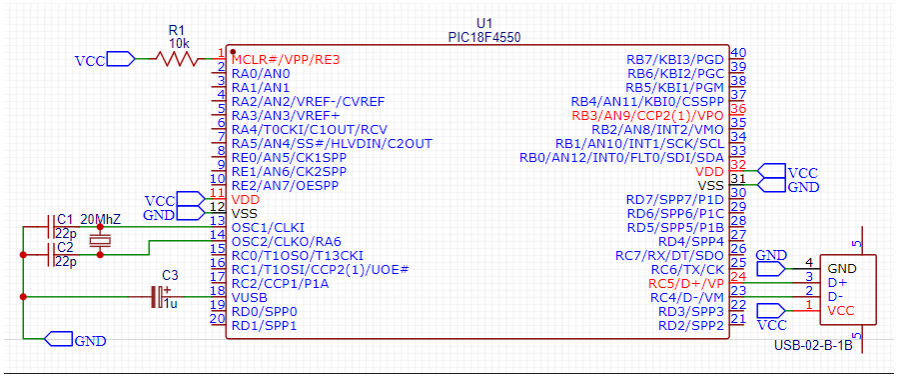


Figura 27. Montaje de programa base. Fuente propia

5.2 Ejemplos por Módulo

Teniendo como guía el programa base mostrado anteriormente, véase Figura 25, se procede a realizar ejemplos que utilicen los demás módulos del microcontrolador, véase Figura 28.

Figura 28. Implementación de comunicación USB y módulos del microcontrolador. Fuente propia

5.2.1TIMER0

El primer módulo a implementar es el TIMER, este permite llevar el conteo del tiempo teniendo en cuenta la ecuación 1.

$$T = \frac{4}{F_{osc}} * Preescaler * (65535 - TMR0) \qquad (1)$$

Para este programa se plantea un periodo de 1s utilizando la interrupción del TIMER, Véase Figura 29, para llevar el conteo del tiempo que el dispositivo ha estado funcionando y guardando el dato para que sea mostrado en la pantalla del computador.

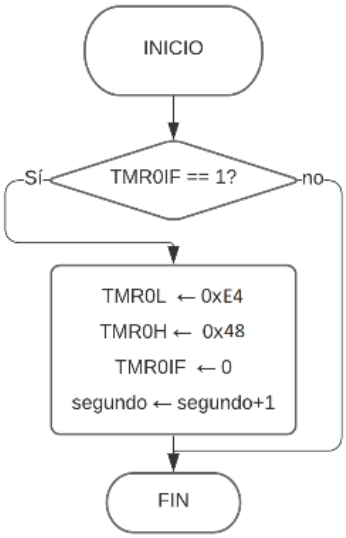


Figura 29. Interrupción TIMER0. Fuente propia

El planteamiento de este programa se encuentra en la Figura 30, donde se observa que se lleva el conteo de horas, minutos y segundos, además su visualización dependerá del envío de información desde el computador.

Cabe resaltar que para la implementación de este programa se utilizará el mismo circuito del programa base (Figura 27).

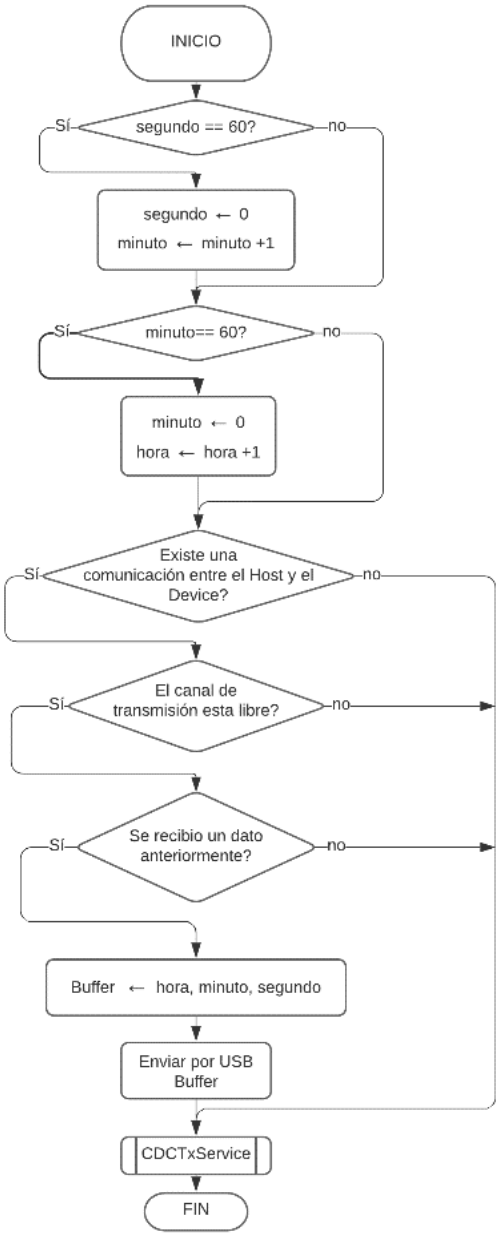


Figura 30. Función USBTask del programa TIMER0. Fuente propia

5.2.2 Módulo de entradas y salidas (I/O)

Este módulo es el encargado del control de entradas y salidas en el microcontrolador, por eso para el uso de este se propone el uso de un teclado matricial que servirá para enviar caracteres hacia el computador, además de un pulsador que envía un mensaje y Leds que serán controlados por comandos recibidos por el PIC®. Véase Figura 31.

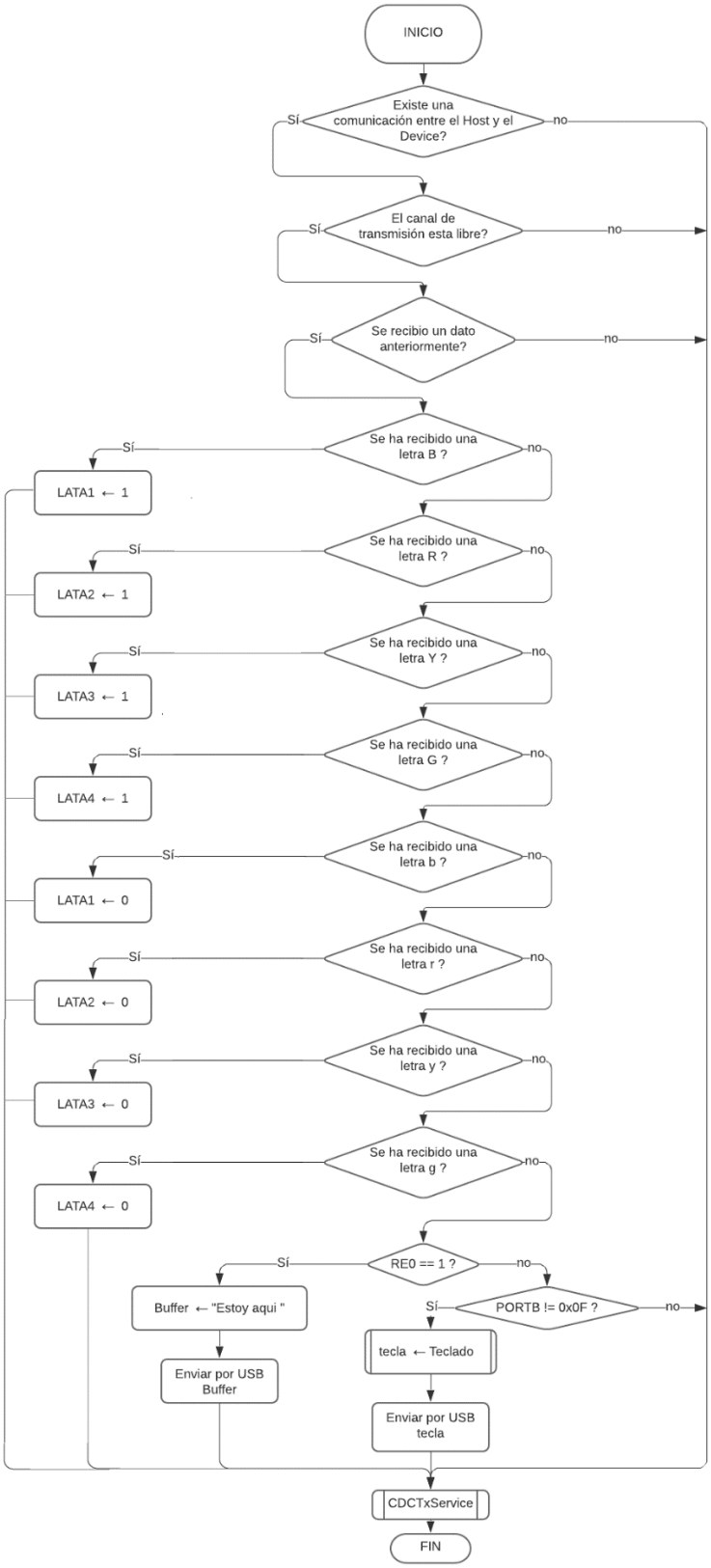


Figura 31. Función USBTask Módulo I/O. Fuente propia

El montaje para este ejemplo se puede observar en la Figura 32, donde en el puerto B se encuentra el teclado matricial, en el puerto A se encuentran los leds y en el puerto E el pulsador.

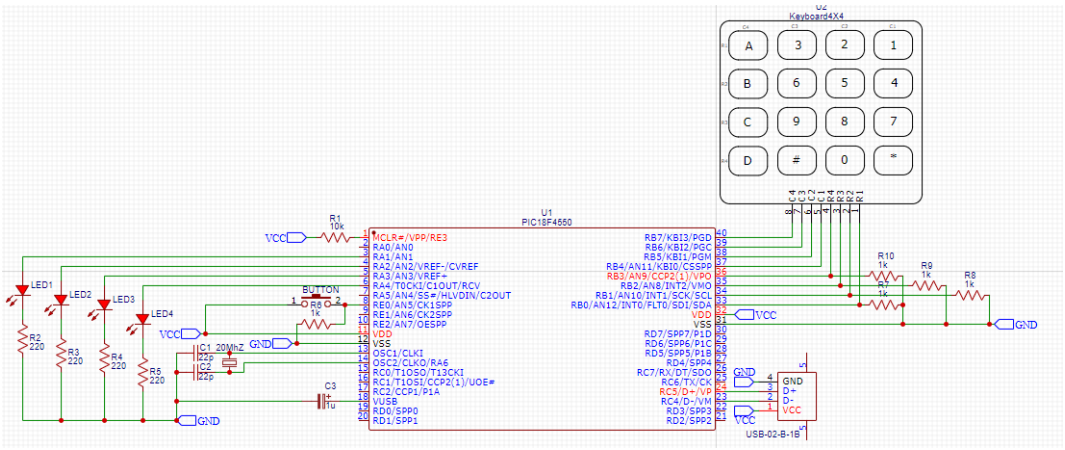


Figura 32. Montaje programa Módulo entradas y salidas. Fuente propia

5.2.3 Módulo ADC (Conversor Análogo-Digital)

Como su nombre lo indica este módulo permite realizar conversiones de datos análogos a datos digitales, por lo tanto, para este programa se propone una recolección y visualización de información análoga utilizando un potenciómetro para graduar el voltaje leído por el microcontrolador entre 0 voltios y 5 voltios. Su programación inicia con la función que permite la lectura de los valores análogos y su respectiva conversión, véase Figura 33. Para iniciar el uso de esta función se inicia la comunicación mediante la letra 'C' y finaliza al completar 100 muestras, Véase Figura 34.

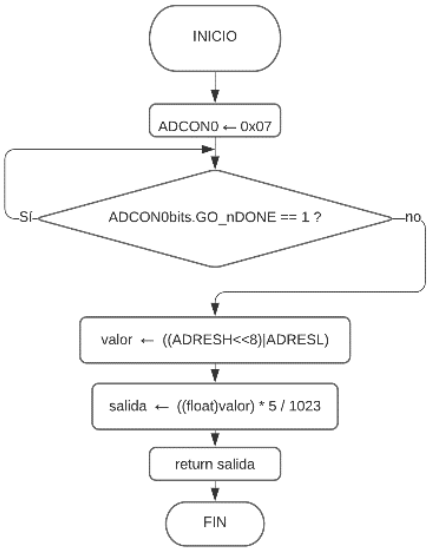


Figura 33. Función ADC. Fuente propia

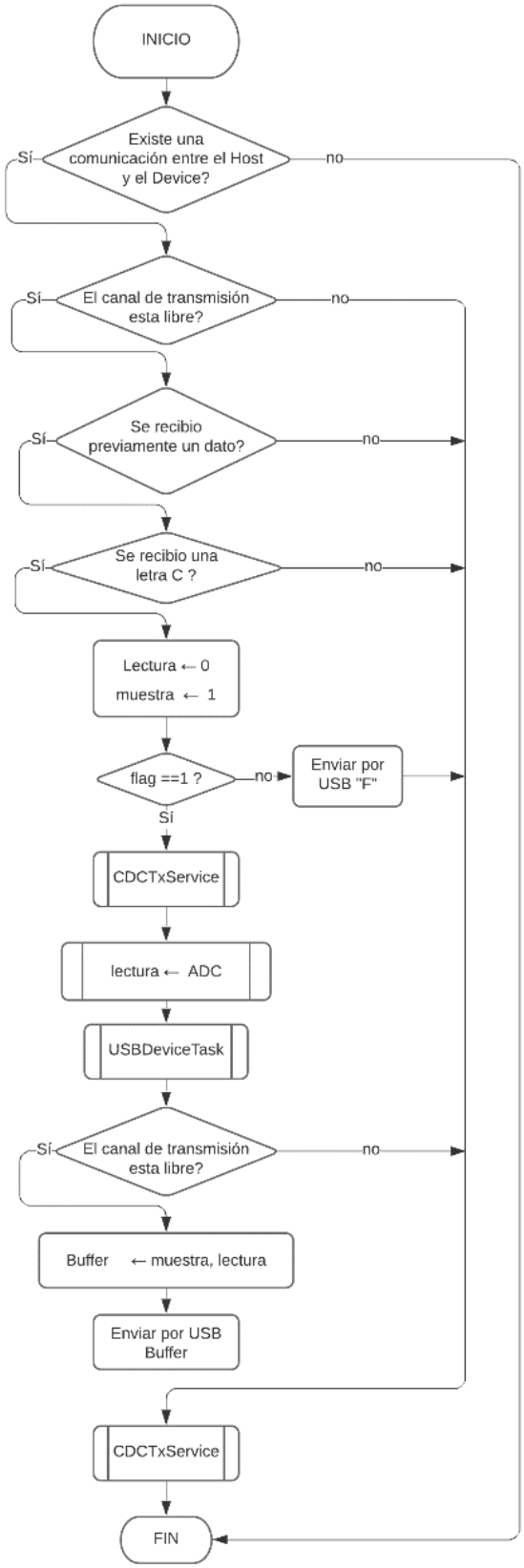


Figura 34. Función USBTask Módulo ADC. Fuente propia

Al programa anterior se le realiza una modificación al añadir una interrupción que permite detener la recolección de datos al presionar un pulsador. Véase Figura 35. El circuito para su implementación se puede observar en la Figura 36.

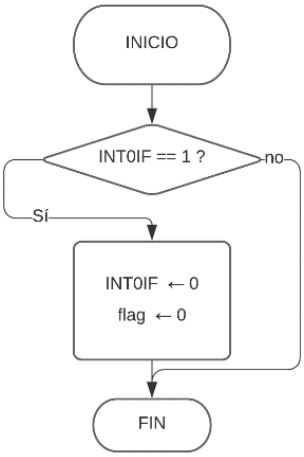


Figura 35. Interrupción por pulsador. Fuente propia

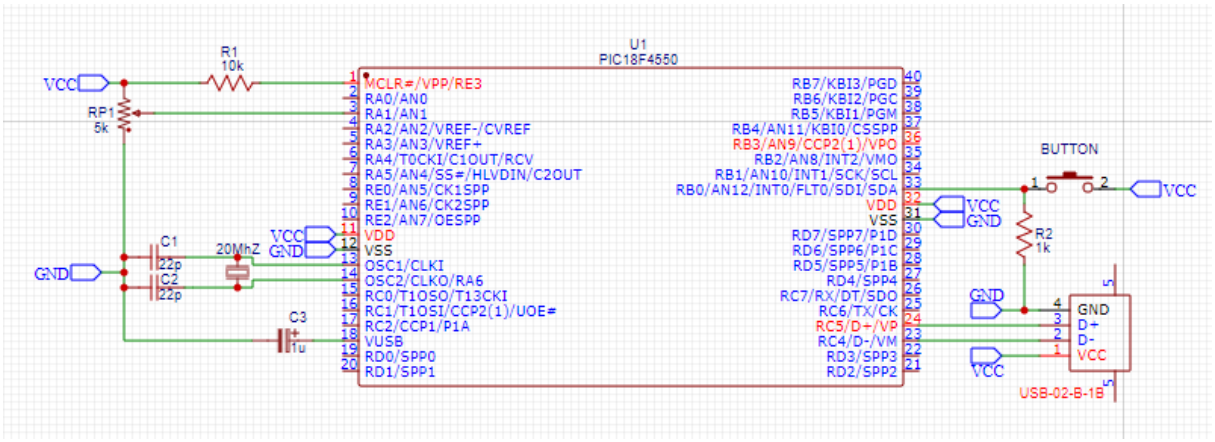


Figura 36. Montaje de programa Módulo ADC. Fuente propia

5.2.4 Módulo de Referencia

Este módulo permite generar señales análogas entre 0 voltios y 3 voltios por medio de 16 etapas. Para este programa se propone la modulación de dicho voltaje enviando un numero entre 0 y 15 desde el computador para cambiar la cantidad de voltaje en el pin del microcontrolador, Véase Figura 37

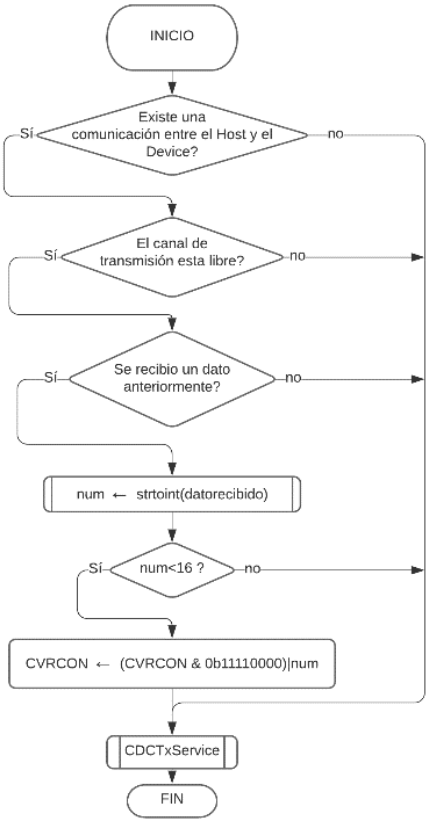


Figura 37. Función USBTask Módulo Referencia. Fuente propia

Para este programa se utiliza el pin RA2 del microcontrolador, véase Figura 38, en el cual se planteó poner un led para variar su intensidad. Sin embargo, en este montaje se muestra un multímetro ya que mediante este instrumento se puede conocer con mayor exactitud el voltaje que hay en este pin.

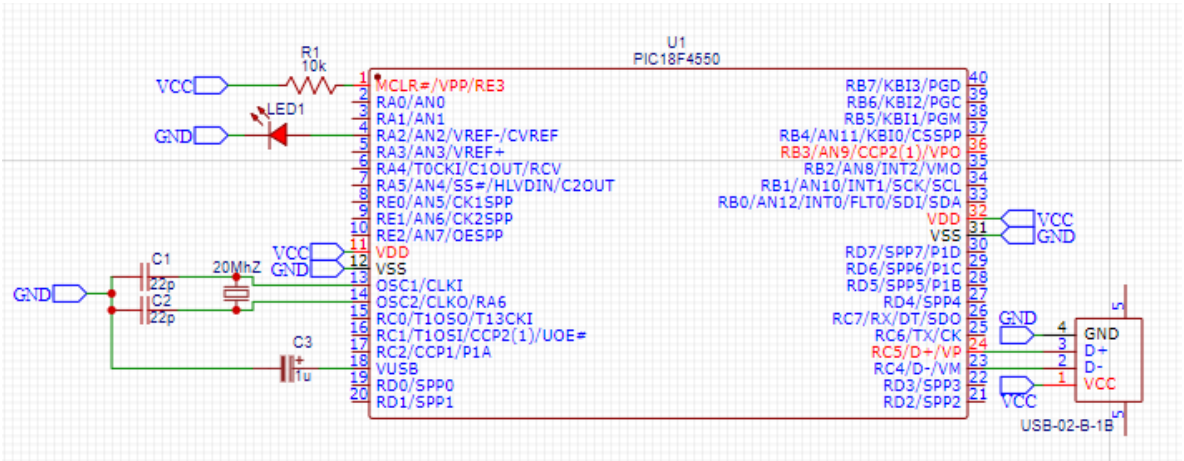


Figura 38. Montaje de Módulo Referencia. Fuente propia

5.2.5 Módulo CMP (Comparador)

Este módulo utiliza los comparadores internos del microcontrolador para comparar dos entradas de voltaje, determinar la de mayor valor y de esta manera tomar un estado de salida de 0 o 1 lógico. Para este programa se utiliza este módulo comparando los 3 voltios conectados a la entrada no inversora con el voltaje leído en la entrada inversora y mostrando por pantalla 'Estado activo' en caso de tener un voltaje mayor al de referencia (1 lógico) y 'Estado no activo' al tener un voltaje menor (0 lógico). Véase Figura 39.

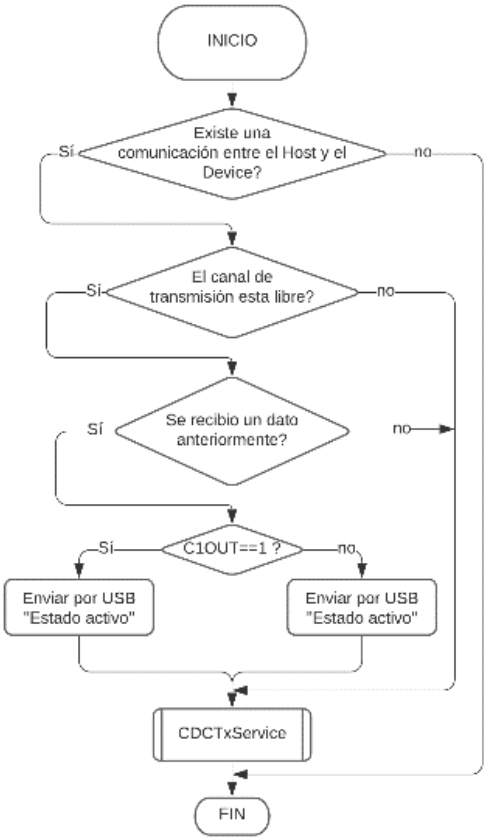


Figura 39. Función USBTask Módulo Comparador. Fuente propia

Para este ejemplo se utilizó un divisor de voltaje con dos resistencias entre Vcc (5V) y GND para obtener los 3 voltios de referencia conectados al pin RA0, además se utilizó un pulsador en *pull-down* para obtener 5V al presionarlo y una conexión a tierra en caso contrario, véase Figura 40.

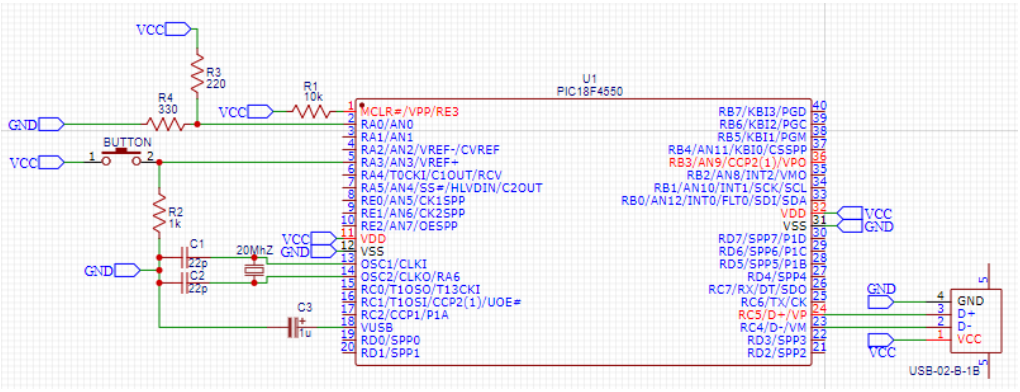


Figura 40. Montaje programa Módulo CMP. Fuente propia

5.2.6 Módulo CCP

Para la implementación de este módulo se proponen dos ejemplos utilizando dos funciones diferentes del mismo.

5.2.6.1 PWM

Este módulo funciona con el TIMER2 mediante el cual se establece la frecuencia de trabajo usando el registro PR2 que se configura teniendo en cuenta la Ecuación 2 y el registro CCPR1L que se encarga de definir el ancho del pulso. En la se muestra la Figura 41 implementación de este.

$$TMR2 = Preescaler * (PR2 + 1) * Postescaler * \frac{4}{F_{osc}} \qquad (2)$$

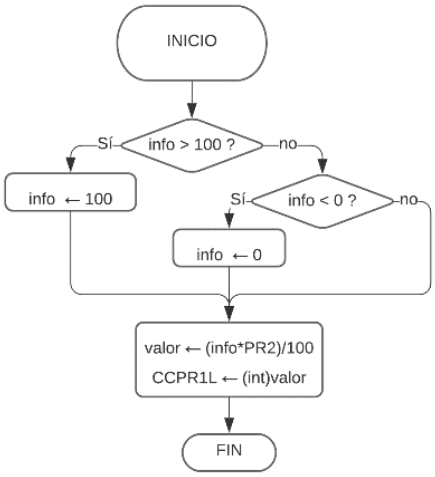


Figura 41. Función PWM. Fuente propia

Para este programa se varia la intensidad de un led usando el PWM del microcontrolador y controlándolo según el valor enviado desde el computador, teniendo la estructura del programa que se muestra en la Figura 42. El led se debe ubicar en el pin RC2, véase Figura 43, ya que este es el utilizado por el módulo.

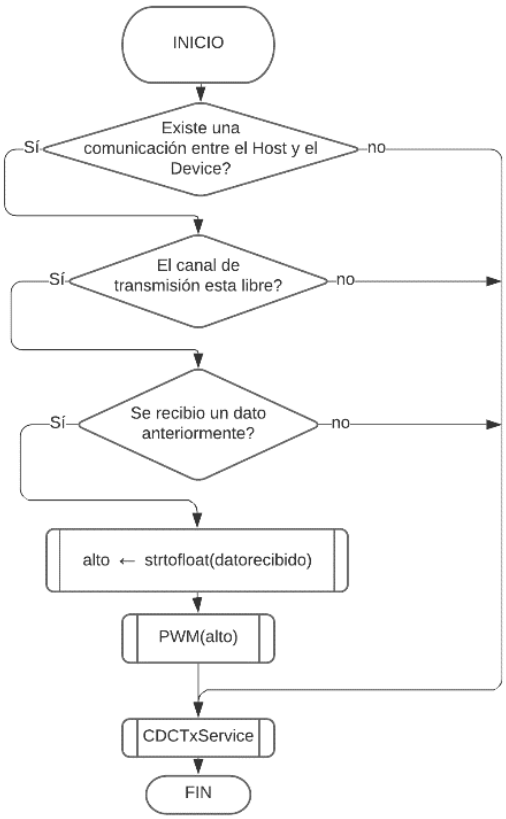


Figura 42. Función USBTask Módulo PWM. Fuente propia

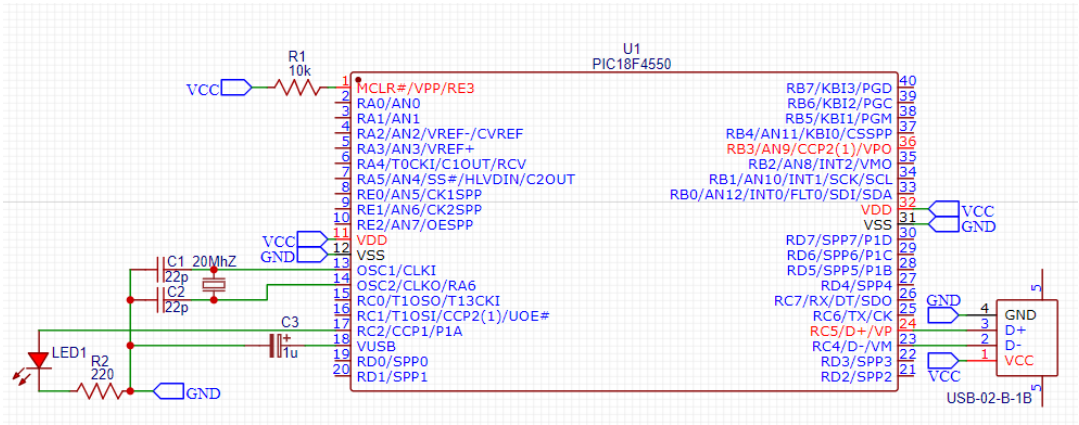


Figura 43. Montaje programa Módulo CCP (PWM). Fuente propia

5.2.6.2 Modo Comparación

Este módulo utiliza el TIMER1 para definir su frecuencia de trabajo haciendo uso de la Ecuación 3. Además, utiliza una interrupción que le permite regular el ancho de pulso, Véase Figura 44.

$$T = \frac{4}{F_{osc}} * Preescaler * (65535 - TMR1) \qquad (3)$$

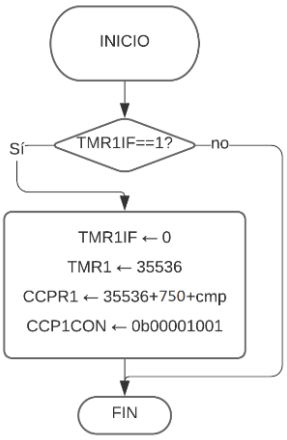


Figura 44. Interrupción Timer1. Fuente propia

Siendo así, el programa propuesto consiste en controlar la posición de un Servomotor definiéndola mediante el valor enviado por el computador, Véase Figura 45.

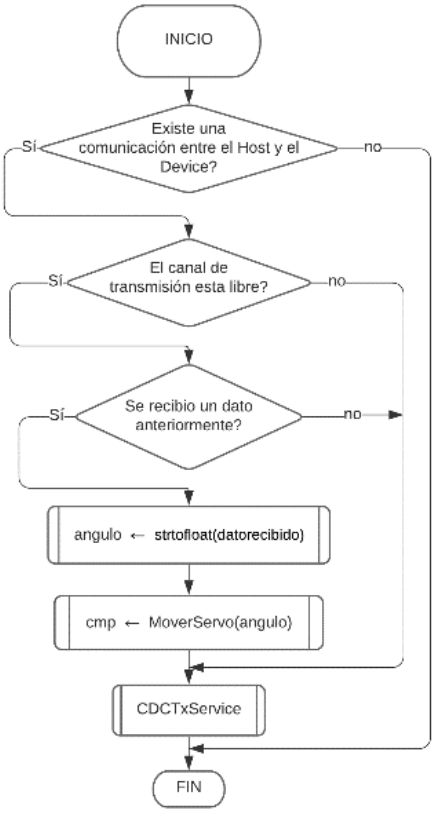


Figura 45. Función USBTask Módulo CCP (Comparador). Fuente propia

Donde la Función MoverServo realiza una conversión del ángulo (en deg) que se desea para obtener el valor correspondiente al pulso requerido para posicionar el servomotor. Véase Figura 46.

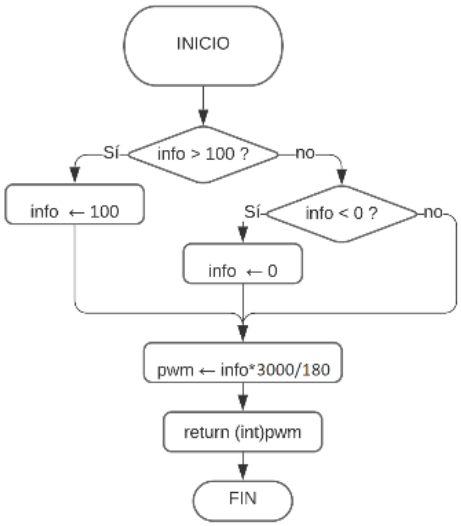


Figura 46. Función MoverServo. Fuente propia

Para la implementación de este programa se usa el mismo circuito que en el programa anterior cambiando el led por el servomotor, véase Figura 47

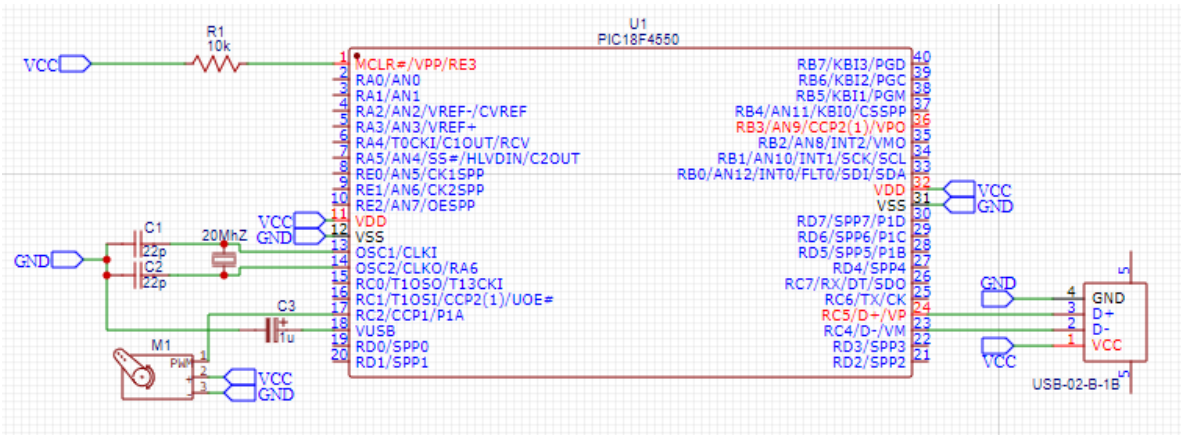


Figura 47. Montaje programa Módulo CCP (Comparador). Fuente propia

5.2.7 Módulo EUSART

Este módulo permite al microcontrolador realizar la comunicación serial mediante el protocolo RS232, para lo cual se usa un registro que permite el envío de datos y una función de interrupción para la lectura de los mismos, Véase Figura 48.

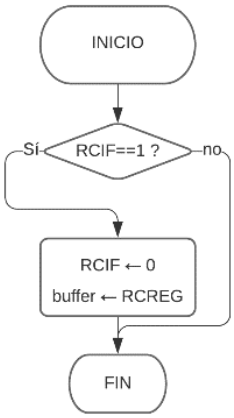


Figura 48. Interrupción módulo EUSART. Fuente propia

Para este módulo se propone un programa que comunica un celular con un computador utilizando el módulo Bluetooth HC-05 y la comunicación USB-CDC® entre el microcontrolador y el computador, cuyo programa se muestra en la Figura 49 y la función de transmisión en la Figura 49

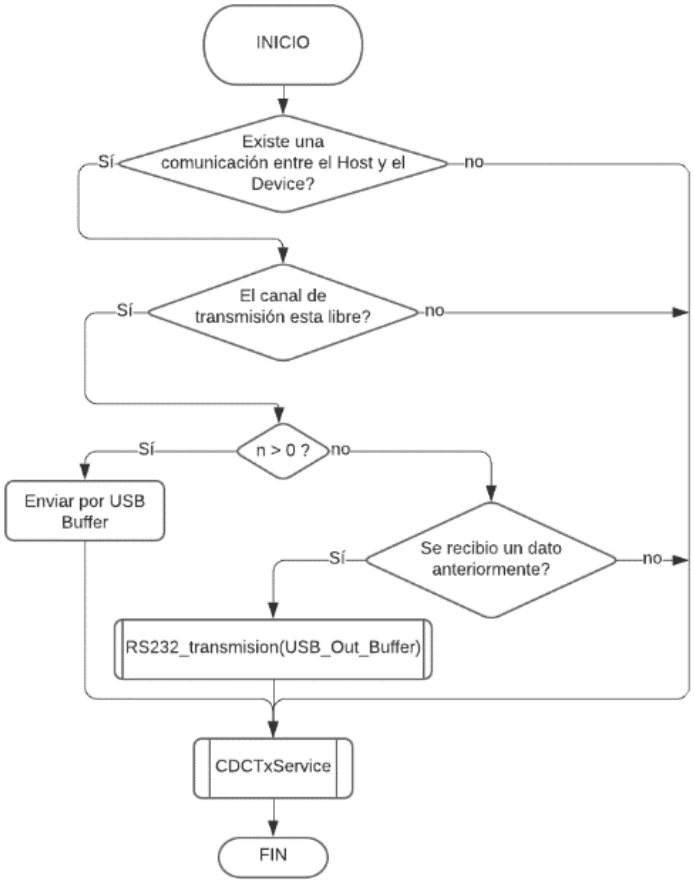


Figura 49. Función USBTask Módulo EUSART. Fuente propia

En la función USBTask se muestra también la función RSS232_transmisión que es la encargada de enviar la información del PIC® al celular, véase Figura 50.

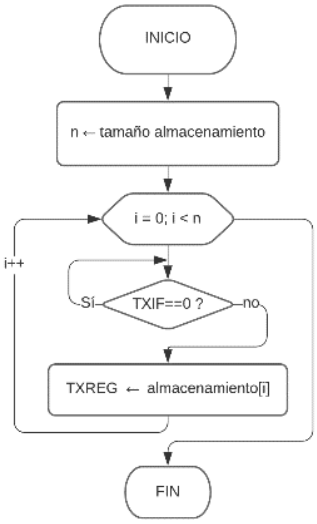


Figura 50. Función RS232_transmision. Fuente propia

De igual manera para visualizar la información que recibe el celular se diseña una aplicación donde además de visualizar la información permite el envío de datos hacia el PIC® y eventualmente al computador esta se muestra en la Figura 51 donde además se aprecia el montaje respectivo del programa.

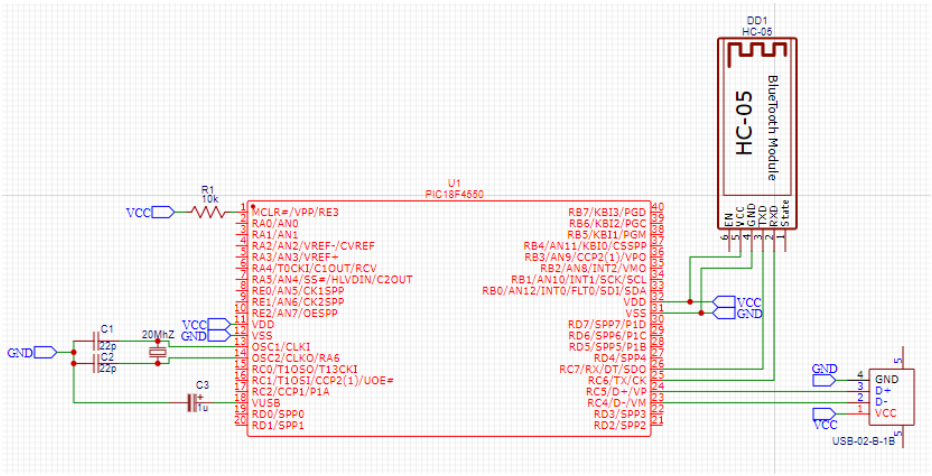


Figura 51. Montaje y aplicación programa Módulo EUSART. Fuente propia

5.2.8 Módulo MSSP

Este módulo permite realizar comunicación serial SPI (Serial Peripheral Interface) y I2C (Inter-Integrated Circuit) ya sea en modo maestro o esclavo. Siendo así, se propone para este programa el uso de la comunicación USB-CDC® para programar un RTC (Real Time Clock) por medio de la comunicación I2C ejecutada desde el microcontrolador. Véase Figura 52

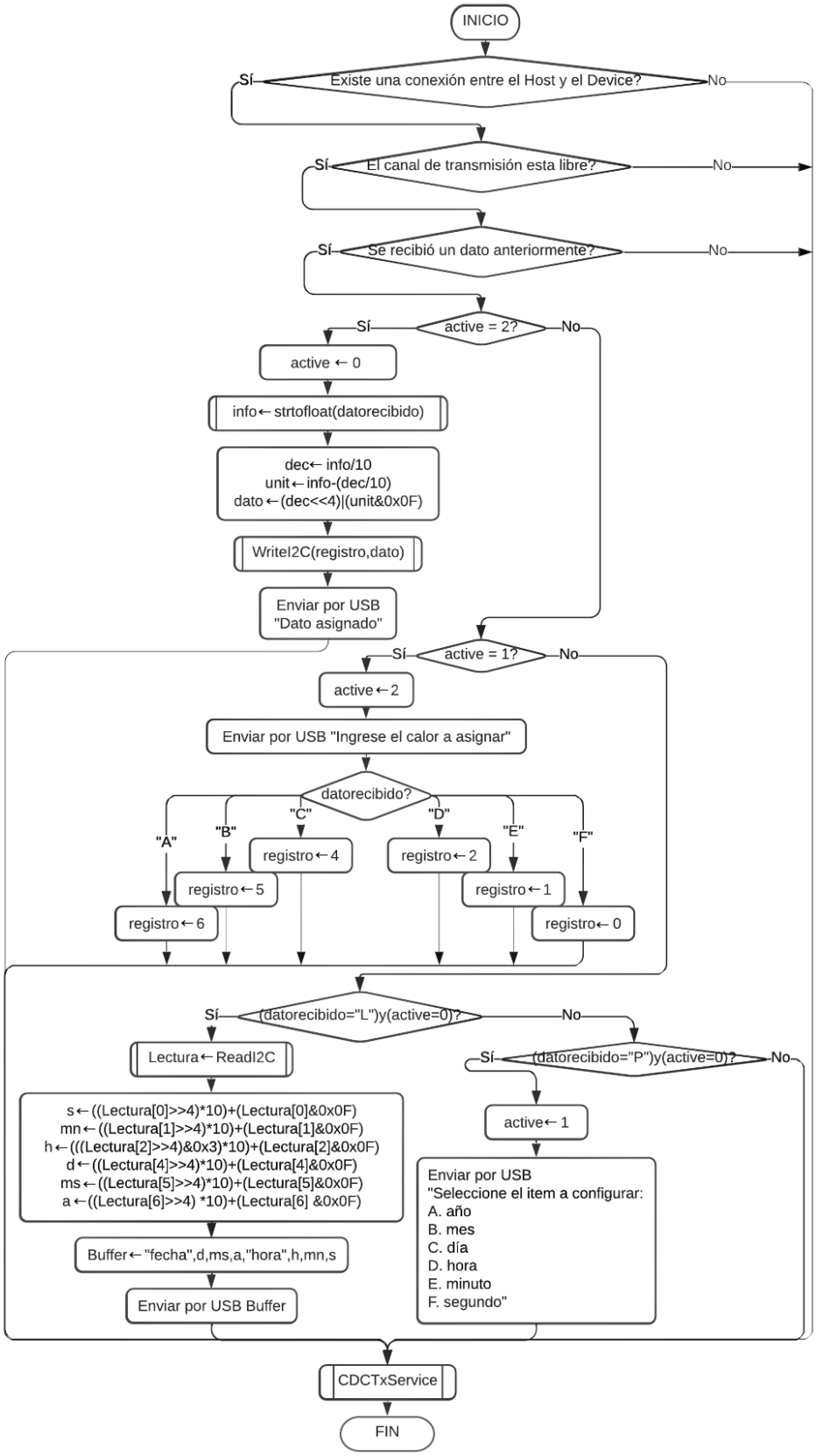


Figura 52. Función USBTask Módulo MSSP. Fuente propia

Para su correcto funcionamiento se implementa el circuito mostrado en la Figura 53 donde se observa la conexión del RTC con los pines RB0 y RB1 del microcontrolador correspondientes a la comunicación I2C.

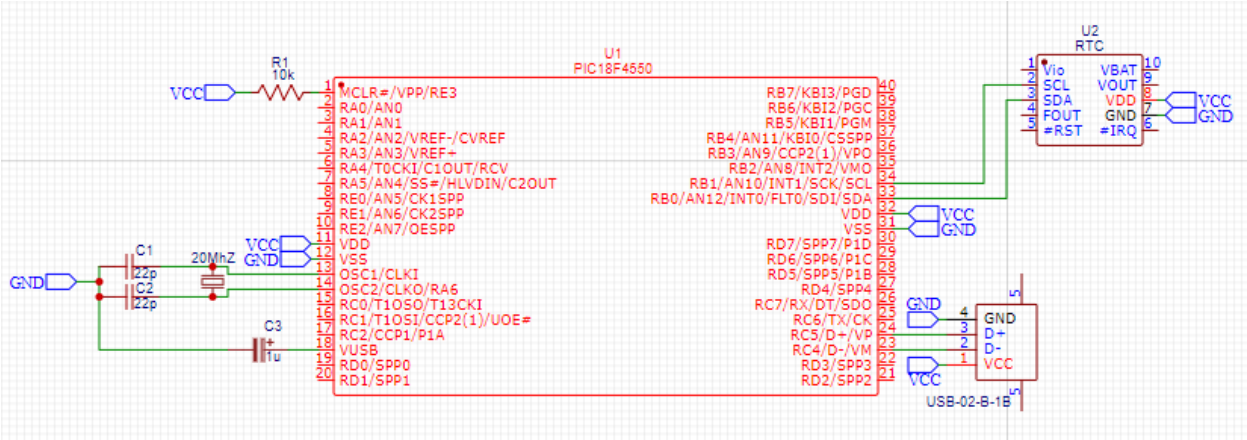


Figura 53. Montaje programa Módulo SSP. Fuente propia

5.2.9 Módulo EEPROM

Este módulo permite almacenar información en la memoria EEPROM del microcontrolador, la cual se conserva aun cuando esta desconectado. Por lo tanto, el programa propuesto permite la interacción con esta memoria ingresando la información desde el computador, Véase Figura 54.

Para el funcionamiento de este programa se requiere de dos funciones específicas, lectura y escritura, que son las que realizan la interacción con la memoria EEPROM. Para realizar la lectura se utiliza la función LeerEEPROM, véase Figura 55, que lee un registro especifico, mientras para escribir se utiliza la función EscribirEEPROM, véase Figura 56, que permite escribir en un registro determinado por el usuario. De igual manera para su implementación se utiliza el circuito utilizado en el programa base, véase Figura 27.

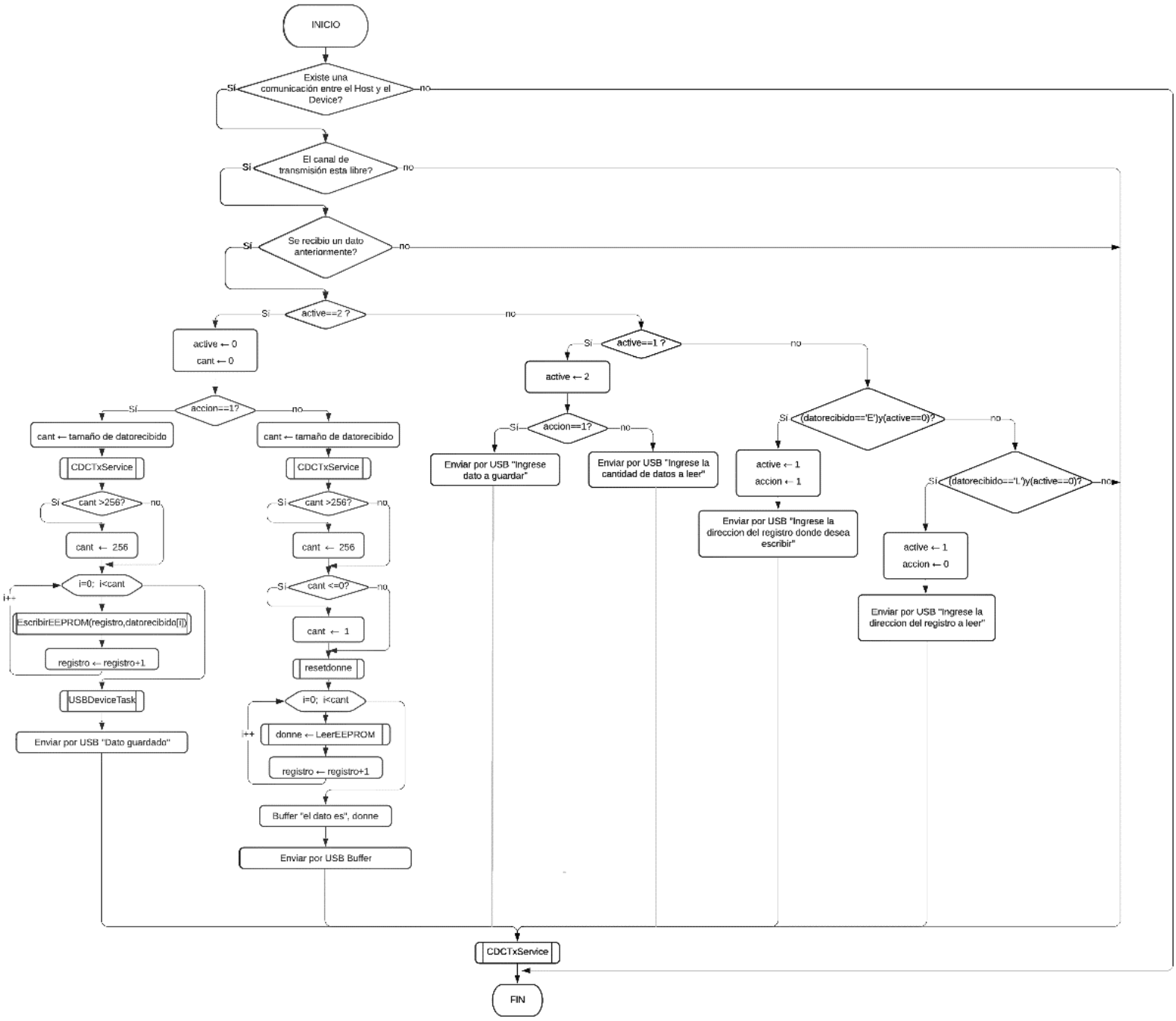


Figura 54. Función USBTask Módulo EEPROM. Fuente propia

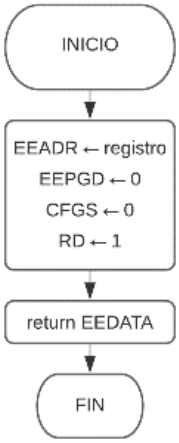


Figura 55. Función LeerEEPROM. Fuente propia

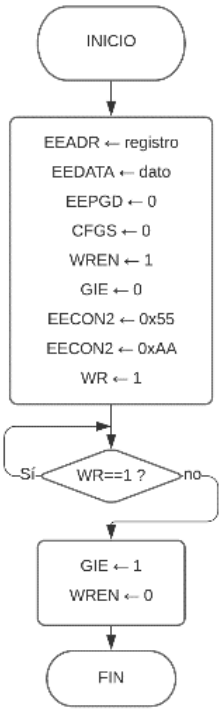


Figura 56. Función EscribirEEPROM. Fuente propia

5.2.10 Programa de comunicación USB-CDC® por interrupción

Viendo el alcance y las desventajas del programa realizado inicialmente, se propone cambiar el modo de operación de POLLING a INTERRUPT lo que permitirá una mayor estabilidad para este, siendo así se genera otro programa con el mismo funcionamiento del programa inicial, pero variando ligeramente su estructura, véase Figura 57, y con una nueva función de interrupción, véase Figura 58.

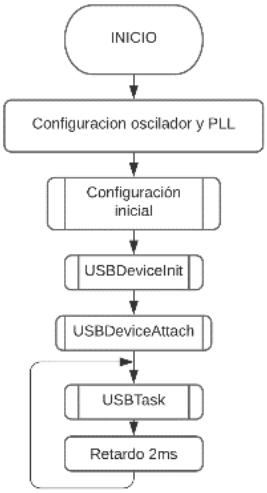


Figura 57. Programa base por interrupción. Fuente propia

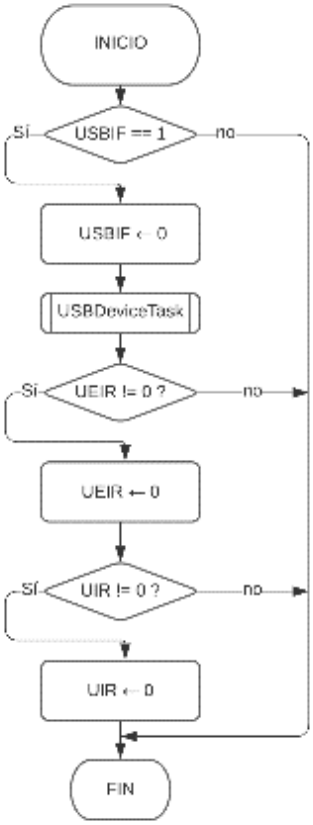


Figura 58. Interrupción USB. Fuente propia

5.3 Diseño de tarjeta PCB

Para complementar el proyecto y permitir un mejor funcionamiento de los ejemplos expuestos anteriormente se realiza el diseño de una tarjeta PCB usando el Software Proteus en la cual exista:

- La interacción con todos los pines disponibles de microcontrolador
- Estabilidad del conector USB® tipo B y del cristal
- Un led indicador de Voltaje y un led indicador de funcionamiento (RC0)
- Puerto de programación para Pickit3
- Condensadores para regular el voltaje de alimentación y condensador para el pin V_{USB}
- Puerto para interacción con LCD 16x2
- Pulsador de Reset (MCLR)
- Pines de alimentación (Vcc y GND)

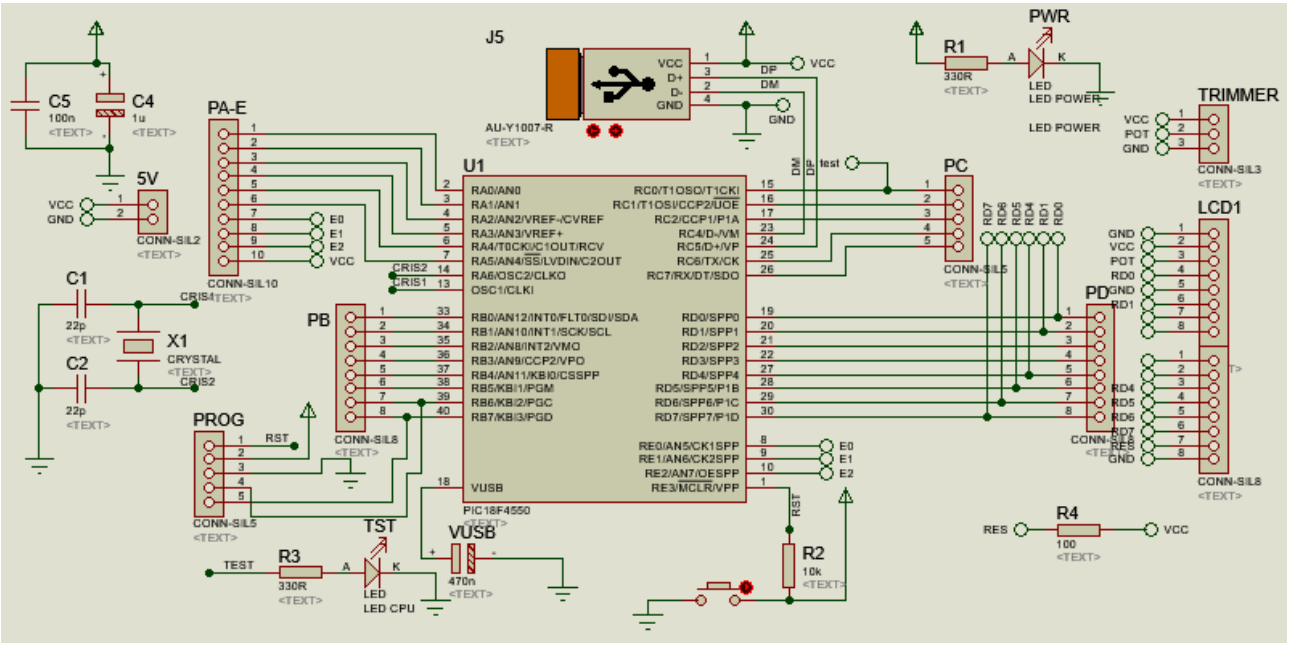


Figura 59. Base del diseño de la tarjeta PCB. Fuente propia

5.4 Ejercicios de aplicación

Con el fin de mostrar las aplicaciones de la comunicación USB-CDC® en la ingeniería se presentan tres programas que utilizan este protocolo de comunicación en las áreas de instrumentación y control.

5.3.1 Componentes electrónicos

Antes de iniciar con los ejercicios de aplicación es necesario conocer el funcionamiento y uso de tres dispositivos que se van a utilizar.

5.3.1.1 Sensor de temperatura y humedad DHT11

Este sensor cuenta con cuatro pines de conexión, dos de alimentación (3V a 5.5V) y uno de datos el cual se conecta a una resistencia *pull-up* para su correcto funcionamiento, véase Figura 60.

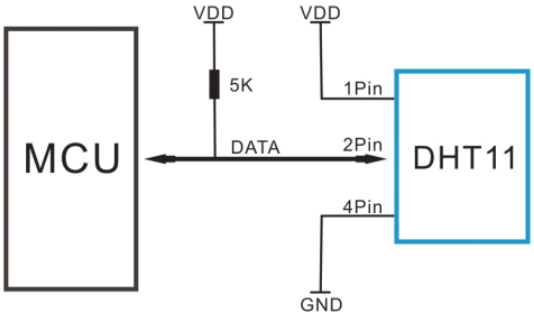


Figura 60. Conexión DHT11. Tomada de www.mouser.com

Para realizar la comunicación con este dispositivo se requiere enviar una señal hacia el sensor que indique el inicio de la comunicación, en respuesta el DHT11 envía una señal de 40 bits que incluye los datos de temperatura y humedad. Finalmente, se debe enviar otra señal que indique el recibido de la información e indique al sensor que debe volver al modo de espera, véase Figura 61.

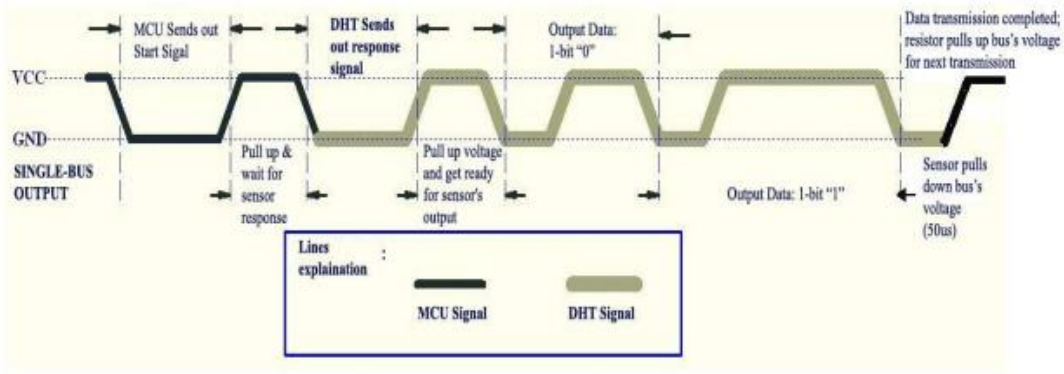


Figura 61. Proceso de comunicación con DHT11. Tomada de www.mouser.com

Al iniciar la comunicación se debe cambiar el nivel de voltaje del bus de alto a bajo en un tiempo de aproximadamente 18ms y en respuesta se recibirán pulsos de un tiempo entre 20us (0 lógico) y 40us (1 lógico) [64]. Teniendo en cuenta lo mencionado anteriormente, se realiza una función que permita la lectura de este sensor utilizando el microcontrolador PIC18F4550, véase Figura 62.

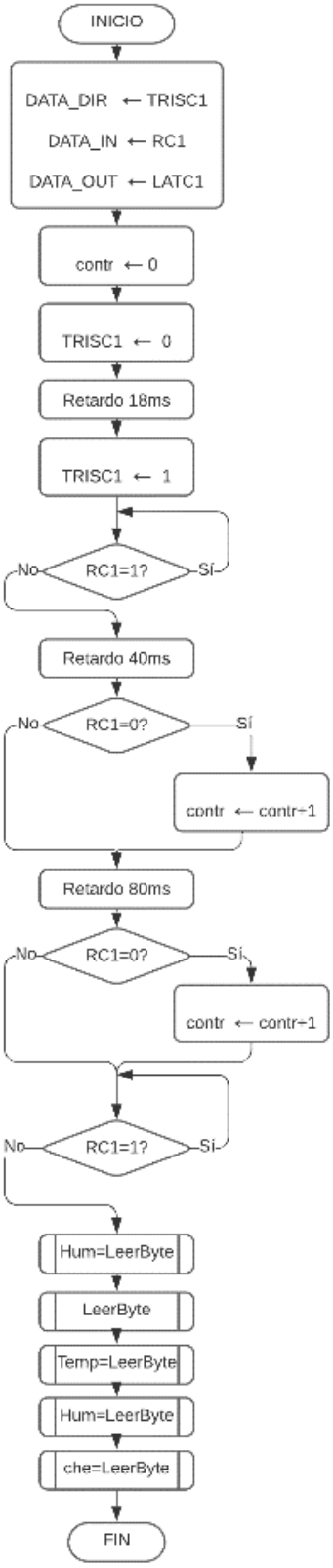


Figura 62. Función de lectura del sensor DHT11. Fuente propia

5.3.1.2 Relé de estado solido

La conexión de este dispositivo se basa en tres terminales de entrada, dos de alimentación DC y uno que envía la señal de switch, y dos terminales de salida que funcionan como interruptor para la carga AC [65], véase Figura 63.

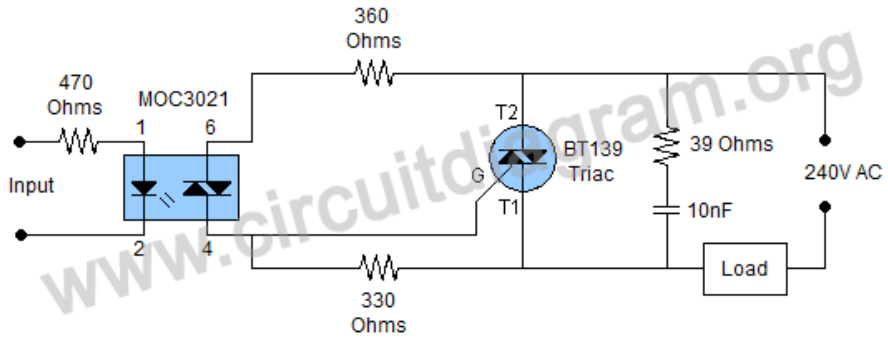


Figura 63. Diagrama de relé de estado sólido. Tomada de http://www.circuitdiagram.org

Según lo anterior se deduce que para utilizar este dispositivo se requiere únicamente de un pin digital del microcontrolador que cambie de estado dependiendo de la necesidad que se presente.

5.3.1.3 Dimmer digital

Este dispositivo cuenta con seis conexiones DC entre las que se encuentran dos pines de alimentación (+5V y GND), un pin SYNC por donde envía una señal de cruce por cero con el cual se tiene un control de inicio y fin de la señal análoga, y un pin CH1 que sirve para controlar la activación del triac interno que se encarga de permitir o interrumpir el paso de la señal análoga; también cuenta con cuatro conexiones AC dos para la carga y dos para la alimentación AC (110V a 220V) [66], véase Figura 64.

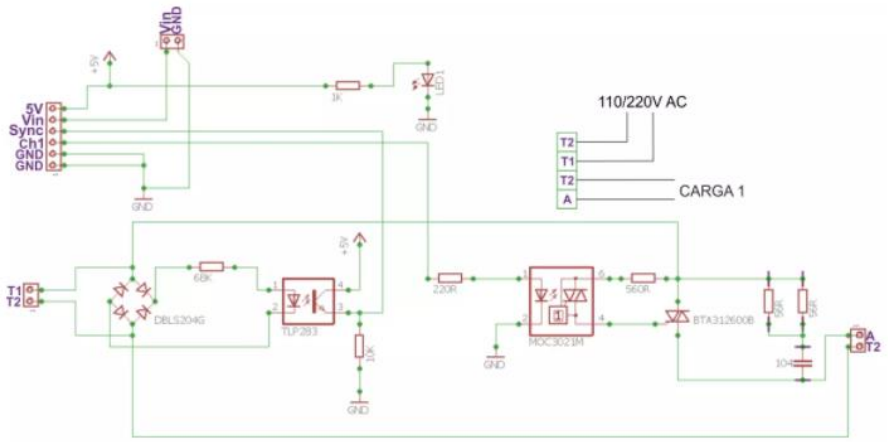


Figura 64. Diagrama Dimmer digital. Tomada de www.vistronica.com

Para la implementación de este dispositivo se realiza un programa de prueba donde se presenta una interrupción de entradas y salidas que se activa por la señal del pin SYNC (cruce por cero), véase Figura 65, en este se cuentan dos ciclos para determinar el inicio ya que se tiene un cruce por cero para el cambio entre la señal positiva y negativa y otro para la acción contraria.

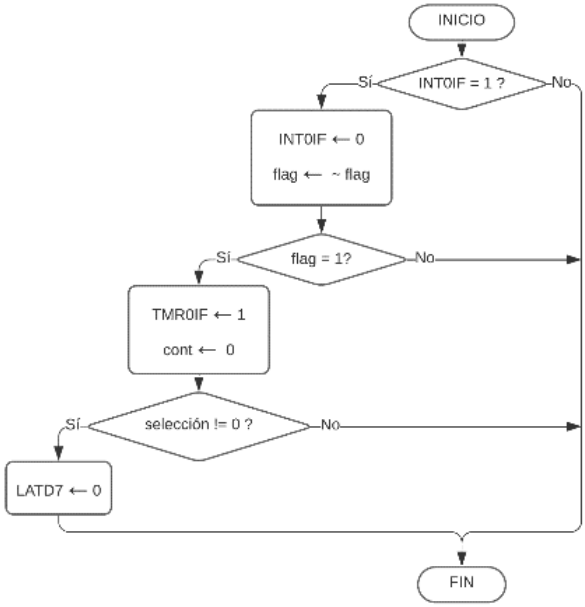


Figura 65. Función de interrupción para Dimmer digital. Fuente propia

Además, se realiza una interrupción de TIMER0 llevando un periodo de 8,33ms para subdividir el periodo de la señal análoga en 20 partes y así obtener 20 intensidades de luz diferentes, véase Figura 66. Estas intensidades serán controladas por el valor de la variable "selección" que tomará un valor entre 0 y 19 según se establezca en el programa principal.

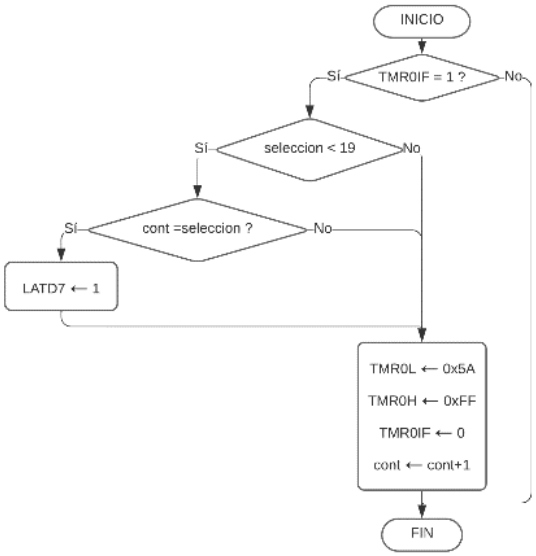


Figura 66. Interrupción de TIMER0 para Dimmer digital. Fuente propia

5.3.2 Ejercicio de Instrumentación

En este ejercicio se propone el uso de una caja cerrada con un bombillo en el interior, el cual se encarga de calentar el entorno, este se encenderá por medio de un interruptor que es controlado por el usuario. En el interior de la caja se tiene un sensor DHT11 que mide la temperatura y humedad del ambiente y es controlado por el microcontrolador que, a su vez, utilizando la comunicación USB-CDC®, envía los datos recolectados hacia el computador, véase Figura 67.

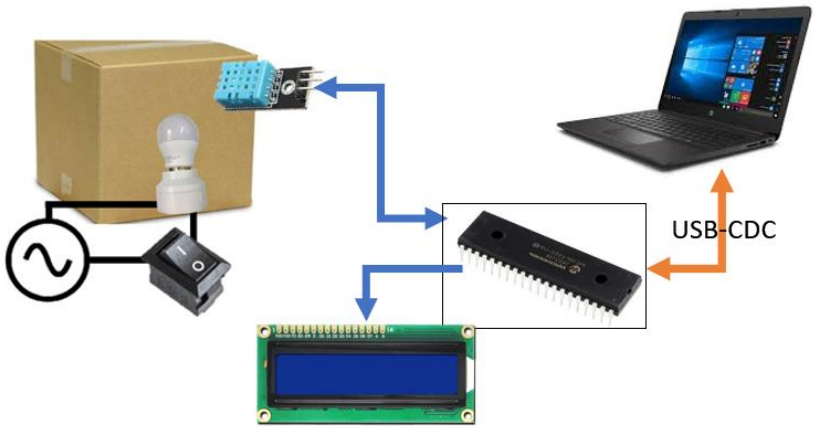


Figura 67. Estructura ejercicio de Instrumentación. Fuente propia

Para el desarrollo de este ejercicio se parte del programa de comunicación USB-CDC® por interrupciones realizado anteriormente y se incluye la función mediante la cual se lee el sensor, véase Figura 62. Igualmente se implementa una interrupción de TIMER0 para llevar un control de la frecuencia con la cual se envía el dato hacia el computador y a la LCD, véase Figura 68.

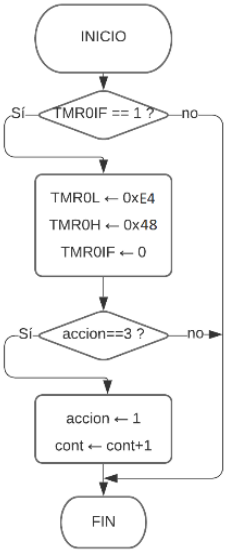


Figura 68. Interrupción del TIMER0. Fuente propia

En el programa principal se tienen las configuraciones iniciales tanto de la comunicación USB® como de la LCD, además, de la lectura del sensor. Además, se incluye la función USBTask del programa donde se realizan las funciones correspondientes a la comunicación con el computador, entre las cuales se encuentra el inicio de la toma de muestras, la selección de escala de temperatura y la finalización de la toma de muestras, véase Figura 69 y Figura 70

Figura 69. Función USBTask programa de Instrumentación. Fuente propia

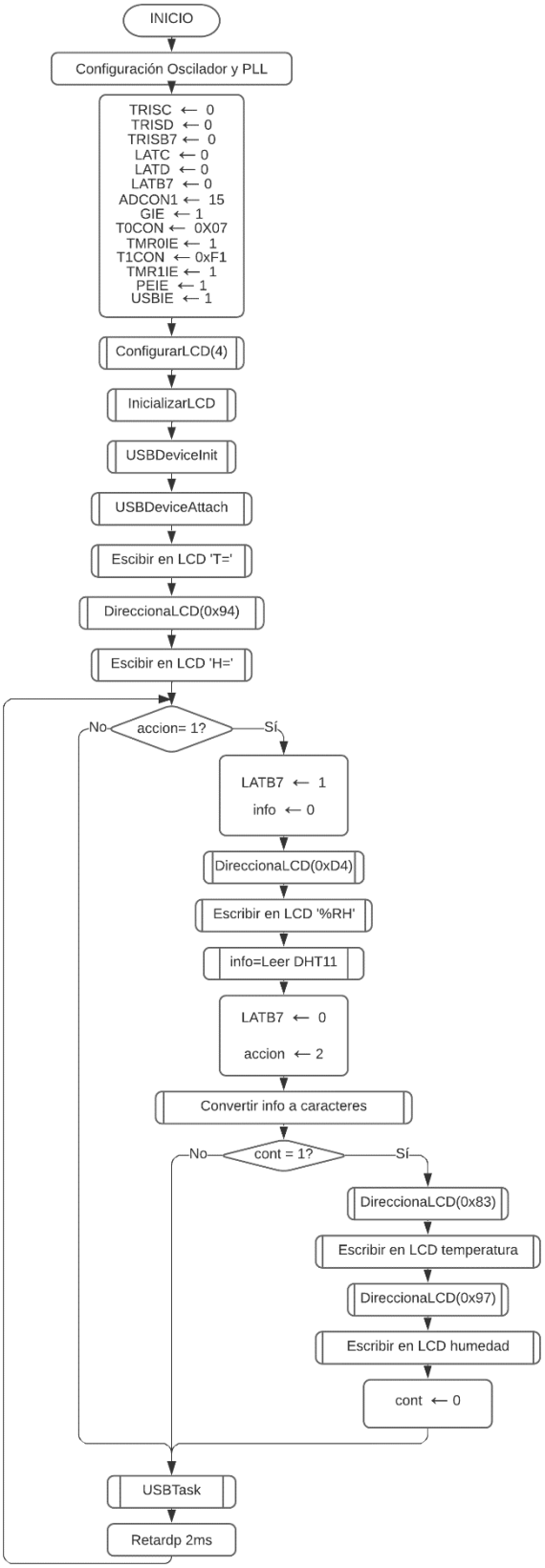


Figura 70. Programa PIC® ejercicio de instrumentación. Fuente propia

5.3.1.1 Software

Para este ejercicio se realiza una interfaz gráfica utilizando el GUI de Matlab, véase Figura 71, donde se tiene una serie de opciones que permiten el cambio de escala de temperatura, el inicio y la finalización de toma de muestras y la opción de salida de la interfaz además de su propósito de graficar los valores leídos por el sensor.

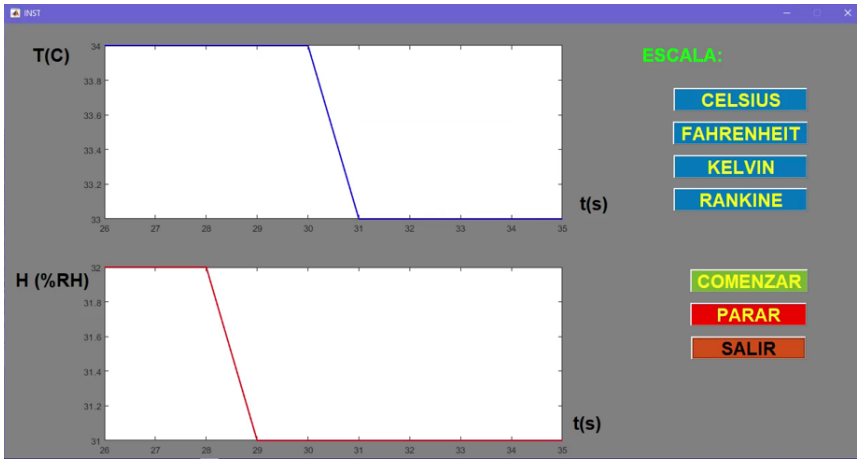


Figura 71. Interfaz gráfica en Matlab. Fuente propia

Este programa los botones que cambian la escala de temperatura se encargan de enviar el carácter correspondiente a la temperatura deseada programada en el microcontrolador y a su vez reinician la grafica tanto de la temperatura como de la humedad, véase Figura 72.

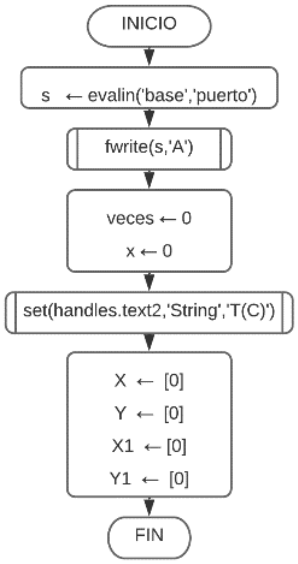


Figura 72. Programa Botón para cambio de temperatura a Celsius. Fuente propia

De igual manera, los botones de Comenzar y Parar envian al microcontrolador el carácter correspondiente a dicha función además de guardar la información recompilada en un archivo de excel, véase Figura 73.

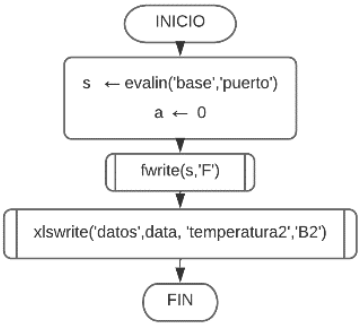


Figura 73. Programa de botón PARAR. Fuente propia

Este programa tiene una función de interrupción que se encarga de leer los datos recibidos y graficarlos utilizando la función plot, véase Figura 74.

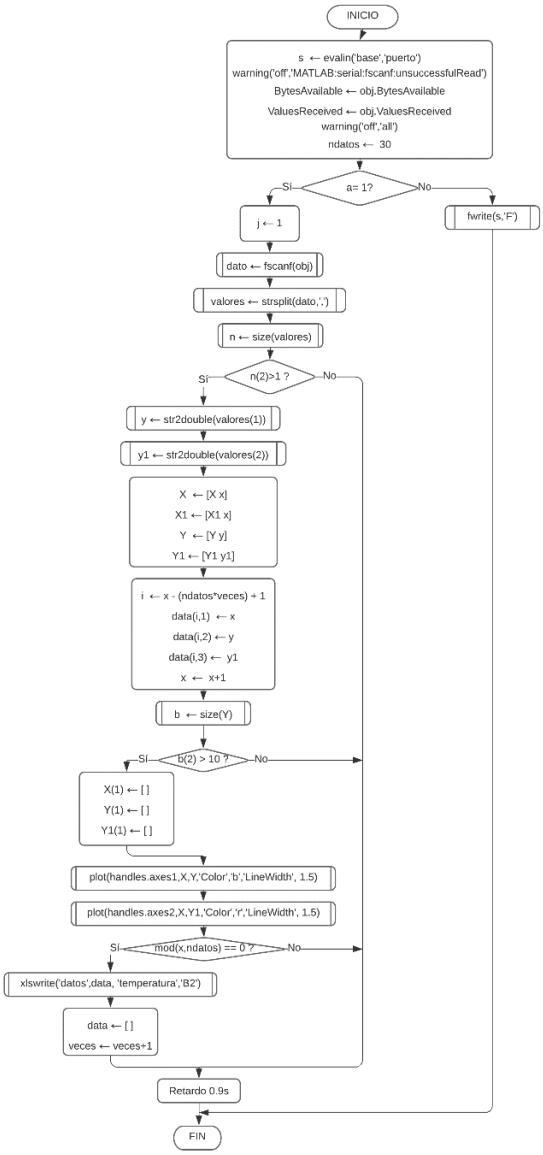


Figura 74. Función de recepción de datos. Fuente propia

Finalmente el boton de Salir cierra la interfaz a la vez que deshace la conexión en tre el Software y el microcontrolador, véase Figura 75.

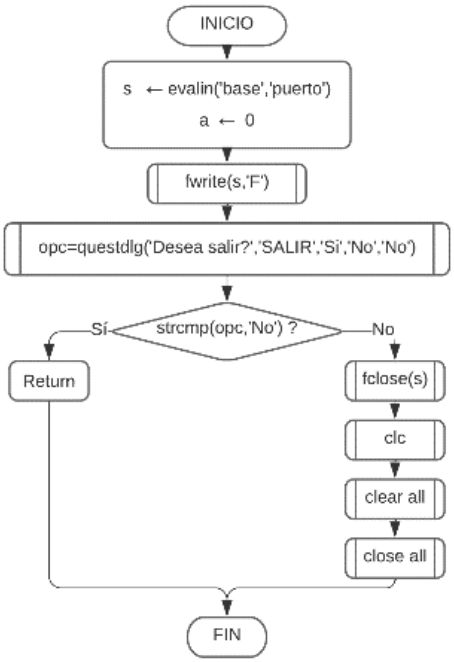


Figura 75. Programa de botón SALIR. Fuente propia

5.3.3 Ejercicio de Control ON-OFF

Para este ejercicio se plantea un primer acercamiento a un programa de control utilizando un control ON OFF. Por lo tanto, se plantea un entorno similar al anterior en el cual se tiene una caja sellada con un bombillo en su interior el cual se enciende o apaga mediante un relé de estado sólido controlado por medio del microcontrolador dependiendo de la temperatura leída con el sensor DHT11 y una temperatura de referencia que puede ser ingresada de tres maneras diferentes:

- Estática: Se establece una temperatura de referencia que no puede cambiarse
- Potenciómetro: La referencia se determina según la posición de un potenciómetro, esta puede ser entre 20°C y 40°C
- Software: Esta se establece por medio de la comunicación USB-CDC®

En este caso, a diferencia de los anteriores, se utiliza la Raspberry Pi para hacer la comunicación con el microcontrolador, véase Figura 76.

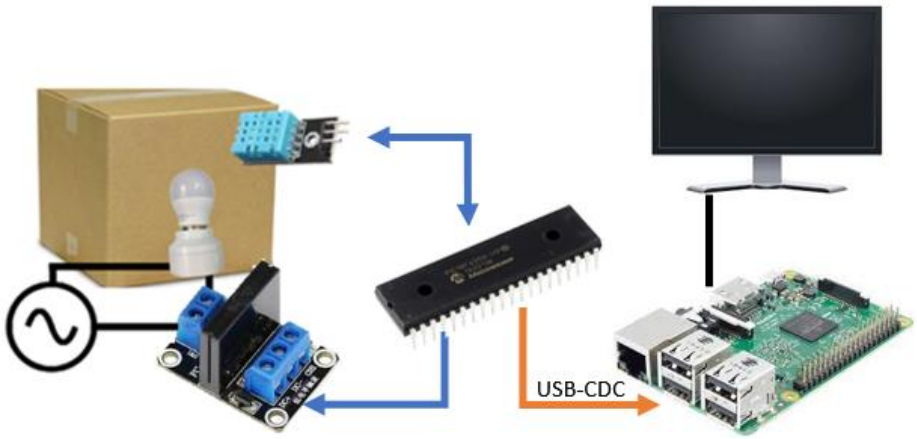


Figura 76. Estructura ejercicio de Control ON OFF. Fuente propia

Ya que se plantean tres formas de leer la resistencia se plantean tres programas diferentes para el desarrollo de este ejercicio teniendo en cuenta la entrada de referencia. Para el caso de la referencia estática únicamente se asigna el valor deseado en la variable "ref". Sin embargo, para los otros dos casos se utilizan funciones específicas que permiten identificarla. La referencia por potenciómetro se asigna mediante el módulo ADC del microcontrolador, véase Figura 77; mientras la referencia por software se asigna en la función USBTask, véase Figura 78.

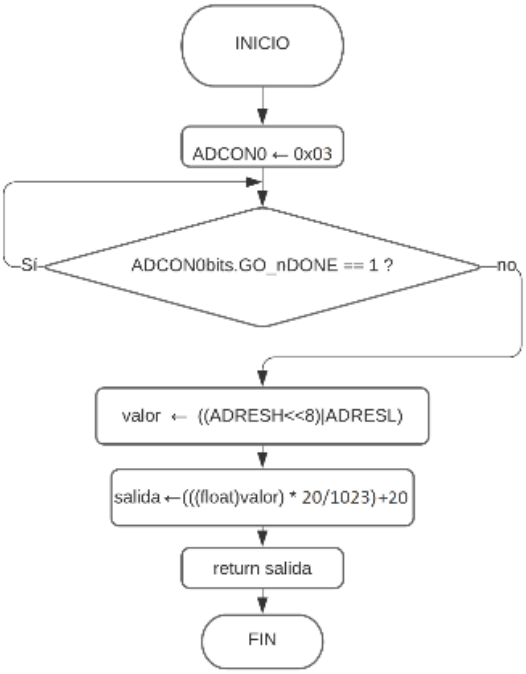


Figura 77. Función de referencia por potenciómetro. Fuente propia

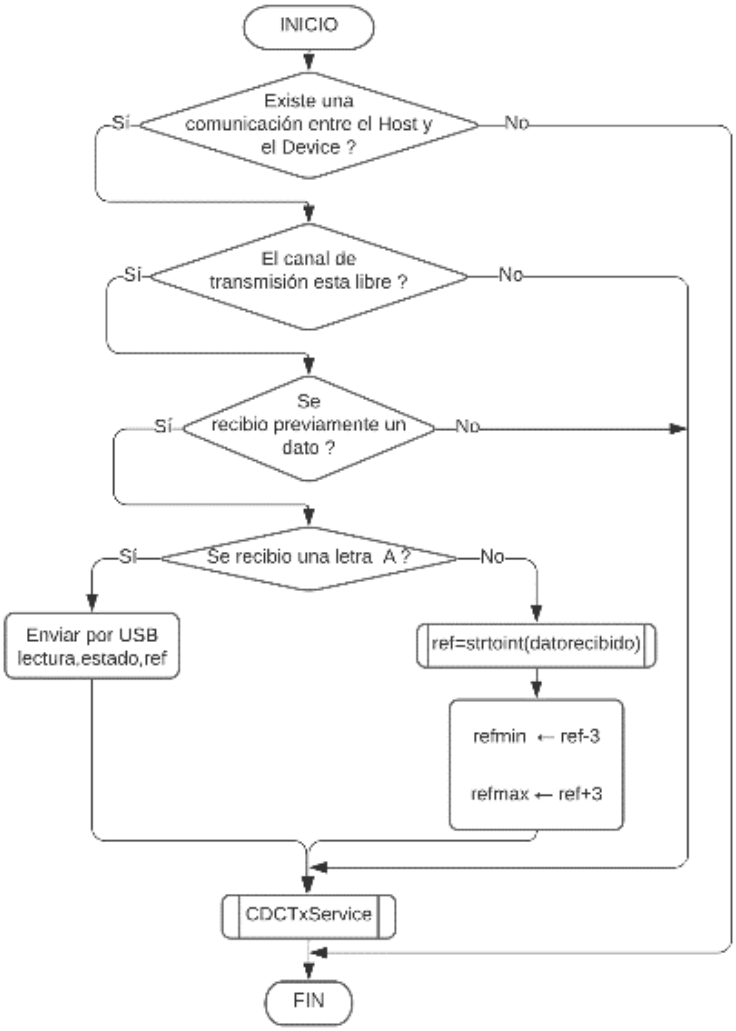


Figura 78. Función USBTask para programa de control on off. Fuente propia

Para el correcto funcionamiento del programa, se utiliza la comparación de dicha referencia con un rango de operación de ±3 que permite una mayor estabilidad del sistema y dependiendo de esto se encenderá o apagará el bombillo. Cabe aclarar que este proceso se realiza cada segundo para tener mayor estabilidad y una mejor respuesta del sistema. Por otro lado, se tiene en cuenta que para cada tipo de referencia se tiene un programa diferente aun así la estructura es similar y se muestra en la Figura 79.

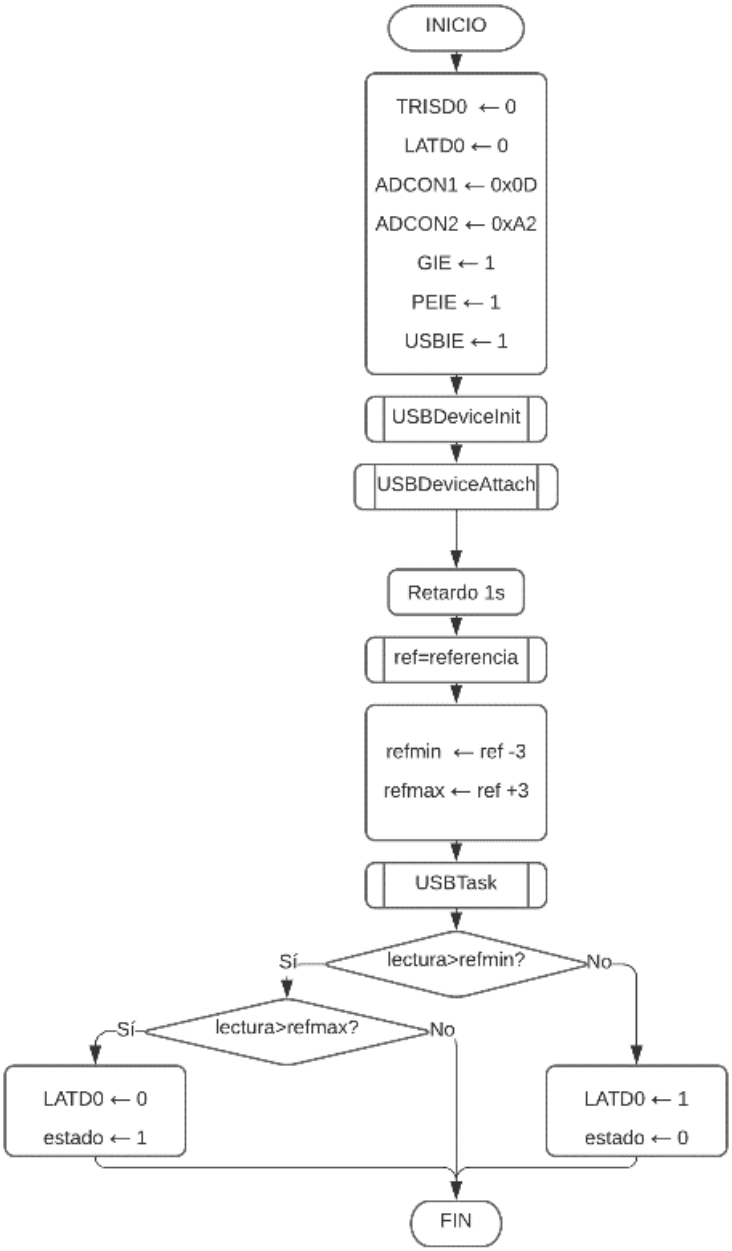


Figura 79. Programa Control ON OFF. Fuente propia

5.3.3.1 Software

Como se mencionó anteriormente, para este programa se utiliza la Raspberry Pi, por lo tanto, el software a usar es Python con el cual se realiza una interfaz gráfica por medio de la librería Tkinter, véase Figura 80. Igualmente, se usan otras librerías como lo son serial para la comunicación con el PIC®, numPy para el uso de vectores, math para las operaciones matemáticas y time para funciones relacionadas con el tiempo.

Figura 80. Interfaz para programa Control ON OFF. Fuente propia

Esta interfaz incluye un espacio donde se muestra el dato de lectura de la temperatura actual, un espacio para la escritura de la referencia deseada en caso de que este sea el método de entrada del programa seguido por un cuadro de texto que muestra la referencia actual, además de dos botones que permiten enviar la referencia digitada al microcontrolador y salir del programa y finalmente un indicador para saber cuándo está encendido el bombillo. Los datos de temperatura y referencia actual se actualizan en cada siclo del programa al igual que el indicador de estado, véase la Figura 81. Por otro lado, para el funcionamiento correcto del botón de envió y de salida se tienen dos funciones independientes que funcionan únicamente al presionar el botón especifico, véase Figura 82 y Figura 83.

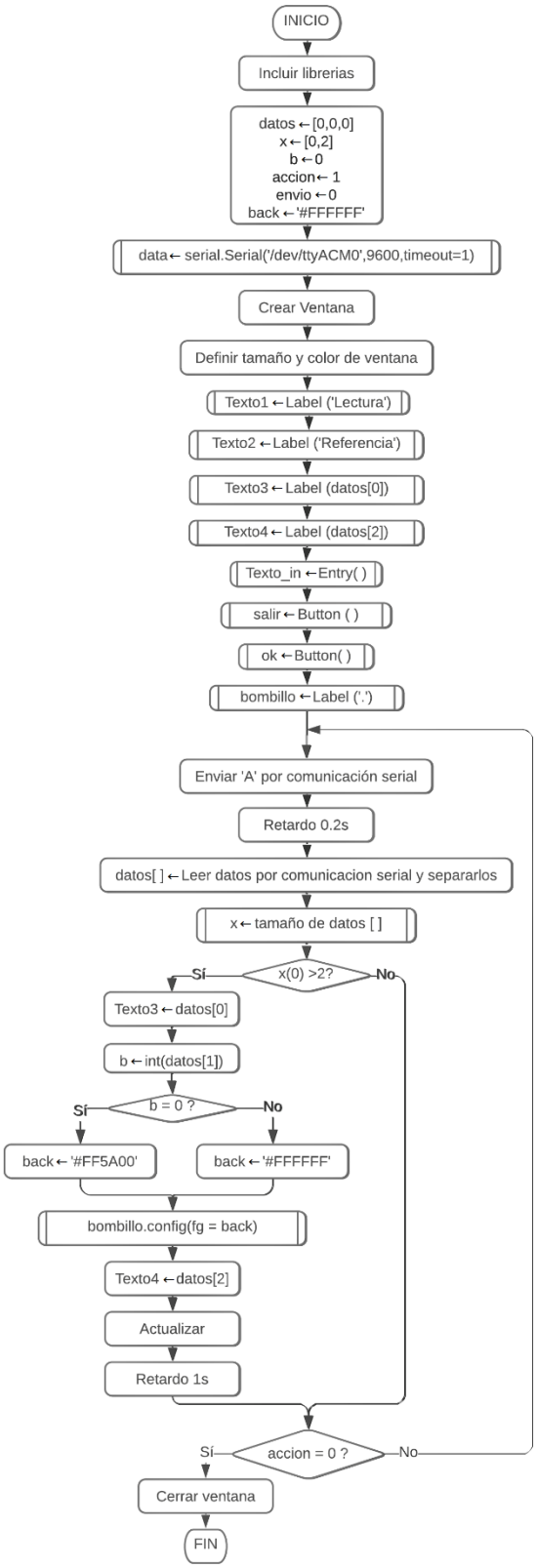


Figura 81. Programa Interfaz Python Control ON OFF. Fuente propia

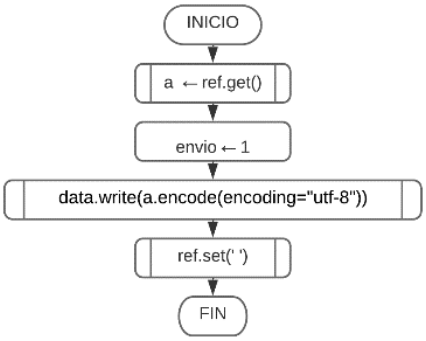


Figura 82. Función de botón Enviar. Fuente propia

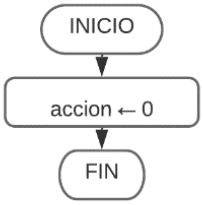


Figura 83. Función de botón Salir. Fuente propia

5.3.4 Ejercicio de Control Difuso

Para este último ejercicio se propone la misma situación del ejercicio anterior con la diferencia del método para encender el bombillo. En este caso el bombillo podrá tomar 20 diferentes intensidades para lo cual se utiliza el Dimmer digital, véase Figura 84.

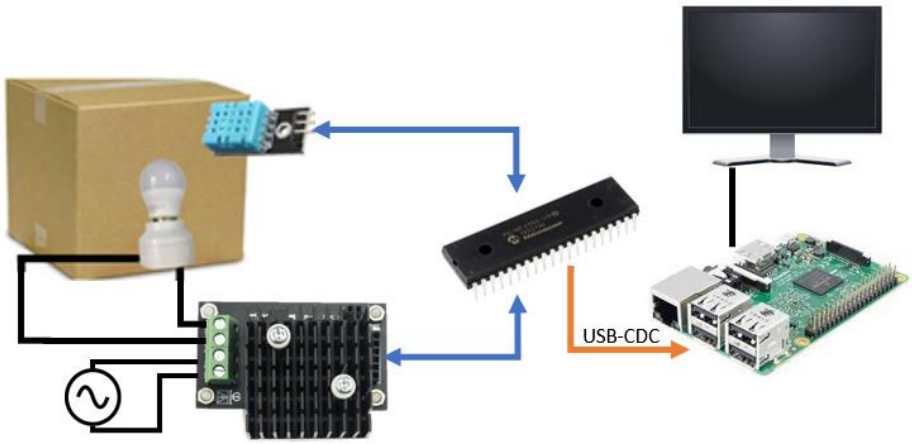


Figura 84.Estructura ejercicio de Control difuso. Fuente propia

Inicialmente se plantea un control difuso en el cual su única entrada será el valor leído por el sensor de temperatura que una vez implementado el control generará una salida

correspondiente a la intensidad necesaria en el bombillo para llegar a la referencia, para esto se plantea el programa de la Figura 85.

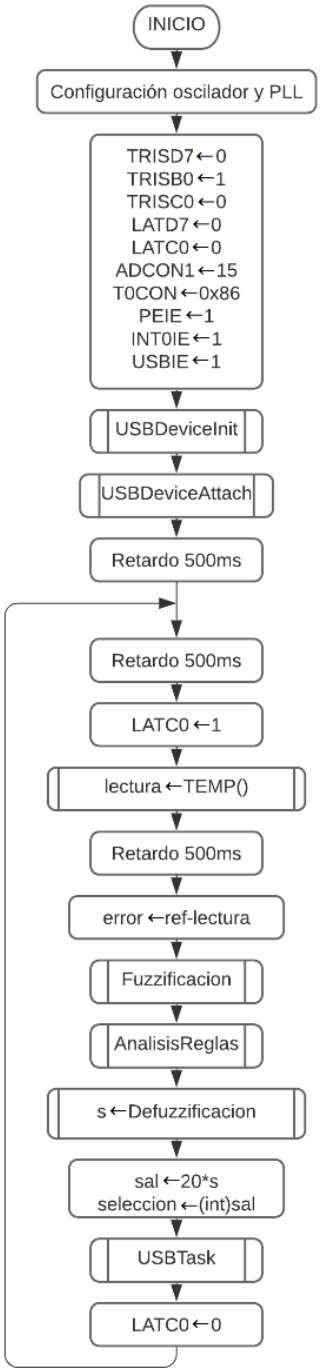


Figura 85. Programa de control difuso. Fuente propia

En este programa la función de Fuzzificación está dada por las funciones mostradas en la Figura 86 donde función N corresponde a un valor negativo del error, la función Z a un valor cero y la función P a un valor positivo. Teniendo en cuenta estas funciones, en el análisis de reglas se hace una asignación a cada una de las tres reglas establecidas

para cada una de las tres funciones mencionadas anteriormente. Finalmente, en la Defuzzificación se da un valor correspondiente a la acción de control que se realizará según la temperatura evaluada, véase Figura 87.

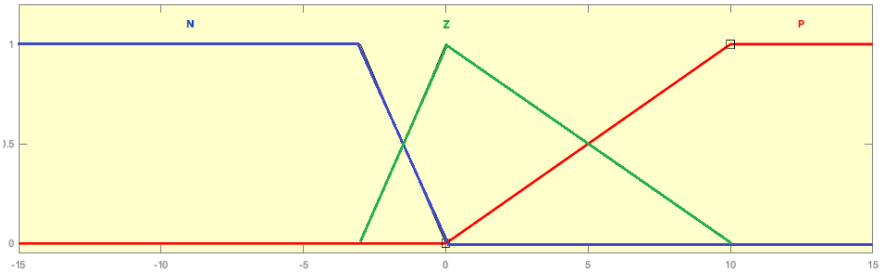


Figura 86. Función de Fuzzificación. Fuente propia

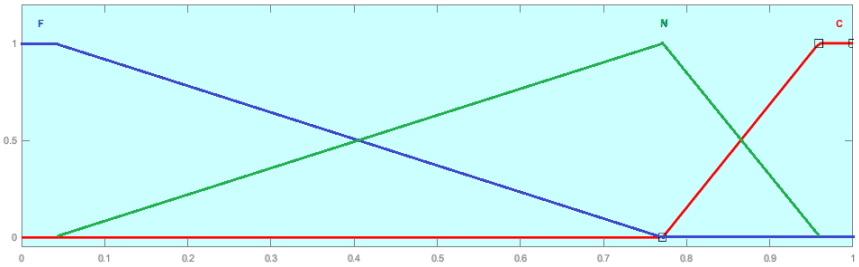


Figura 87. Función de Defuzzificación. Fuente propia

Por último, la función USBTask realiza la comunicación con la Raspberry para enviar la información requerida para la interfaz de usuario y recibir de igual manera la referencia de la temperatura. Esta función tiene la misma estructura que la utilizada anteriormente el ejercicio de Control On Off, véase Figura 78.

Seguidamente, se realiza una adaptación al programa para obtener una mayor precisión teniendo como variables de entrada la temperatura y su derivada, Ecuación 4, teniendo un tiempo de muestreo de 1 segundo.

$$dE = \frac{e_{actual} - e_{anterior}}{t} \qquad (4)$$

Para la función de Fuzzificación se utilizan las funciones mostradas en la Figura 88 y Figura 89 donde N indica un error negativo, Z un error cero y P un error positivo; de igual manera, dN indica una derivada del error negativo, dZ una derivada cero y dP una derivada positiva.

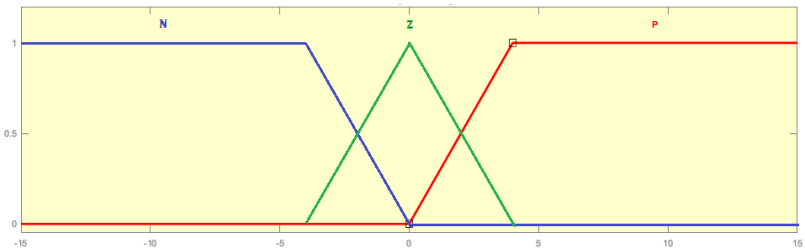


Figura 88. Entrada de error. Fuente propia

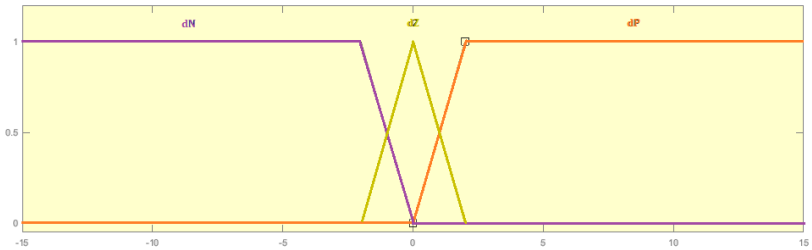


Figura 89. Derivada del error. Fuente propia

Teniendo en cuenta lo anterior para el análisis de reglas se aplica la Tabla 4, donde según el valor indicado del error y la derivada se aplicará una de las reglas establecidas, teniendo nueve posibles opciones. Finalmente, la Defuzzificación se aplica bajo las mismas funciones mostradas para el programa anterior, véase Figura 87.

		DERIVADA		
		N	Z	P
ERROR	N	N	N	Z
	Z	N	Z	P
	P	Z	P	P

Tabla 4. Análisis de reglas para control difuso de dos entradas. Fuente propia

5.3.4.1 Software

Para la interfaz de este programa se toma como referencia la interfaz anterior cambiando el método del indicador de encendido del bombillo, véase Figura 90. Donde a partir de las barras se muestra la intensidad con la cual se enciende.

Figura 90. Interfaz control difuso. Fuente propia

Para el desarrollo de esta interfaz se tiene un programa similar al anterior en el cual se añade la definición de las barras y un procedimiento que define el color de cada una de ellas teniendo en cuenta 4 tonos (blanco, amarillo, naranja y rojo), véase Figura 91.

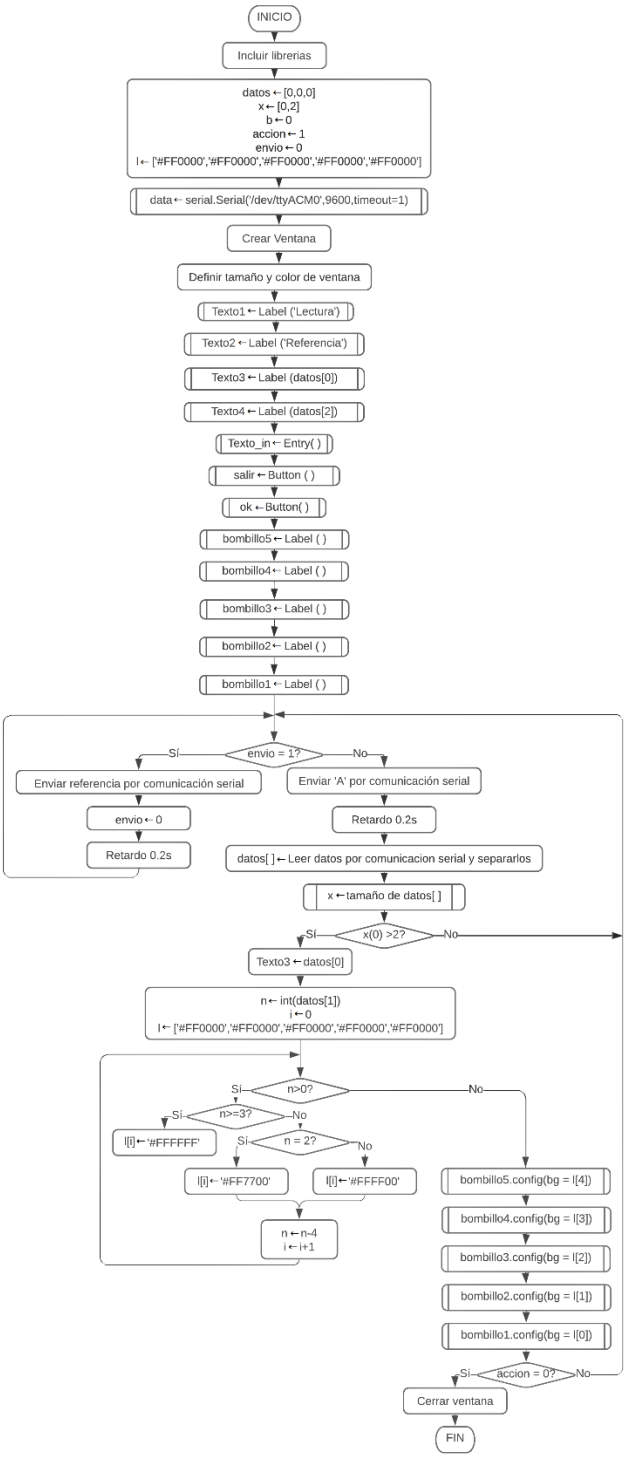


Figura 91. Programa Interfaz Python Control difuso. Fuente propia

6. Resultados

Inicialmente, se obtiene un programa ejemplo con el microcontrolador PIC18F45K50 generado mediante el MPLAB® Code Configurator, de manera tal que haciendo uso de las librerías generadas para el módulo USB-CDC® y tomando como ejemplo el programa implementado en C18® se obtiene el programa base para este proyecto el cual utiliza un puerto COM para realizar la comunicación, véase Figura 92. A partir de este programa se comprende la estructura y funcionamiento de las diferentes funciones aplicadas.

Figura 92. Funcionamiento programa base. Fuente propia

El módulo TIMER tuvo un funcionamiento optimo contabilizando el tiempo de conexión transcurrido con una gran exactitud además de mostrarlo por pantalla, véase Figura 93.

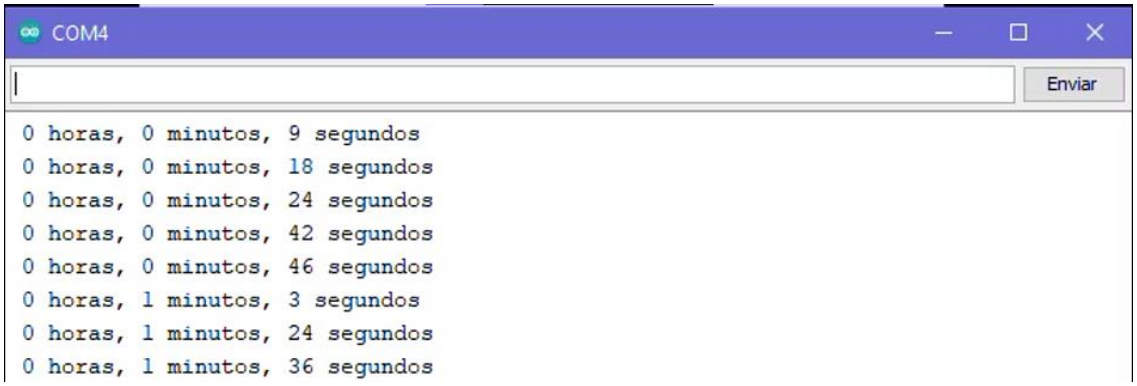


Figura 93. Resultado programa con TIMER0. Fuente propia

El programa con el módulo de entradas y salidas (I/O) tuvo un buen resultado ya que realizó todos los procedimientos propuestos inicialmente mostrando lo indicado en el teclado por pantalla, véase Figura 94, además de tener una perfecta interacción con los Leds controlados desde el computador.

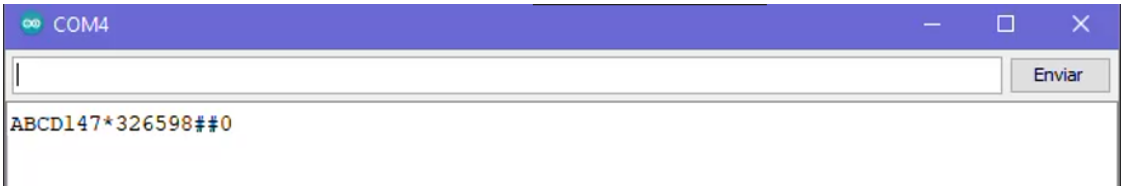


Figura 94. Resultado programa con módulo de entradas y salidas. Fuente propia

Para el módulo ADC (Conversor análogo-digital) se tiene como resultado una buena resolución de los datos al usar 10 bits en la conversión. Sin embargo, debido al tiempo de conectividad del protocolo, la frecuencia de recolección de datos que se presenta es elevada, véase Figura 95.

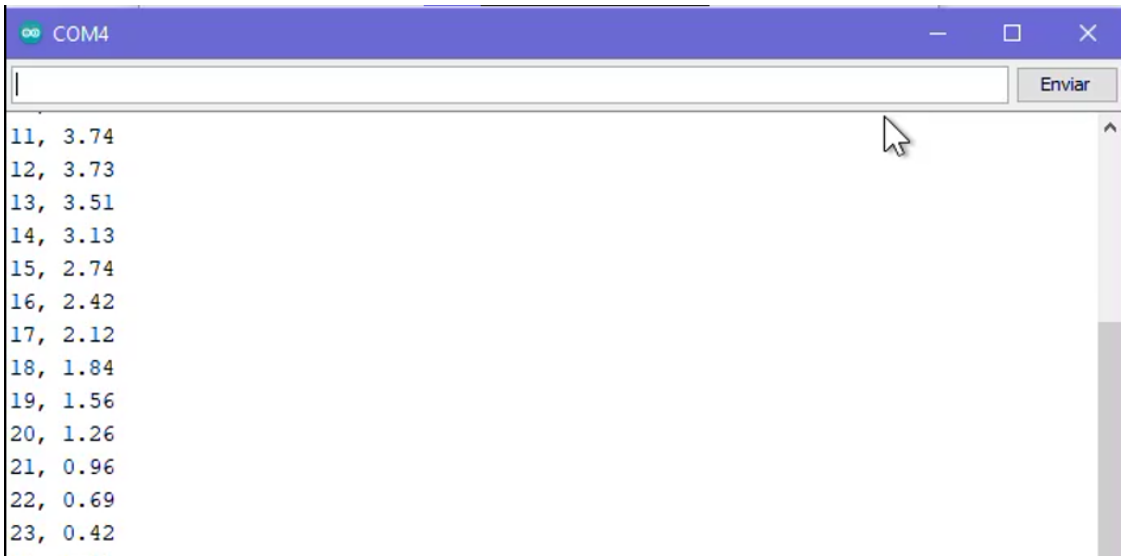


Figura 95. Resultado programa con módulo ADC. Fuente propia

Al implementar el módulo de referencia no se obtienen los resultados esperados frente a los valores de voltaje generados ya que no son los calculados, aun así, el intervalo presentado entre los mismos corresponde al establecido por el Datasheet del microcontrolador.

Por otro lado, el módulo comparador permitió interactuar y mostrar el funcionamiento de los comparadores internos del microcontrolador visualizando su funcionamiento por la pantalla del computador y permitiendo comprender su funcionamiento y conectividad.

El módulo CCP en modo PWM tuvo un buen rendimiento utilizando como salida el Led al cual se le gradúa su intensidad. Sin embargo, es importante destacar que debido a la alta frecuencia de trabajo que se requiere para la implementación de la comunicación USB® el rango de frecuencias que puede usar es de 3KHz a 769KHz, por lo tanto, no es muy funcional en motores ya que normalmente estos usan una frecuencia de 1KHz. Por otro lado, el modo comparación permite mayor flexibilidad frente a la frecuencia de trabajo ya que este utiliza el TIMER1 que tiene mayor capacidad de conteo. La mayor desventaja encontrada en este módulo al combinarlo con la comunicación USB® es la precisión frente al posicionamiento del servomotor, pero aun con dicha desventaja su posicionamiento en bastante bueno.

Con el módulo EUSART se obtuvo una interacción entre el computador y el celular donde se evidencia el uso del protocolo RS232 y la comunicación USB®, véase Figura 96.

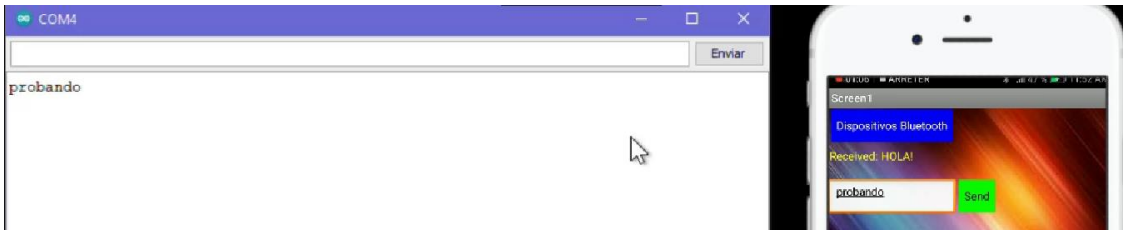


Figura 96. Resultado de programa con módulo EUSART. Fuente propia

Para el módulo MSSP haciendo uso de la comunicación I2C se obtiene un programa que permite la interacción con un RTC DS1307 por medio de diferentes etapas que permiten configurar la fecha y hora actuales además de leer los datos anteriormente guardados, véase Figura 97.

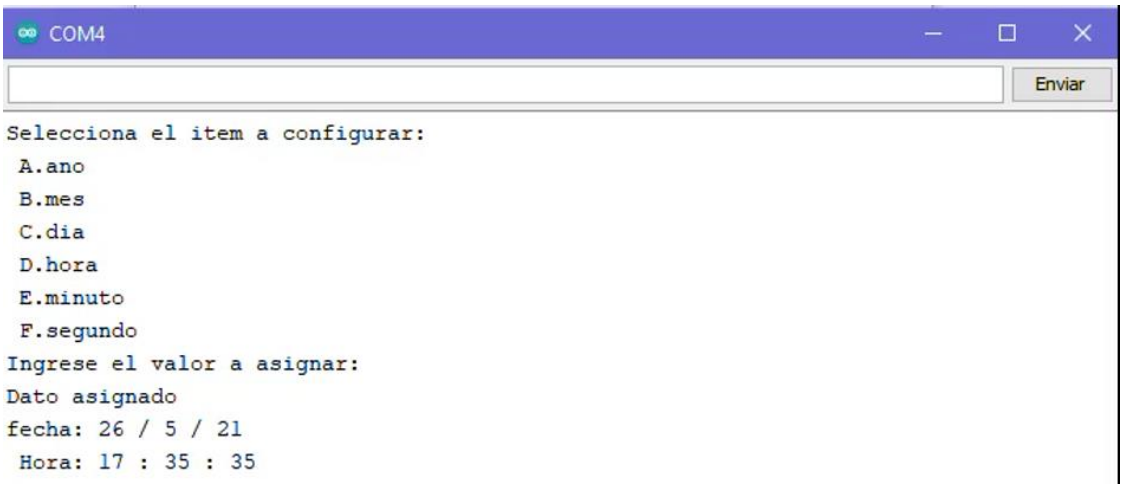


Figura 97. Resultado de programa con módulo MSSP. Fuente propia

En el programa del módulo EEPROM se obtiene una muy buena interacción entre la memoria EEPROM del microcontrolador y el computador, permitiendo guardar hasta 256 caracteres sin tener perdidas de información además de permitir la lectura de cualquier registro manteniendo esa información guardada, véase Figura 98. Aun así, es importante tener en cuenta que a la hora de enviar los caracteres desde el computador el carácter final corresponde a un "salto de línea".

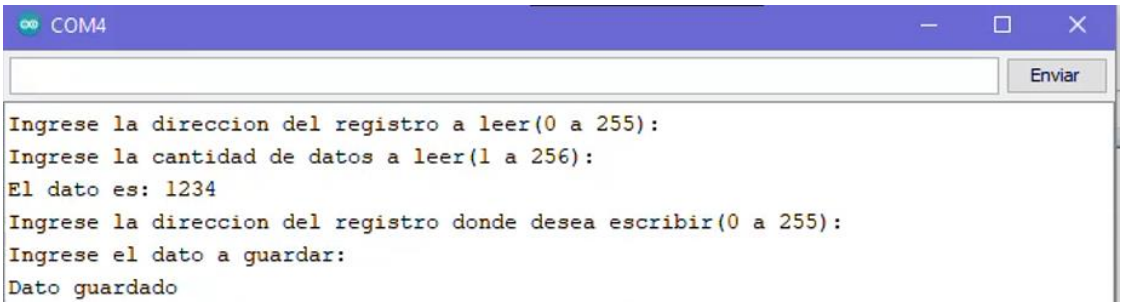


Figura 98. Resultado programa con módulo EEPROM. Fuente propia

Finalmente, para el programa base por interrupciones se tiene un programa más estable que el original, pero con el mismo funcionamiento el cual sirve como ejemplo para aplicaciones más complejas. (Para ver los resultados de los programas mencionados ingrese a: https://youtube.com/playlist?list=PLg3WXOmGn8LXhbtcIpuzT1xJ0KGVc8RFV)

Por otro lado, se diseñó una tarjeta PCB que permite la interacción con los diferentes programas mencionados anteriormente utilizando el software Proteus, véase Figura 99. Este tiene unas dimensiones de 9.8 cm x 7.3 cm, Además, incluye conectores para cada uno de los pines de entradas y salidas del microcontrolador, un conector para el uso de una LCD en el puerto D con un Trimmer que permite modular su brillo, un botón de RESET, un conector USB®

tipo B, un espacio para conectar el programador PICKIT3, entre otras características que la hacen muy funcional.

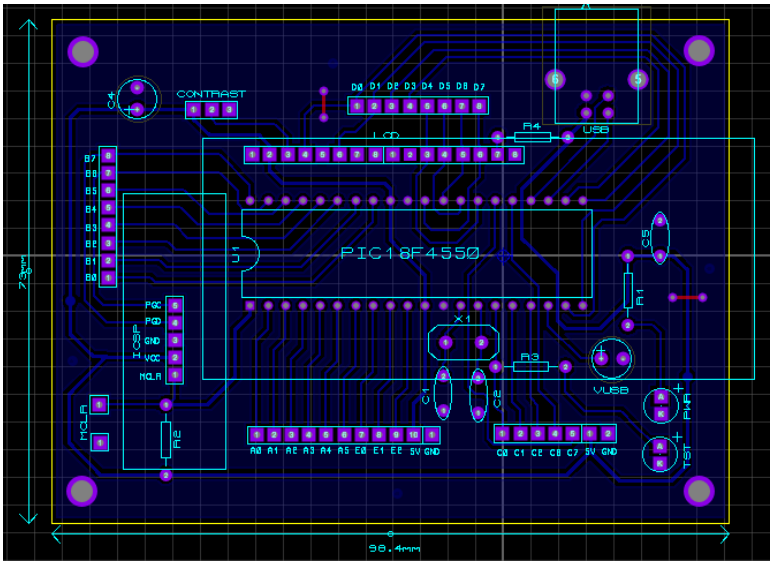


Figura 99. Diseño de PCB. Fuente propia

Para el ejemplo de Instrumentación se obtiene un sistema que permite visualizar la temperatura y humedad presentada en la caja en una gráfica por medio de MATLAB además de su visualización en la LCD. Inicialmente se presenta inestabilidad electromagnética en el sensor por lo cual se aplicaron filtros estadísticos que permitieran establecer con mayor exactitud las medidas tomadas. Por otro lado, se implementó una base de datos en Excel que permitiera tener un registro de los datos recolectados.

En el ejemplo de control ON OFF se obtiene un control bastante estable de la temperatura en el interior de la caja con un rango de referencia de ±3 del error. En los diferentes modos de interacción con la referencia se observa un buen tiempo de respuesta tanto del sistema como de la visualización de los datos en la interfaz de Python.

Finalmente, en el ejemplo de control difuso se observa un buen funcionamiento en cuanto a la variación de la intensidad del bombillo respecto a la temperatura presentada en la caja, pero al inicio con el uso de una sola entrada (error) se presenta una pequeña desestabilidad de la temperatura, por lo cual se opta por añadir una segunda entrada (derivada del error) que le da una mayor estabilidad al control del sistema. En cuanto a la interfaz, su tiempo de respuesta no es tan rápido, pero es apropiado para lo requerido frente a la visualización de la información. (Para ver el resultado de los ejercicios de aplicación ingrese a: https://youtube.com/playlist?list=PLg3WXOmGn8LUJKDT6TsLu0pxjNFJ1yyLA).

Cabe destacar que a partir del trabajo realizado en este proyecto se presentó una ponencia nacional mediante el articulo mostrado en el anexo 3.

7. Conclusiones

- La clase USB-CDC® es una buena opción para realizar un primer acercamiento a la programación de este protocolo de comunicación, ya que de manera general es la más sencilla de aplicar debido a la manera de uso.
- Se vuelve muy importante aprender cómo funciona la comunicación USB® para tener un mayor alcance en los diferentes proyectos a implementar, debido a su alta cantidad de aplicaciones además de su funcionalidad y velocidad.
- La comunicación USB® es más optima utilizándola por el método de interrupción (INTERRUPT) ya que hay más control sobre el tiempo de respuesta. Sin embargo, el modo sondeo (POLLING) permite otras aplicaciones más sencillas y requiere menos instrucciones de inicio.

8. Alcances

Partiendo de este trabajo, se proyecta la aplicación de este protocolo a otras áreas de la ingeniería como lo son robótica y automatización donde teniendo como base los programas presentados se puede realizar el manejo de diversos dispositivos utilizando la comunicación USB® con un computador teniendo como base un microcontrolador PIC18®. Además, del uso de otros dispositivos *Host* como un celular (haciendo uso del OTG) lo cual permitirá la interacción con un sistema operativo Android.

Por otro lado, se planea el uso de otras clases como la Clase HID que permite la interacción de dispositivos como lo son los ratones, teclados y apuntadores, la cual tolera de igual manera el alcance a mayores velocidades de transmisión. Siendo así, se podrían realizar proyectos que requieran de una velocidad de reacción mayor.

9. Referencias

[1] J. M. López, «USB: el cable universal que acabó con todos los demás,» Hipertextual, 29 Junio 2019. [En línea]. Available: https://hipertextual.com/2019/06/historia-usb-cable-universal. [Último acceso: 10 Febrero 2022].

[2] C. González, «¿Qué tipos de USB existen y en qué se diferencia cada cable?,» adsl zone, 14 Julio 2021. [En línea]. Available: https://www.adslzone.net/reportajes/tecnologia/tipos-clases-usb/. [Último acceso: 10 Febrero 2022].

[3] EcuRed, «USB,» EcuRed, 2019. [En línea]. Available: https://www.ecured.cu/USB. [Último acceso: 10 Febrero 2022].

[4] J. R. Hernández Martín, «USB1.0, USB1.1, USB2.0 y USB3.0,» EMEZETA, 2022. [En línea]. Available: https://www.emezeta.com/articulos/usb10-usb11-usb20-usb30. [Último acceso: 13 Febrero 2022].

[5] Tecnonautas, «USB 1.1: Todo lo que necesita saber,» 2020. [En línea]. Available: https://tecnonautas.net/usb-1-1-todo-lo-que-necesita-saber/. [Último acceso: 10 Febrero 2022].

[6] B. Gómez, «Descúbrelo todo sobre USB 4, el estándar basado en Thunderbolt,» Profesional Review, 25 Abril 2021. [En línea]. Available: https://www.profesionalreview.com/2021/04/25/usb-4/. [Último acceso: 10 Febrero 2022].

[7] Microcontroladoressesv, «Microcontroladores PIC y sus variedades,» 2020. [En línea]. Available: https://microcontroladoressesv.wordpress.com/microcontroladores-pic-y-sus-variedades/. [Último acceso: 10 Febrero 2022].

[8] H. U. Villamil Gonzalez y M. Pinto Aparicio, «MODULO DE MICROPROCESADORES & MICROCONTROLADORES,» *UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA,* pp. 130-131, 2009.

[9] Mecafenix, 06 Abril 2017. [En línea]. Available: https://www.ingmecafenix.com/electronica/microcontrolador-pic-partes-aplicaciones/.

[10] Z. Zamudio Beltrán, D. López Gutiérrez, R. Molina Pérez, D. E. Morales Palma y F. J. Osorio Oliveros, «Universidad La Salle Mexico,» 2017. [En línea]. Available: https://ingenieria.lasalle.mx/la-importancia-y-aplicaciones-de-los-microcontroladores-en-la-electronica/.

[11] Microcontroladores, «Empresas Fabricantes de microcontroladores,» 2015. [En línea]. Available: https://microcontroladoressesv.wordpress.com/empresas-fabricantes-de-microcontroladores/. [Último acceso: 26 Febrero 2022].

[12] O. Palermo, «microcontroladores pic,» 4 Abril 2013. [En línea]. Available: http://omarpalermopic.blogspot.com/2013/04/v-behaviorurldefaultvmlo_4.html. [Último acceso: 26 Febrero 2022].

[13] MICRO CONTROLADORES, «Microchip PIC18F4550,» 2022. [En línea]. Available: https://microcontroladoress.com/pic18f4550/. [Último acceso: 16 Febrero 2022].

[14] Microchip Technology Inc., «PIC18F2455/2550/4455/4550 Datasheet,» 2006.

[15] Ictea, «Base de Conocimientos,» 2021. [En línea]. Available: https://www.ictea.com/cs/index.php?rp=/knowledgebase/8817/iQue-es-un-programa-compilador.html.

[16] Microchip Technology Inc., «MPLAB® XC8 C Compiler Legacy User's Guide,» 2012.

[17] O. Weis, «What is called "USB"?,» Flexihub, 27 Agosto 2021. [En línea]. Available: https://www.flexihub.com/all-about-usb.html#usb. [Último acceso: 16 Febrero 2022].

[18] J. Axelson, USB Complete: Everything You Need to Develop Custom USB Peripherals. Third Edition, Madison: Lakeview Research, U.S., 2005.

[19] crisross, «PROTOCOLO USB,» ARQUITECTURA DE COMPUTADORAS, 21 Mayo 2013. [En línea]. Available: https://arquitecturadecomputadora.wordpress.com/2013/05/21/protocolo-usb/. [Último acceso: 16 Febrero 2022].

[20] Electronics notes, «USB Operation: Protocol, Data Transfer & Packets,» Radio-Electronics, 2022. [En línea]. Available: https://www.electronics-notes.com/articles/connectivity/usb-universal-serial-bus/protocol-data-transfer.php. [Último acceso: 16 Febrero 2022].

[21] V. Akpan, «¿Qué es un USB OTG y qué cosas increíbles puedes hacer con él?,» suntrustblog, Enero 2022. [En línea]. Available: https://suntrustblog.com/es/what-is-a-usb-otg/. [Último acceso: 17 Febrero 2022].

[22] B. Murphy, «What is Enumeration and Why are USB Descriptors Important?,» totalphase, 11 Agosto 2020. [En línea]. Available: https://www.totalphase.com/blog/2020/08/what-is-enumeration-why-usb-descriptors-important/. [Último acceso: 17 Febrero 2022].

[23] Microchip Inc., «How USB Communicates,» 2021. [En línea]. Available: https://microchipdeveloper.com/usb:how-it-works. [Último acceso: 17 Febrero 2022].

[24] G. Marker, «Historia y evolución del USB,» Tecnologia+informatica, 9 Marzo 2019. [En línea]. Available: https://www.tecnologia-informatica.com/historia-evolucion-usb/#Un_poco_de_historia. [Último acceso: 16 Febrero 2022].

[25] Y. Fernández, «Tipos de USB: estándares, conectores y características de cada uno,» Xataka, 21 Abril 2021. [En línea]. Available: https://www.xataka.com/basics/tipos-usb-estandares-conectores-caracteristicas-cada-uno. [Último acceso: 16 Febrero 2022].

[26] Y. Fernández, «USB4 o USB 4.0: qué es y cuáles son sus diferencias respecto a USB 3.0,» Xataka, 21 Abril 2021. [En línea]. Available: https://www.xataka.com/basics/usb4-usb-4-0-que-cuales-sus-diferencias-respecto-a-usb-3-0. [Último acceso: 16 Febrero 2022].

[27] CMD ltd, «USB DATA TRANSFER GUIDE,» 2022. [En línea]. Available: https://www.cmd-ltd.com/advice-centre/usb-chargers-and-power-modules/usb-and-power-module-product-help/usb-data-transfer-guide/. [Último acceso: 16 Febrero 2022].

[28] J. Roca, «¿Qué tipos de conectores USB existen y en qué se diferencian?,» hardzone, 19 Noviembre 2021. [En línea]. Available: https://hardzone.es/tutoriales/rendimiento/conector-usb-tipos/. [Último acceso: 16 Febrero 2022].

[29] Profesional Review, «USB: Qué es, tipos, formatos y velocidades,» 2019. [En línea]. Available: https://www.profesionalreview.com/conectores-pc/usb/#Como_funciona_un_USB. [Último acceso: 16 Febrero 2022].

[30] B. Noguera, «Cuál es la diferencia entre un micro USB y un mini USB,» Culturación, 2021. [En línea]. Available: https://culturacion.com/cual-es-la-diferencia-entre-un-micro-usb-y-un-mini-usb/. [Último acceso: 16 Febrero 2022].

[31] P. López, «¿Qué es Micro USB y para qué sirve?,» Geeknetic, 14 Julio 2020. [En línea]. Available: https://www.geeknetic.es/Micro-USB/que-es-y-para-que-sirve. [Último acceso: 16 Febrero 2022].

[32] WhistleOut MX, «¿Qué es USB Tipo C y qué beneficios tiene?,» 8 DIciembre 2021. [En línea]. Available: https://www.whistleout.com.mx/CellPhones/Guides/usb-tipo-c. [Último acceso: 16 Febrero 2022].

[33] USB-IF, «Defined Class Codes,» 7 Enero 2022. [En línea]. Available: https://www.usb.org/defined-class-codes. [Último acceso: 16 Febrero 2022].

[34] USBlyzer, «Communication Device Class (CDC) Decoder,» 2006. [En línea]. Available: http://www.usblyzer.com/usb-communication-device-class-cdc-decoder.htm. [Último acceso: 16 Febreo 2022].

[35] armKeil, «CDC: Communication Device Class,» 25 Mayo 2021. [En línea]. Available: https://www.keil.com/pack/doc/mw/USB/html/_c_d_c.html. [Último acceso: 16 Febrero 2022].

[36] USBlyzer, «Human Interface Device (HID) Class Decoder,» 2006. [En línea]. Available: http://www.usblyzer.com/usb-human-interface-device-hid-class-decoder.htm. [Último acceso: 16 Febrero 2020].

[37] armKeil, «HID: Human Interface Device Class,» 25 Mayo 2021. [En línea]. Available: https://www.keil.com/pack/doc/mw/USB/html/_h_i_d.html. [Último acceso: 16 Febrero 2022].

[38] Technopedia, «What Does Mass Storage Device (MSD) Mean?,» 2022. [En línea]. Available: https://www.techopedia.com/definition/11901/mass-storage-device-msd. [Último acceso: 16 Febrero 2022].

[39] armKeil, «MSC: Mass Storage Class,» 25 Mayo 2021. [En línea]. Available: https://www.keil.com/pack/doc/mw/USB/html/_m_s_c.html. [Último acceso: 16 Febrero 2022].

[40] R. Adeva, «Qué es Windows e historia del famoso sistema operativo,» adslzone, 11 Junio 2021. [En línea]. Available: https://www.adslzone.net/reportajes/software/que-es-windows/. [Último acceso: 23 Febrero 2022].

[41] R. Velasco, «Exprime al máximo tu Raspberry Pi con estos sistemas operativos,» softzone, 02 Diciembre 2021. [En línea]. Available: https://www.softzone.es/programas/sistema/mejores-sistemas-operativo-raspberry-pi/. [Último acceso: 20 Febrero 2022].

[42] Microchip Inc., «MPLAB® X IDE,» 2022. [En línea]. Available: https://www.microchip.com/en-us/tools-resources/develop/mplab-x-ide#Downloads%20and%20Documentation. [Último acceso: 16 Febrero 2022].

[43] MouserElectronics, «Microchip Technology MPLAB® X Integrated Development Environment (IDE),» 2022. [En línea]. Available: https://co.mouser.com/new/microchip/microchip-mplab-x-ide/. [Último acceso: 16 Febrero 2022].

[44] Laboratorio de Electrónica, «Proteus,» 2022, [En línea]. Available: http://labelectronica.weebly.com/proteus.html. [Último acceso: 16 Febrero 2022].

[45] J. Carlos, «Que es Mat Lab?,» 2015. [En línea]. Available: https://juancarlosusomatlab2015.weebly.com/definicion-matlab.html. [Último acceso: 16 Febrero 2022].

[46] Linkfang, «MATLAB,» 2021. [En línea]. Available: https://es.linkfang.org/wiki/MATLAB. [Último acceso: 16 Febrero 2022].

[47] BeJob, «7 RAZONES PARA PROGRAMAR EN PYTHON,» 18 Septiembre 2016. [En línea]. Available: https://www.bejob.com/7-razones-para-programar-en-python/. [Último acceso: 16 Febrero 2022].

[48] Oscar, «Arduino teclado 4x4,» codigoelectronica, 14 Enero 2019. [En línea]. Available: http://codigoelectronica.com/blog/arduino-teclado-4x4#que-es-un-teclado4x4. [Último acceso: 16 Febrero 2022].

[49] areatecnologia, «SERVOMOTORES,» 2021. [En línea]. Available: https://www.areatecnologia.com/electricidad/servomotor.html#%C2%BFQu%C3%A9_es_un_Servomotor. [Último acceso: 16 Febrero 2022].

[50] naylampmechatronics, «MÓDULO BLUETOOTH HC05,» 2021. [En línea]. Available: https://naylampmechatronics.com/inalambrico/43-modulo-bluetooth-hc05.html. [Último acceso: 16 Febrero 2021].

[51] Technopedia, «Real-Time Clock (RTC),» 2022. [En línea]. Available: https://www.techopedia.com/definition/2273/real-time-clock-rtc. [Último acceso: 16 Febrero 2022].

[52] 5Hertz Electronica, «INTRODUCCIÓN A LAS LCD,» 2022. [En línea]. Available: https://www.5hertz.com/index.php?route=tutoriales/tutorial&category_id=1&tutorial_id=9#11. [Último acceso: 16 Febrero 2022].

[53] Components101, «DHT11–Temperature and Humidity Sensor,» 16 Julio 2021. [En línea]. Available: https://components101.com/sensors/dht11-temperature-sensor. [Último acceso: 16 Febrero 2022].

[54] Electroindustria, «¿Qué es un relé de estado sólido?,» Septiembre 2020. [En línea]. Available: http://www.emb.cl/electroindustria/articulo.mvc?xid=3691&srch=%BFQu%E9%20es%20un%20rel%E9%20de%20estado%20s%F3lido?&act=4&tip=7&xit=. [Último acceso: 16 Febrero 2022].

[55] Aitor, «DIMMER- QUE ES, TIPOS Y COMO FUNCIONA UN ATENUADOR DE LUZ,» ERENOVABLE, 19 Octubre 2021. [En línea]. Available: https://erenovable.com/que-es-un-dimmer/. [Último acceso: 16 Febrero 2022].

[56] E. Rodríguez de Luis, «De cero a maker: todo lo necesario para empezar con Raspberry Pi,» Xakata, 15 Julio 2018. [En línea]. Available: https://www.xataka.com/makers/cero-maker-todo-necesario-para-empezar-raspberry-pi. [Último acceso: 16 Febrero 2022].

[57] programoergosum, «Introducción a Raspberry Pi,» 2021. [En línea]. Available: https://www.programoergosum.es/tutoriales/introduccion-a-raspberry-pi/. [Último acceso: 16 Febrero 2022].

[58] S. De Luz, «Raspberry Pi 4: Análisis de este mini ordenador (versión 4GB de RAM),» redes zone, 30 Julio 2019. [En línea]. Available: https://www.redeszone.net/analisis/otros-dispositivos/raspberry-pi-4-analisis/. [Último acceso: 22 Febrero 2022].

[59] Electric Bricks, «PWM: Control de Velocidad en motores eléctricos,» 26 Marzo 2010. [En línea]. Available: http://blog.electricbricks.com/2010/03/pwm-coltrol-de-velocidad-en-motores-electricos/. [Último acceso: 26 Febrero 2022].

[60] Vizcainostore, «¿CÓMO FUNCIONA UN RELÉ DE ESTADO SÓLIDO?,» 2021. [En línea]. Available: https://vizcainostore.com/como-funciona-un-rele-de-estado-solido/. [Último acceso: 26 Febrero 2022].

[61] F. J. Ruvalcaba Coyaso y A. Vermonden, «Lógica difusa para la toma de decisiones y la selección de personal,» *Universidad del Rosario. Universidad & Empresa,* vol. 17, nº 29, pp. 239-256, 2015.

[62] jfmateos2, «TODOPIC,» 05 Febrero 2011. [En línea]. Available: http://www.todopic.com.ar/foros/index.php?PHPSESSID=qcrpvrhug17lm510penrim03m3&topic=33699.0 .

[63] pablinza, «ENTRENAMIENTO PIC CON XC8,» 26 Julio 2019. [En línea]. Available: https://pablinza.blogspot.com/2019/07/usb-serial-cdc-con-pic16f.html. [Último acceso: 20 Febrero 2021].

[64] Mouser Electronics, Inc., *DHT11 Humidity & Temperature Sensor,* Mansfield, Texas USA, 2022.

[65] R. Lozano, «Diferencia entre rele mecánico y rele estado solido (ssr),» taloselectronics, 18 Abril 2020. [En línea]. Available: https://www.taloselectronics.com/blogs/tutoriales/diferencia-entre-rele-mecanico-y-rele-estado-solido-ssr. [Último acceso: 20 Febrero 2022].

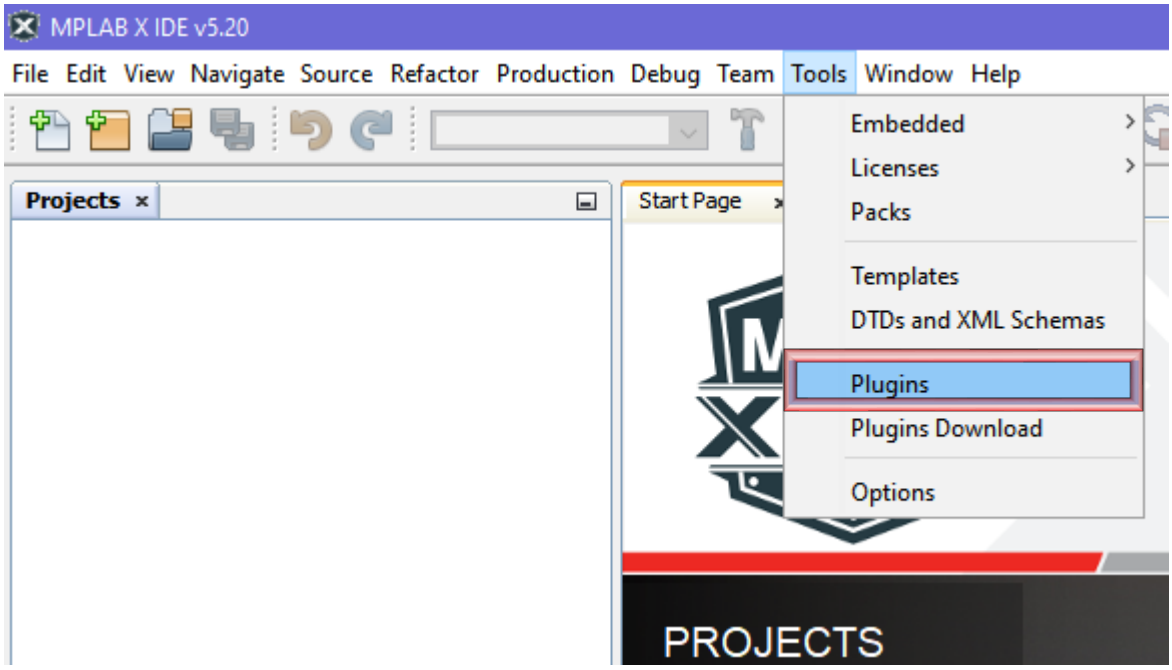
[66] SUTAGO, *Dimmer Digital TRIAC BTA312,* 2022.

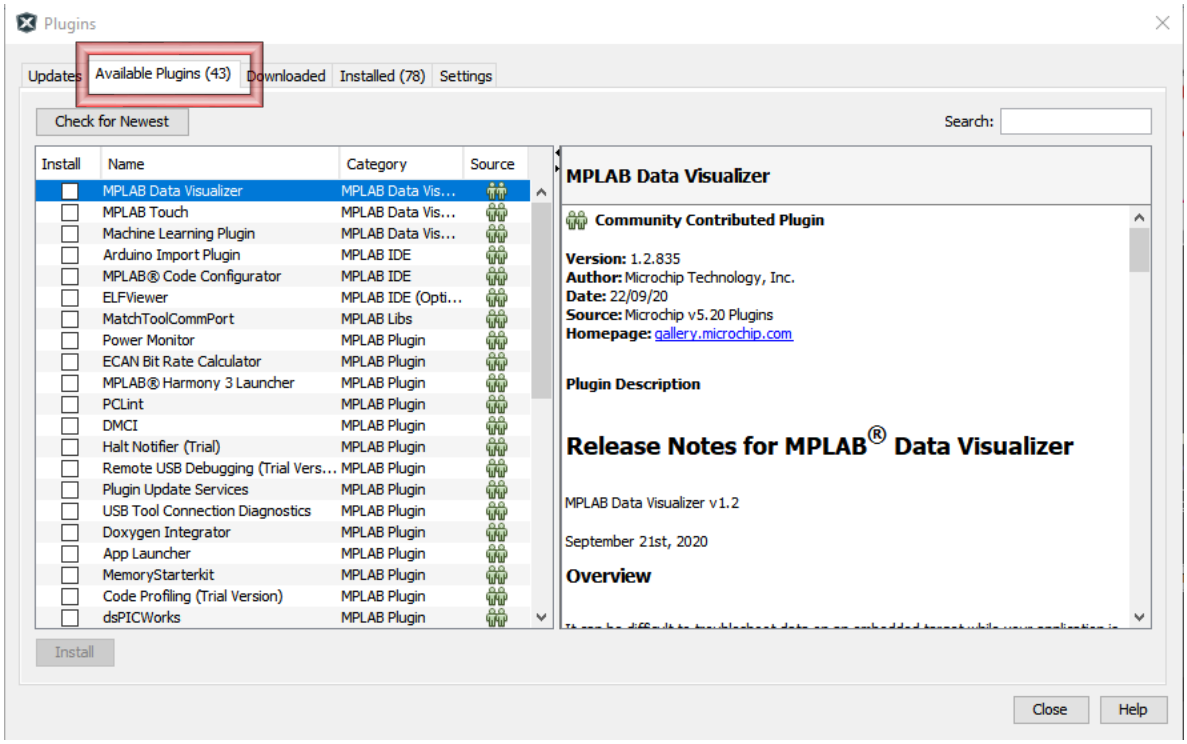
10. Anexos

10.1 Manual para generar las librerías USB-CDC®

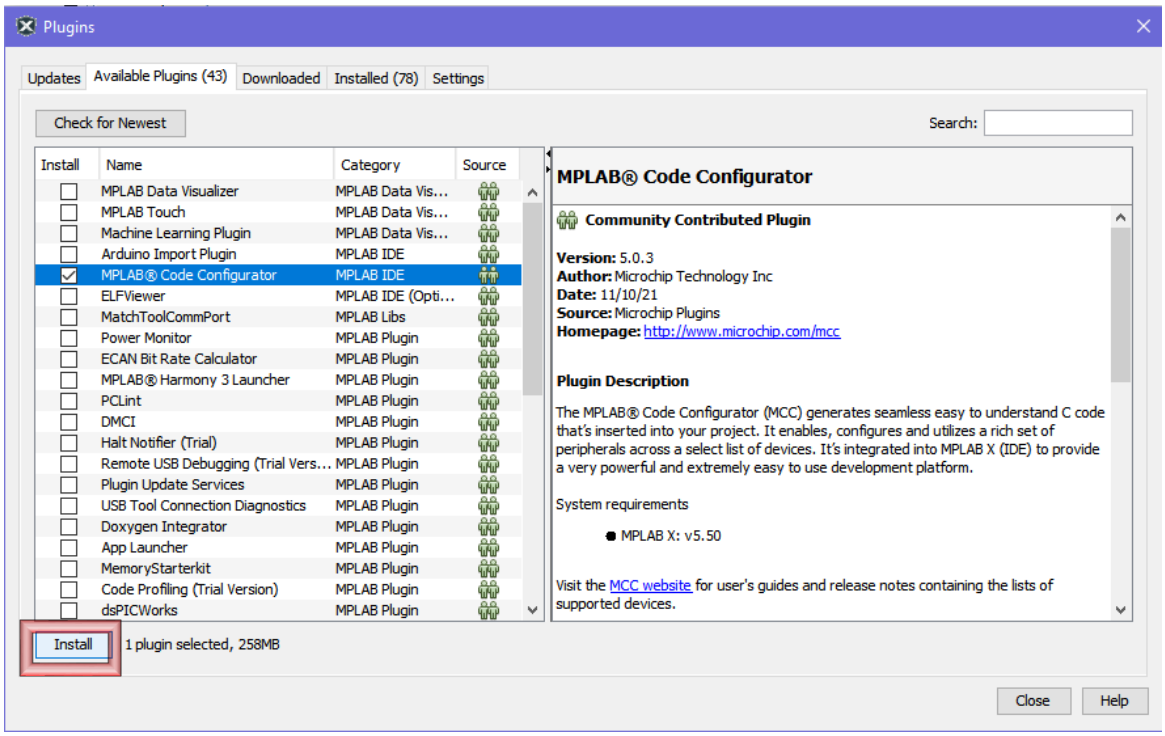
INSTALACIÓN DE MPLAB® CODE CONFIGURATION (MCC)

1. Para empezar, se abre MPLAB®X y se debe instalar el complemento de MPLABX llamado MPLAB code configuration para ello se va a >Tools, >Plugins, >Available plugins



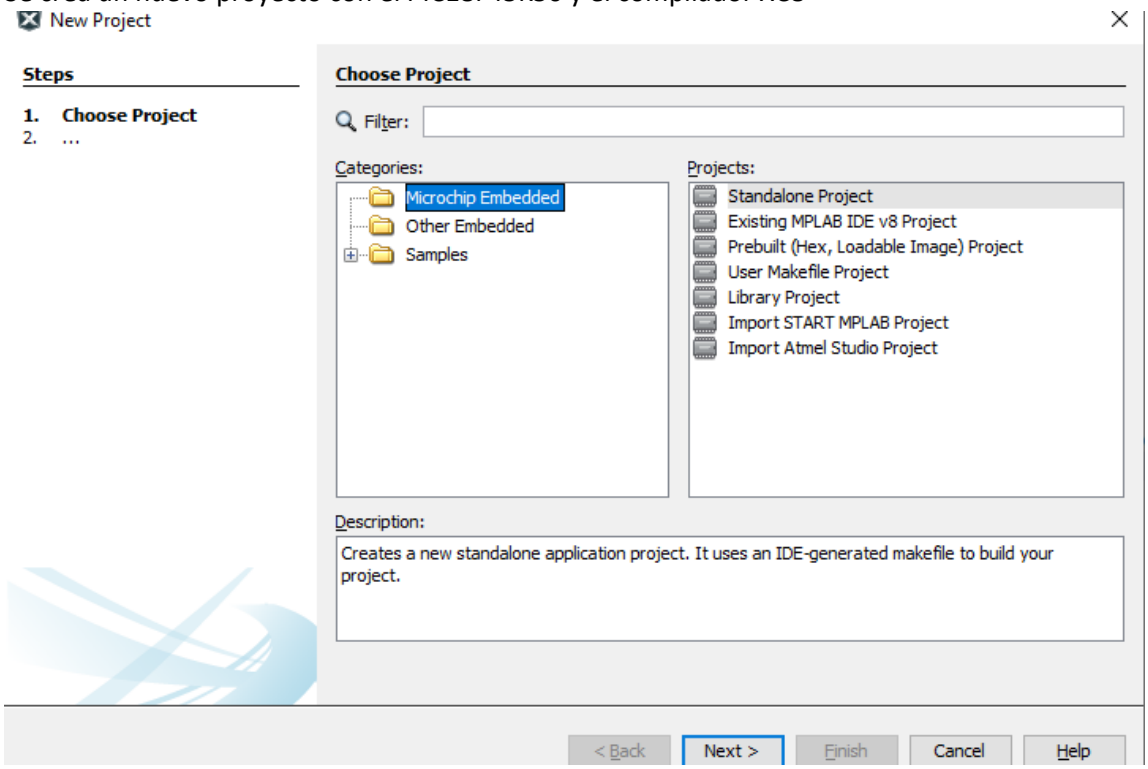


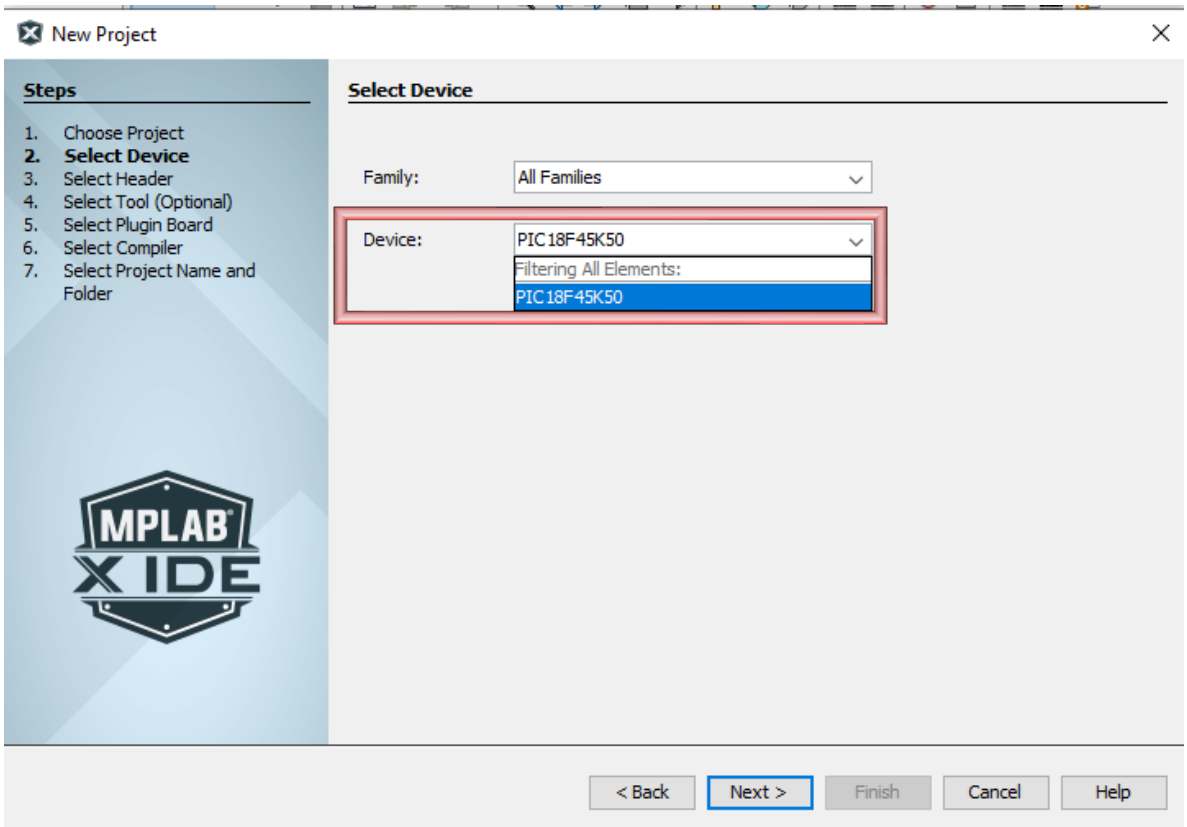
2. Se selecciona MPLAB® code configurator y se da clic en >Install (es posible que una vez instalado toque reiniciar el programa)

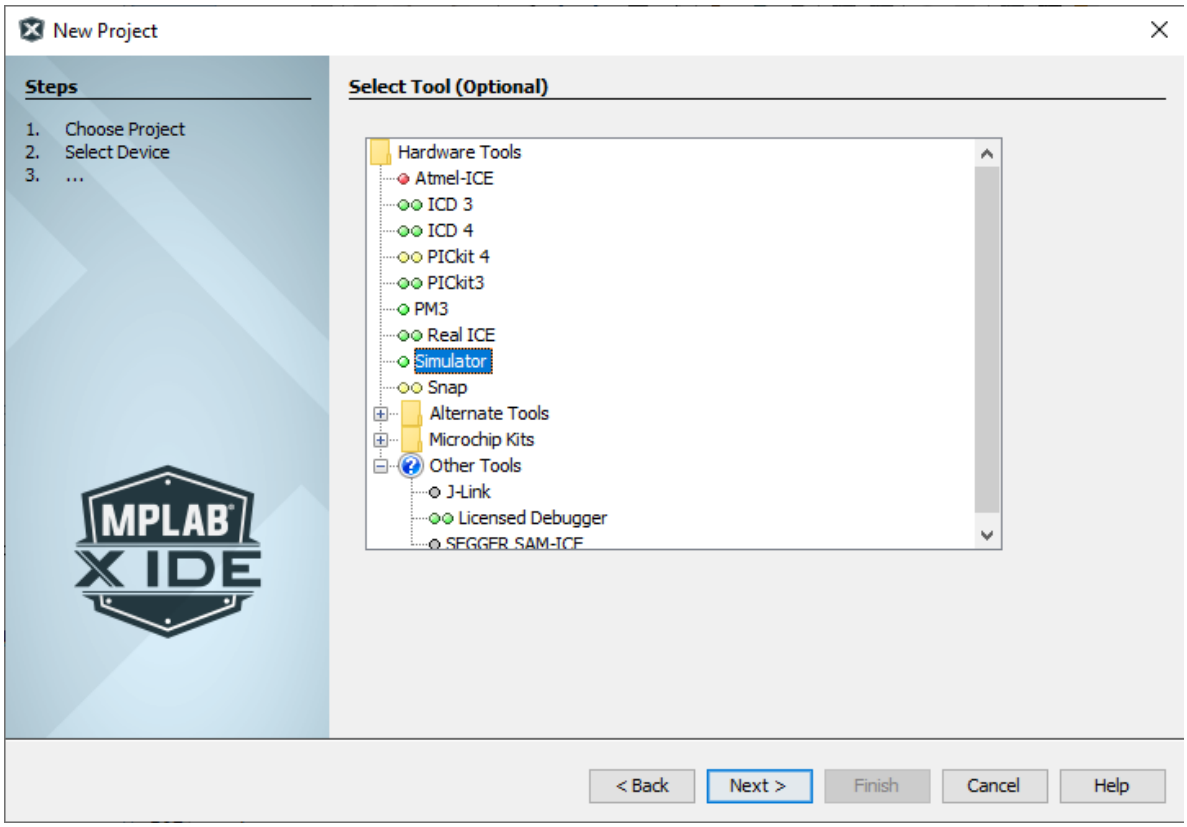


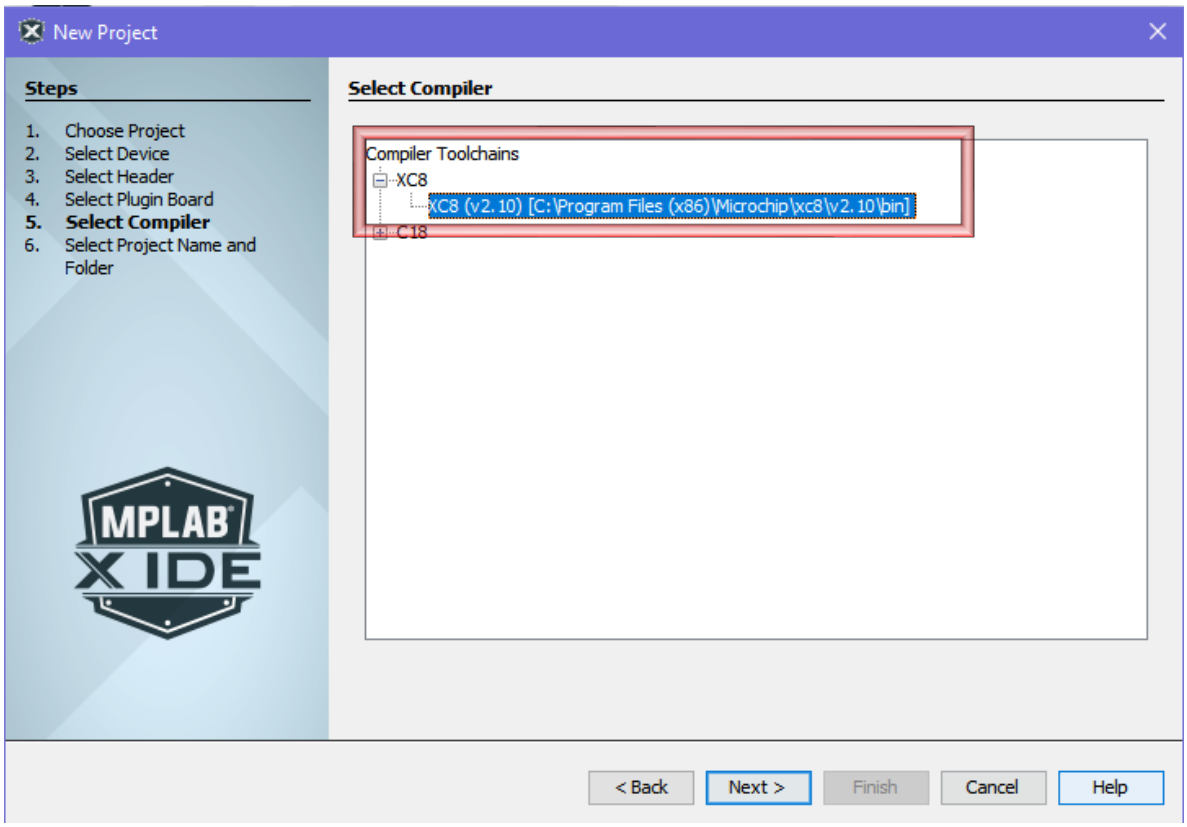
PASO A PASO PROGRAMA USB-CDC PARA PIC18F45K50

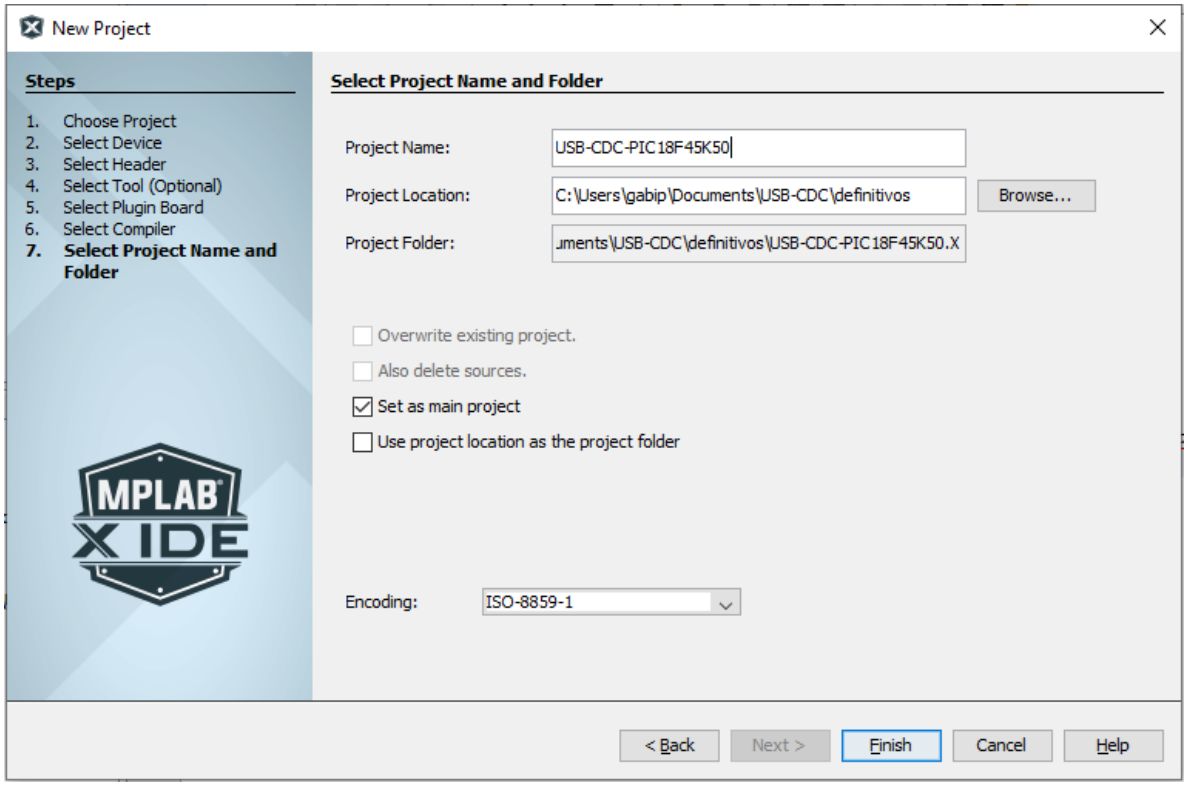
1. Se crea un nuevo proyecto con el PIC18F45K50 y el compilador XC8




New Project
Steps
1. Choose Project
2. Select Device
3. Select Header
4. Select Tool (Optional)
5. Select Plugin Board
6. Select Compiler
7. Select Project Name and Folder
Select Device
Family:
All Families
Device:
PIC18F45K50
Filtering All Elements:
PIC18F45K50
MPLAB X IDE
< Back
Next >
Finish
Cancel
Help

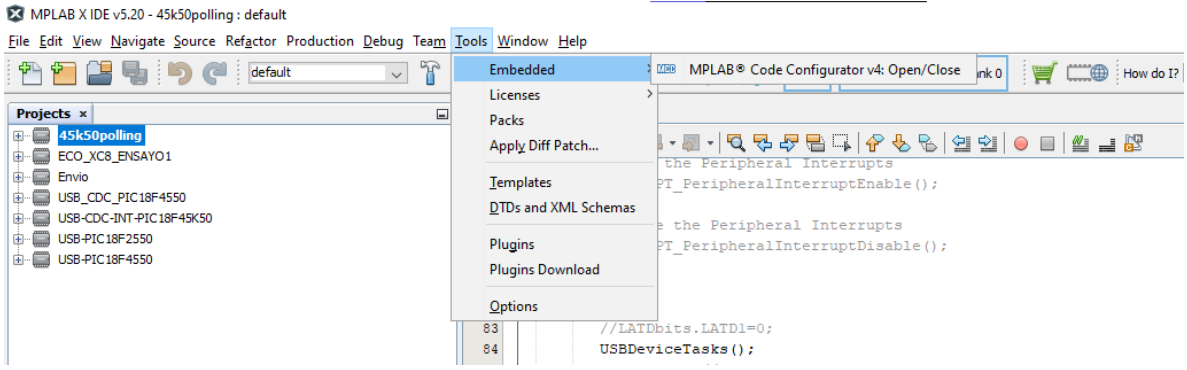

New Project
Steps
1. Choose Project
2. Select Device
3. ...
Select Tool (Optional)
Hardware Tools
Atmel-ICE
ICD 3
ICD 4
PICkit 4
PICkit3
PM3
Real ICE
Simulator
Snap
Alternate Tools
Microchip Kits
Other Tools
J-Link
Licensed Debugger
MPLAB X IDE
< Back
Next >
Finish
Cancel
Help


New Project
Steps
1. Choose Project
2. Select Device
3. Select Header
4. Select Plugin Board
5. Select Compiler
6. Select Project Name and Folder
Select Compiler
Compiler Toolchains
XC8
XC8 (v2.10) [C:\Program Files (x86)\Microchip\xc8\v2.10\bin]
C18
MPLAB X IDE
< Back
Next >
Finish
Cancel
Help


New Project
Steps
1. Choose Project
2. Select Device
3. Select Header
4. Select Tool (Optional)
5. Select Plugin Board
6. Select Compiler
7. Select Project Name and Folder
Select Project Name and Folder
Project Name:
USB-CDC-PIC18F45K50
Project Location:
C:\Users\gabip\Documents\USB-CDC\definitivos
Browse...
Project Folder:
uments\USB-CDC\definitivos\USB-CDC-PIC18F45K50.X
Overwrite existing project.
Also delete sources.
Set as main project
Use project location as the project folder
Encoding:
ISO-8859-1
MPLAB X IDE
< Back
Next >
Finish
Cancel
Help

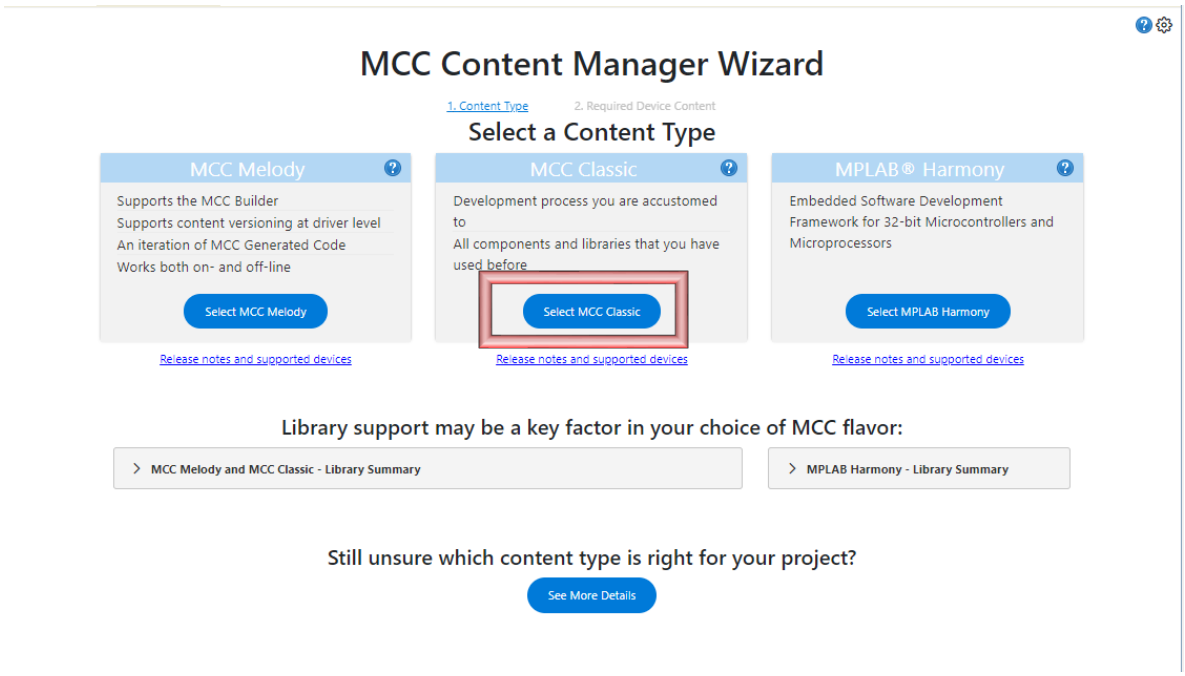
NOTA: El proyecto debe ser reconocido como el proyecto principal (main project)

2. Se abre MCC así: >Tools, >Embedded y >MPLAB® Code Configurator v5 Open/Close

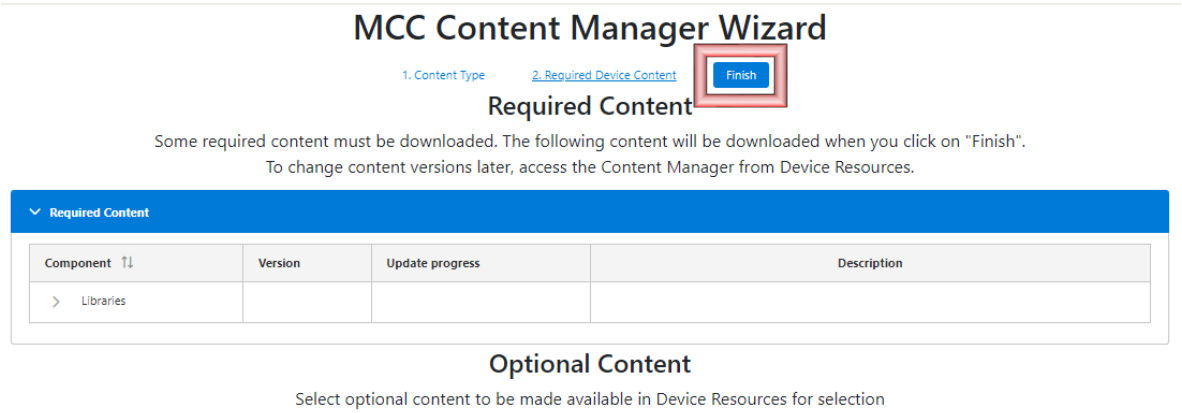


También se puede abrir por medio del icono mostrado

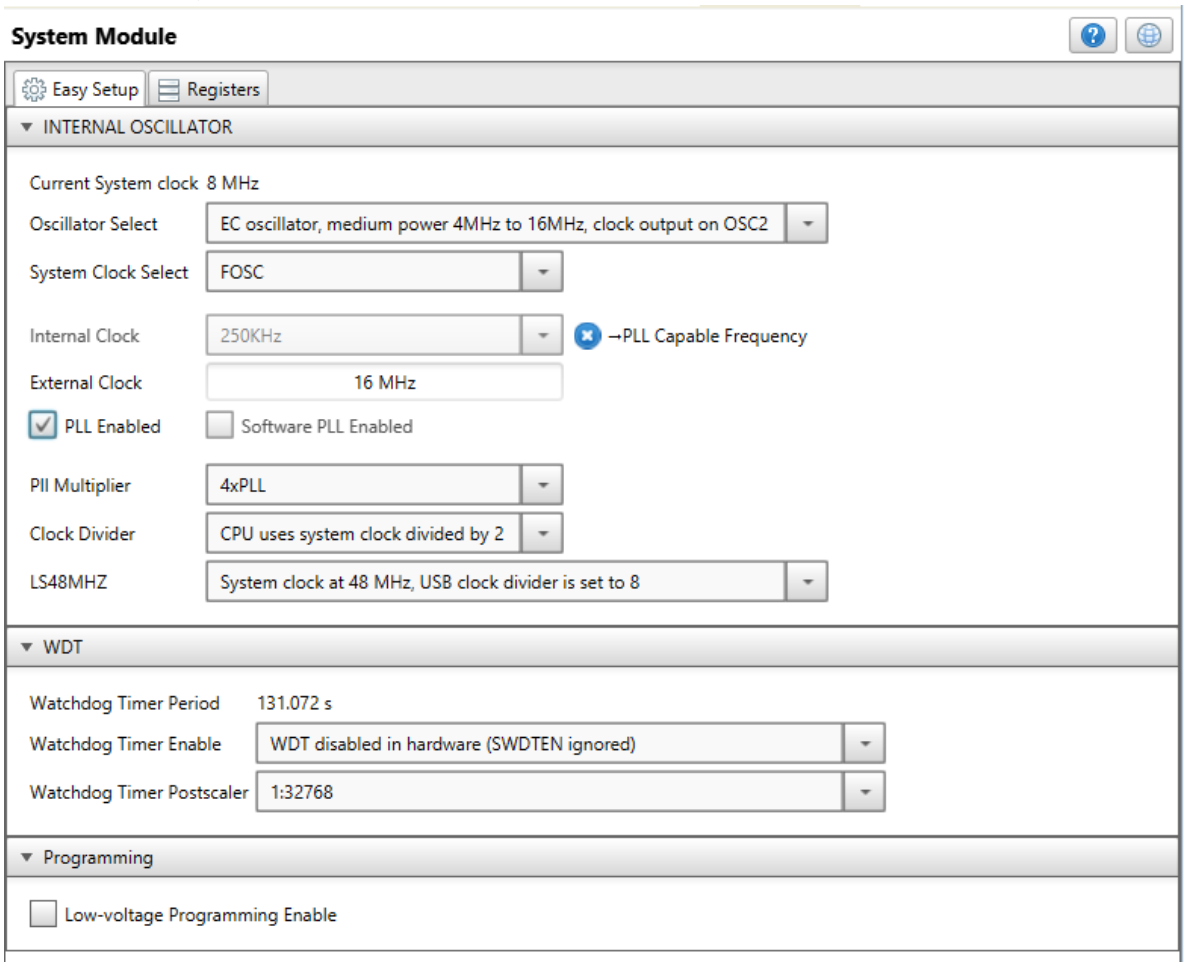
3. Aparece esta ventana; allí se debe seleccionar >Select MCC Classic



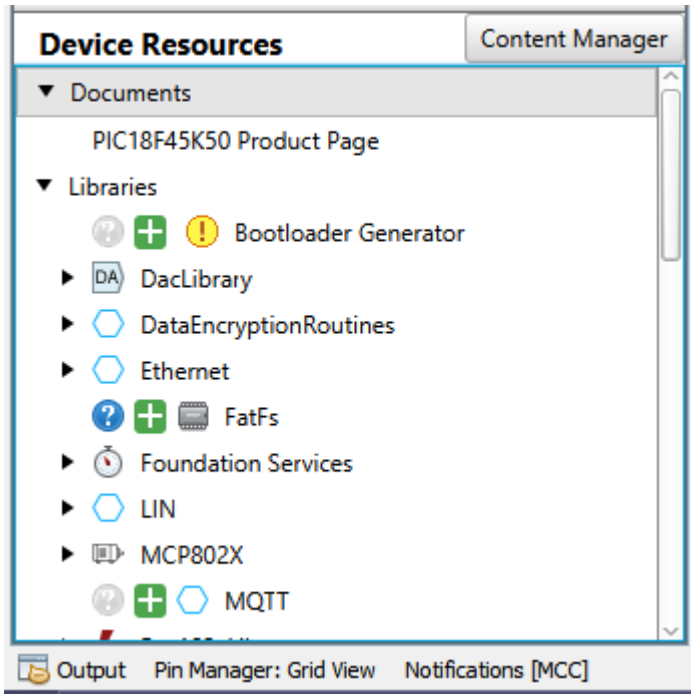
4. Aparece esta ventana; allí se debe seleccionar >Finish



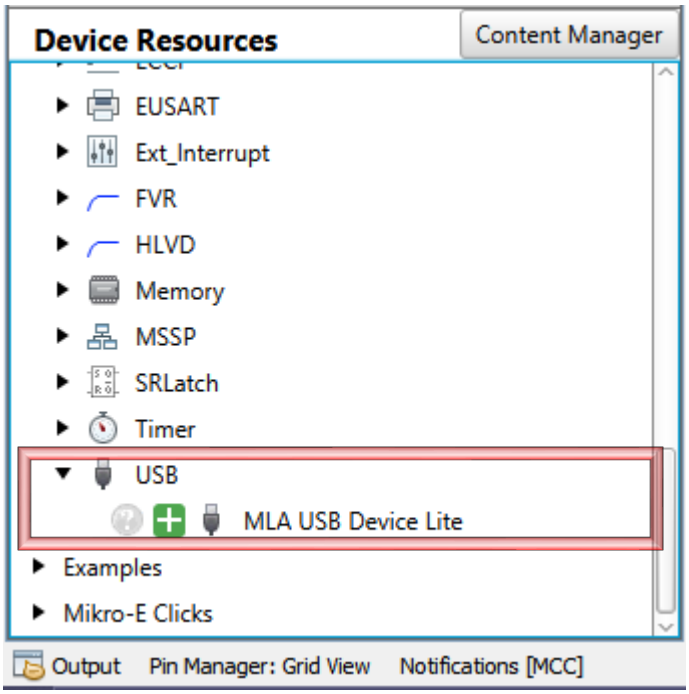
5. Se despliegan unas ventanas.
   En la pestaña >System Module, en >Easy Setup poner la siguiente configuración (cristal externo 16 MHz)



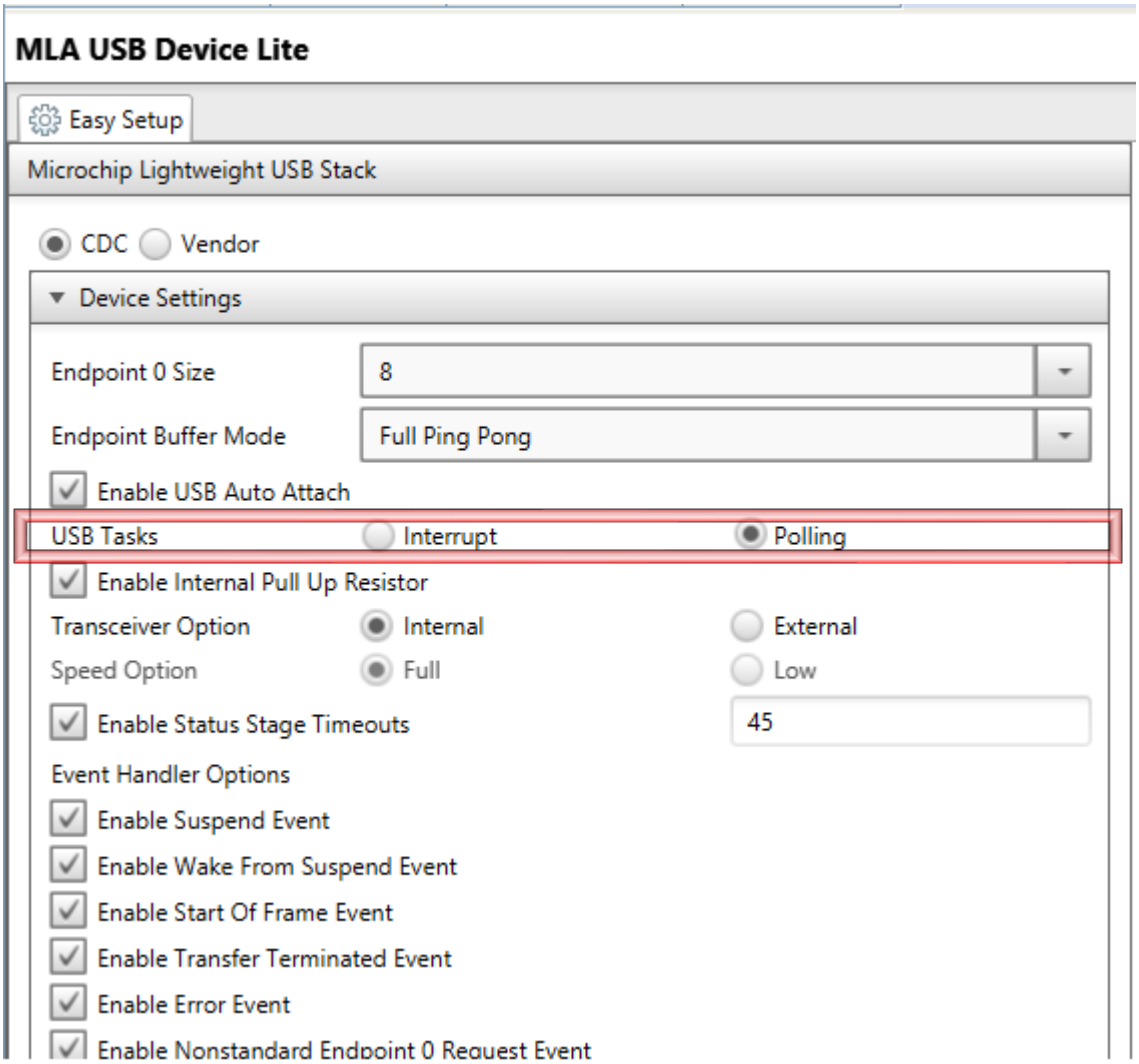
6. En la esquina izquierda- abajo se encuentra la siguiente ventana

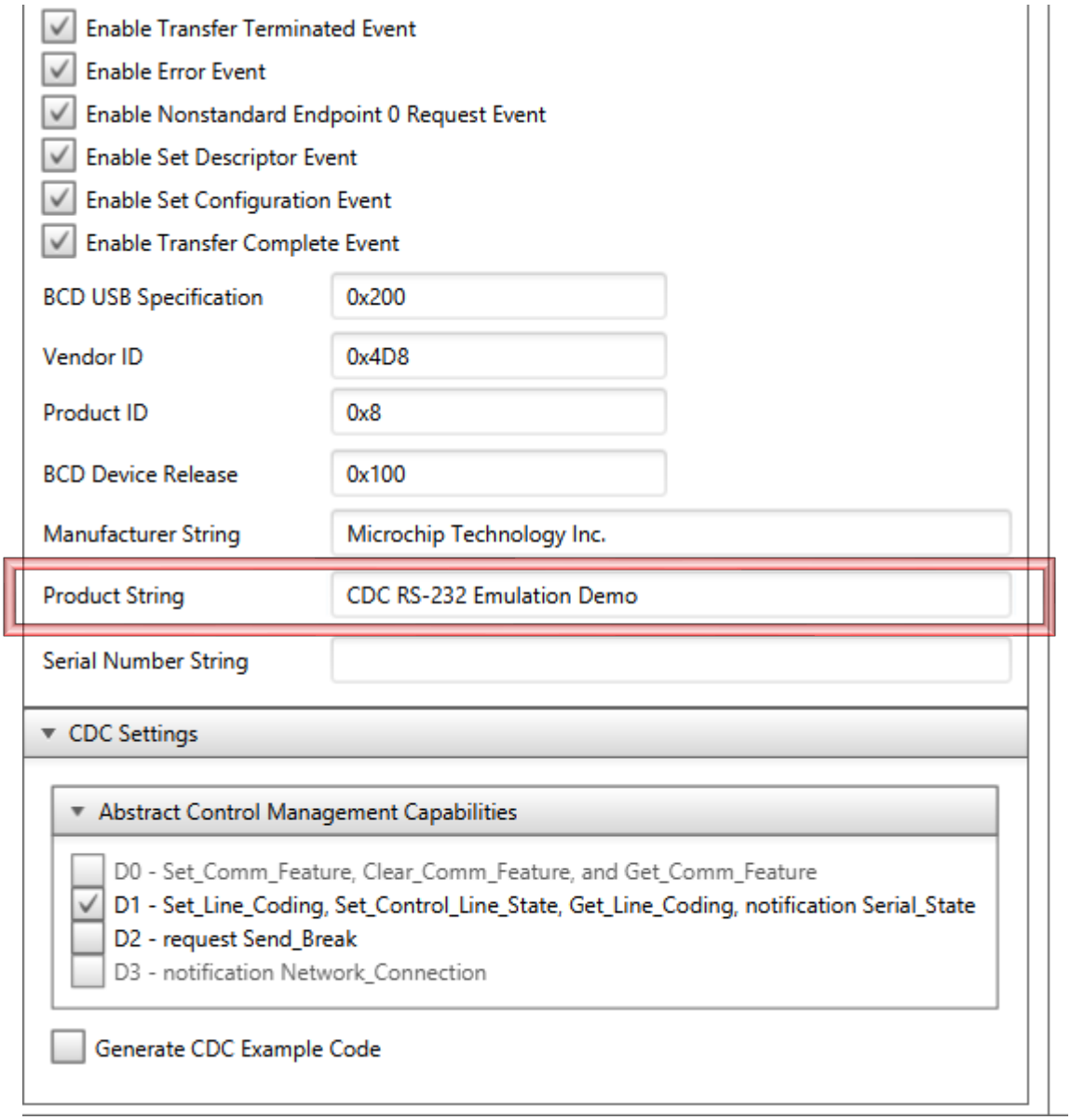


Allí se debe buscar USB y seleccionar >MLA USB Device Lite

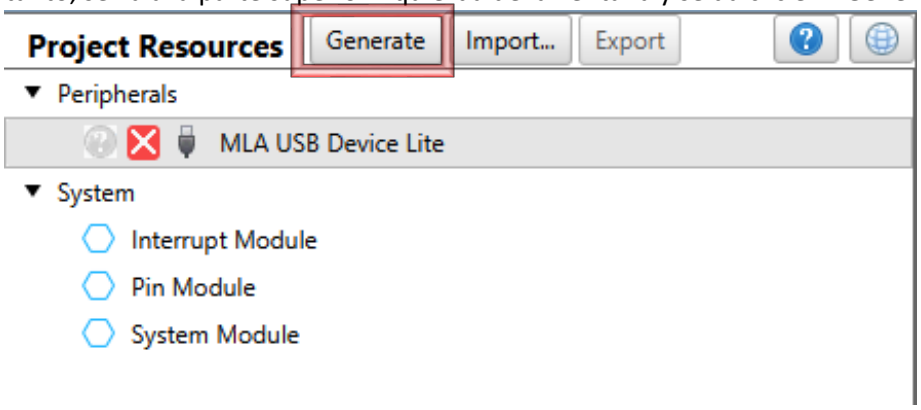


7. Se despliega una ventana, allí se debe colocar la configuración mostrada y en USB Tasks definir si la comunicación se hará por Polling o Interrupt (para este caso se utiliza Polling). Además, en Product String colocar el nombre deseado para el dispositivo USB.

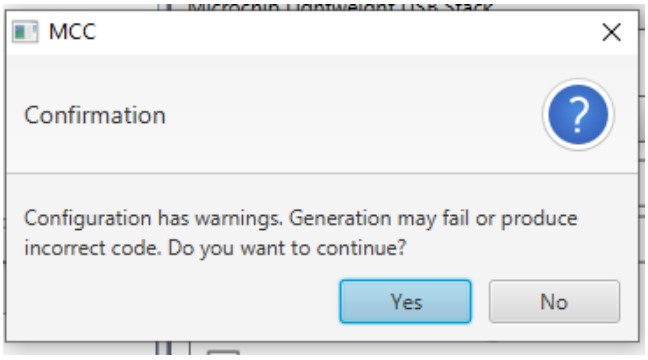

MLA USB Device Lite
Easy Setup
Microchip Lightweight USB Stack
CDC
Vendor
Device Settings
Endpoint 0 Size
8
Endpoint Buffer Mode
Full Ping Pong
Enable USB Auto Attach
USB Tasks
Interrupt
Polling
Enable Internal Pull Up Resistor
Transceiver Option
Internal
External
Speed Option
Full
Low
Enable Status Stage Timeouts
45
Event Handler Options
Enable Suspend Event
Enable Wake From Suspend Event
Enable Start Of Frame Event
Enable Transfer Terminated Event
Enable Error Event
Enable Nonstandard Endpoint 0 Request Event



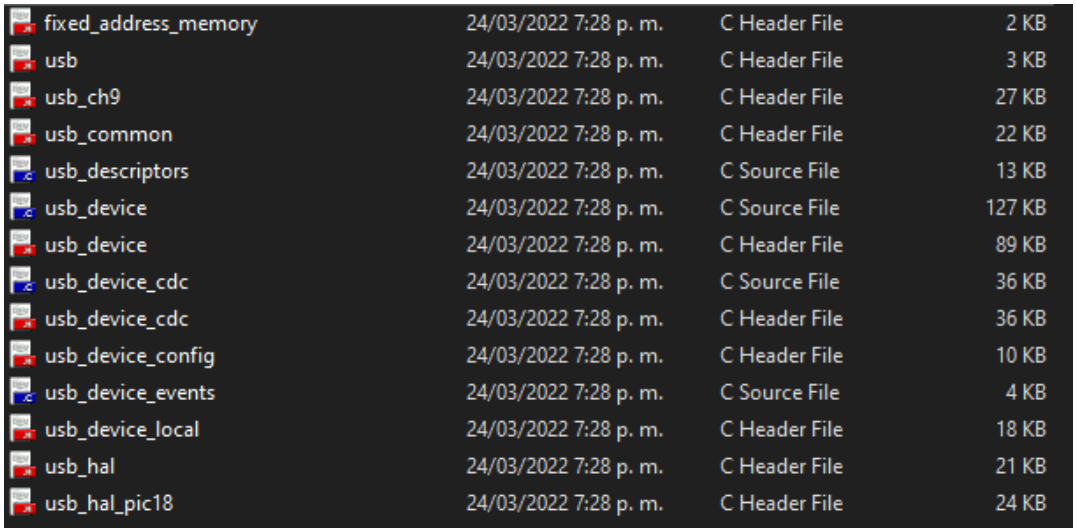
8. Como solo se necesita generar las librerías no se realizarán más configuraciones; por lo tanto, se va a la parte superior izquierda de la ventana y se da clic en >Generate



9. Si muestra un mensaje de confirmación, se da clic en SI <Yes>



10. Ahora se abre la carpeta del programa, allí se abre la carpeta <mcc_generated_files> y finalmente a la carpeta <usb>. Ahí estarán las librerías que se utilizan para los programas realizados.



fixed_address_memory	24/03/2022 7:28 p. m.	C Header File	2 KB
usb	24/03/2022 7:28 p. m.	C Header File	3 KB
usb_ch9	24/03/2022 7:28 p. m.	C Header File	27 KB
usb_common	24/03/2022 7:28 p. m.	C Header File	22 KB
usb_descriptors	24/03/2022 7:28 p. m.	C Source File	13 KB
usb_device	24/03/2022 7:28 p. m.	C Source File	127 KB
usb_device	24/03/2022 7:28 p. m.	C Header File	89 KB
usb_device_cdc	24/03/2022 7:28 p. m.	C Source File	36 KB
usb_device_cdc	24/03/2022 7:28 p. m.	C Header File	36 KB
usb_device_config	24/03/2022 7:28 p. m.	C Header File	10 KB
usb_device_events	24/03/2022 7:28 p. m.	C Source File	4 KB
usb_device_local	24/03/2022 7:28 p. m.	C Header File	18 KB
usb_hal	24/03/2022 7:28 p. m.	C Header File	21 KB
usb_hal_pic18	24/03/2022 7:28 p. m.	C Header File	24 KB

10.2 Funciones

FUNCION	USO
USBDeviceInit	Configura la Comunicación USB
USBDeviceTasks	Iniciar tareas de comunicación USB
USBEnableEndpoint	Habilita un Endpoint especifico
USBTransferOnePacket	Realiza una única transacción en el bus de datos USB
void USBStallEndpoint	Configura un Endpoint mediante el envío de STALL al Host
USBCancelIO	Cancela la transferencia pendiente
USBDeviceDetach	Configura modo "Soft Detach" para el Host
USBDeviceAttach	Activa las diferentes interrupciones establecidas para la Comunicación USB
USBCtrlEPAllowStatusStage	Selecciona el Endpoint requerido (IN o OUT) para permitir la transferencia de control
USBCtrlEPAllowDataStage	Permite la transferencia de datos
USBGetDeviceState	Identifica el estado del Device (activo o inactivo)
USBIsDeviceSuspended	Indica si el módulo USB está suspendido
USBIsBusSuspended	Indica si el bus USB está suspendido
USBHandleBusy	identifica si la salida está ocupada
USBHandleGetLength	Identifica el tamaño del buffer de entrada
USBHandleGetAddr	Recupera la dirección del destino del Buffer
USBTxOnePacket	Envía los datos al Endpoint especificado
USBRxOnePacket	Recibe los datos del Endpoint especificado
CDCSetBaudRate	Definir BaudRate
CDCSetCharacterFormat	Definir el formato de carácter
CDCSetParity	Definir Bit de Paridad
CDCSetDataSize	Definir # bits del bus de datos
CDCSetLineCoding	Define BaudRate, formato de carácter, Bit de parada y numero de bits del bus de datos
USBUSARTIsTxTrfReady	Revisa si se realiza transferencia o recepción de datos por CDC
mUSBUSARTTxRam	Transferencia de datos localizados en la memoria
mUSBUSARTTxRom	Transferencia de datos localizados en la memoria del programa
CDCInitEP	Inicializa el Driver de la función CDC
USBCheckCDCRequest	Revisa que los datos recibidos estén configurados como CDC
CDCNotificationHandler	Revisa los cambios de estatus y reportes del Host
getsUSBUSART	Copia un string de Bytes recibido por el bus USB CDC en la ubicación especificada por el usuario
putUSBUSART	Envía un arreglo por USB teniendo en cuenta la cantidad de datos y su tamaño de 8bits
putsUSBUSART	Envía una cadena de datos por USB desde la memoria RAM
putrsUSBUSART	Envía una cadena de datos por USB desde la memoria del programa
CDCTxService	Maneja las transferencias entre el Device y el Host
USBPowerModule	Aumenta el voltaje del USB
USBModuleDisable	Deshabilita el módulo USB
USBSetBDTAddress	Aumenta el voltaje del USB
USBClearInterruptFlag	limpia la bandera de interrupción

USBClearInterruptRegister	limpia el registro de interrupción
DisableNonZeroEndpoints	Limpia el registro de control de los non-zero endpoint
USBMaskInterrupts	Desactiva interrupción USB
USBUnmaskInterrupts	Activa interrupción USB
USBCtrlTrfSetupHandler	Inicia el control de tranferencia, llama y selecciona la clase (CDC, HID, etc),
USBCtrlTrfInHandler	Maneja las transferencias de entrada
USBCheckStdRequest	Revisa el paquete de datos setup para poder controlarlo
USBStdGetDscHandler	Maneja el descriptor GET
USBCtrlEPServiceComplete	Revisa los 3 tipos de transferencia (EP0-SETUP, EP0-OUT, EP0-IN)
USBCtrlTrfTxService	Controla las transferencias del Device al Host
USBCtrlTrfRxService	Controla las transferencias del Host al Device
USBStdSetCfgHandler	Deshabilita los Endpoint en el registro UEP
USBStdGetStatusHandler	Maneja el estándar GET-STATUS
USBStdFeatureReqHandler	Maneja los estándar SET, CLEAR y FEATURE
USBCtrlTrfOutHandler	Maneja las transferencias de salida
USBConfigureEndpoint	Configura el Endpoint
USBDisableInterrupts	Deshabilita la interrupción
SetConfigurationOptions	Configuración de USB
USBClearUSBInterrupt	Desactiva la bandera de la interrupción USB
USBSuspend	Se usa cuando el Host intenta suspender al Device
USBCtrlEPServiceComplete	Establece los Endpoint

10.3 Ponencia

DEVELOPMENT OF A USB SYSTEM TO IMPLEMENT IN PROGRAMMABLE DEVICES PIC18

DESARROLLO DE UN SISTEMA USB® PARA LA IMPLEMENTACIÓN EN DISPOSITIVOS PROGRAMABLES PIC18

Gabriela Alejandra Pardo Ortega, Msc. Robin Alfonzo Blanco Canon

Universidad ECCI

E-mail: {gabrielaa.pardoo, rblancoc}@ecci.edu.co

Abstract: This article focuses on the basic information of the USB® protocol and its implementation bases using the PIC18F4550, which allows the introduction to this technology for the optimization of communication between electronic devices, proposing the initial solution to the problem of the lack of information of this protocol for the implementation in the industries and a better teaching in the educational institutions so that there is more consolidation for general use in any area of society. Also, it's proposed the implementation of this communication in an exercise focused in control area.

Keywords: Communication, USB®, Speed, Development, Operation, Control.

Resumen: Este artículo se centra en la información básica del protocolo de comunicación serial USB® y las bases de implementación del mismo por medio del PIC18F4550, que permite la introducción a esta tecnología para la optimización de la comunicación entre dispositivos electrónicos, planteando la solución inicial al problema de la falta de información del mismo para la implementación en las industrias y la mejor enseñanza en las instituciones educativas de manera que haya más afianzamiento para uso general en cualquier ámbito de la sociedad. Adicionalmente, se propone la implementación de esta comunicación en un ejercicio enfocado hacia área de control.

Palabras clave: Comunicación, USB®, Velocidad, Desarrollo, Funcionamiento, Control.

1. INTRODUCCIÓN

El protocolo USB® (Universal Serial Bus) es un protocolo de comunicación serial que se originó a mediados de la década de los 90 con la función de facilitar la conexión de dispositivos al computador o a un equipo determinado. Este es uno de los más conocidos actualmente a nivel comercial y uno de los más utilizados. Este protocolo ha tenido una gran evolución hasta la actualidad generando versiones cada vez más rápidas y con una mejor transmisión. Llegando a alcanzar velocidades de hasta 40Gb/s, volviéndose así un protocolo de gran utilidad. (Tecnologia&informática, 2019)

Entre las diferentes versiones de USB® están USB1.0® que fue la primera versión oficial de USB® con una velocidad de 1.5 Mb/s que era una velocidad bastante eficiente para esta época en la que se produjo por lo que revolucionó todos los protocolos de comunicación. La segunda versión USB1.1® añadió a esto la ventaja de *plug and play* que permite gran facilidad en la conexión y desconexión de los dispositivos periféricos.

Posteriormente, se originaron las versiones USB2.0®, USB3.0®, USB3.1® y USB3.2® mejorando cada vez la velocidad de transmisión siendo estas las utilizadas hoy en día en dispositivos como el pendrive e impresoras. La

versión más reciente es la USB4.0® que permite velocidades de hasta 40 Gb/s (Gómez, 2021)

Sin embargo, a pesar de sus ventajas en velocidad de transmisión y recepción de datos, su uso en las industrias está muy limitado debido al poco conocimiento que tienen los empleados de estas sobre su funcionamiento, por tanto, se debe reconocer que es de gran importancia el aprendizaje de este tema para mejorar la calidad de los procesos industriales y dispositivos comerciales.

Hay que considerar de igual manera la escasa información que se encuentra sobre el uso de este protocolo para su aplicación en sistemas de bajo costo, como los microcontroladores; teniendo en cuenta principalmente en el PIC18F4550 que es uno de los microcontroladores más accesibles y asequibles para la implementación de este tipo de comunicación, sin menospreciar otras comunicaciones de este PIC como RS232, SPI e I2C.

Por esta razón, en este proyecto se propone la recopilación y actualización de información además de la exploración e implementación del protocolo de comunicación USB® mediante el uso del PIC18F4550 y el compilador XC8® de manera tal que se permita la comprensión de las librerías utilizadas además de recompilar y explicar los conceptos básicos de la comunicación USB®, teniendo en cuenta características fundamentales como su funcionamiento, velocidad, conexión, practicidad y aplicaciones en diferentes áreas de la ingeniería, para garantizar un mayor rendimiento de esta en la industria y los centros educativos.

2. MARCO TEÓRICO

2.1 Microcontrolador PIC18F4550

Es un microcontrolador fabricado por Microchip®, funciona a 8 bits, tiene una arquitectura tipo Harvard y pertenece a la gama medio-alta de esta empresa. Además, tiene una muy buena capacidad de procesamiento y su consumo de energía no es muy elevado. Cuenta con 40 pines y seis puertos que permiten realizar diferentes operaciones (MICRO CONTROLADORES, 2022.), véase figura 1.

Este microcontrolador cuenta con otras características como lo son una frecuencia de oscilación entre 31kHz y 8MHz haciendo uso de su oscilador interno o de hasta 48MHz haciendo uso de un oscilador externo, además permite el uso de PLL (Phase Lock Loop) para la implementación de frecuencias mayores.

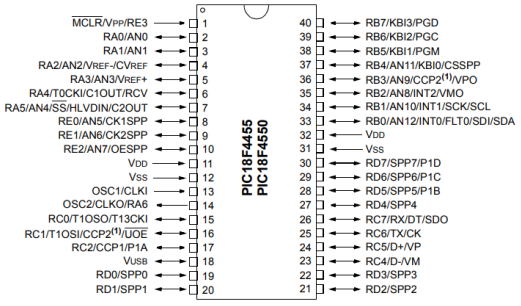


Fig. 1. PIC18F4550. Tomada de PIC18F2455/2550/4455/4550 Data Sheet

Entre sus funciones se encuentran los módulos de comunicación USB®, cuatro Timer, tres interrupciones externas, dos módulos CCP (Capture/Compare/PWM), el módulo USART y MSSP que diferentes tipos de comunicación serial con otros dispositivos, un conversor análogo digital, entre otras funciones que hacen de este un dispositivo de gran utilidad (Microchip Technology Inc., 2006).

2.2 Comunicación USB®

El protocolo de comunicación USB® (Universal Serial Bus) es uno de los estándares de comunicación más utilizados comercialmente en la actualidad ya que en este momento la mayoría de periféricos o dispositivos que se conectan al computador como lo son cámaras, impresoras, escáneres, entre otros utilizan este tipo de comunicación (Weis, 2021).

Se caracteriza por su facilidad de uso ya que se es muy versátil y se utiliza en una gran cantidad de dispositivos además de su característica de *plug and play* que permite la desconexión en caliente, es decir, que se puede conectar o desconectar en cualquier momento y forma sencilla soportando a su vez hasta 127 dispositivos conectados. Otra ventaja que presenta es la alimentación ya que se tiene un suministro de energía para los dispositivos de hasta 500mA (crisross, 2013).

Su estructura consiste en un *Host* (elemento que actúa como elemento principal o controlador del sistema USB®) el cual pude tener dos o tres puertos (zócalo a través del cual se accede a la red USB®) donde se pueden conectar ya sea un Hub (dispositivo que expande la cantidad de puertos disponibles) o un *Device* (dispositivos que van a interactuar con el *Host*) y finalmente al Hub irán conectados los *Device* que se vayan a utilizar (Electronics notes, 2022), véase figura 2.

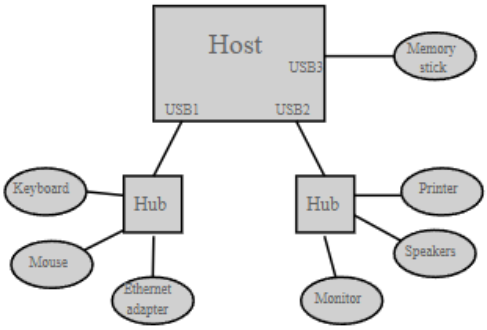


Fig. 2. Estructura típica de una red USB®. Tomada de https://www.electronics-notes.com/

La comunicación USB® está basada en transferencia por *Data Pipes* que son los conductos por los cuales viaja la información para ello se tienen dos tipos de *Pipes* los de mensaje que se encargan de llevar la información de *Host* al *Device* y viceversa y los de corriente que van solo en dirección al *Host* y se encargan de las transferencias de información isócronas, de interrupción o masivas que vienen de los *Endpoints* (Búfer de almacenamiento) del *Device* (Electronics notes, 2022), véase figura 3.

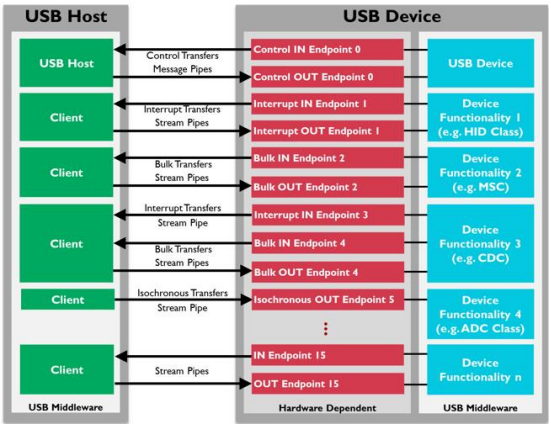


Fig. 3. Estructura de Transferencia de datos por USB®. Tomada de www.keil.com

2.2 Conectores USB®

Con el desarrollo de las diferentes versiones del estándar USB han ido apareciendo diferentes conectores entre los que se destacan:

- Tipo A: Es el primer conector USB® implementado con la versión USB1.0®, inicialmente fue implementado con una línea blanca. Sin embargo, a partir de la versión USB3.0® se empezó a utilizar una línea azul para diferenciarlo de las versiones anteriores. (Roca, 2021)
- Tipo B: Este conector tiene una forma cuadrada con dos de las esquinas redondeadas y sus pines están ubicados dos arriba y dos abajo, se utiliza en las versiones USB2.0® y USB3.0®, sus principales aplicaciones se encuentran en impresoras y escáneres (Profesional Review, 2019)
- Mini USB®: Este conector es más pequeño que el estándar (tipo A) y tiene un pin extra conocido como el identificador ID el cual está diseñado para una mejora en el puerto Se usa especialmente con la versión USB2.0® (Noguera, 2021)
- Micro USB®: Este conector se caracteriza por su tamaño pequeño. Sus usos más comunes actualmente son algunos smartphones y se utilizan con las versiones USB2.0® y USB3.0®, en esta última versión tienen un pin extra como una extensión a un lateral del conector estándar (López, 2020)
- Tipo C: Es un conector pequeño y ovalado, es conocido también como conector reversible ya que puede conectarse en ambos sentidos a diferencia de las versiones anteriores, por lo tanto, desde su salida al mercado se ha intentado convertir en el conector estándar para USB®. Se usa en las versiones USB3.1® en adelante y tiene ofrece mayor velocidad en transferencia de datos e intensidad en la corriente eléctrica (WhistleOut MX, 2021)

2.3 Versiones USB®

A lo largo de los años han salido al público varias versiones del protocolo USB® como se muestra en la Tabla 1 (CMD ltd, 2022)

Tabla 1: Versiones USB®. Tomada de https://www.cmd-ltd.com/

Versión	Año	Nombre	Velocidad
USB1.0®	1996	Full Speed	12Mbits/s
USB1.1®	1998	Full Speed	12Mbits/s
USB2.0®	2000	High Speed	480Mbits/s
USB3.0®	2008	SuperSpeed	5Gbits/s
USB3.1®	2013	SuperSpeed+	10Gbits/s
USB3.2®	2017	SuperSpeed+	20Gbits/s
USB4.0®	2021		40Gbits/s

2.4 Clases de USB®

Las clases del protocolo USB® se utilizan para definir la función que cumplen los dispositivos de tal manera que se pueda identificar con mayor facilidad el *Driver* que se requiere para realizar la comunicación de manera correcta (USB-IF, 2022). Entre las más destacadas se encuentran:

- *Communication Device Class* (CDC): Esta interfaz se genera mediante la emulación de puertos virtuales COM

usando el modelo de control abstracto (armKeil, 2021)

- *Human Interface Device* (HID): Esta clase se utiliza especialmente para dispositivos que tienen una interacción directa con las personas para controlar los sistemas de cómputo (USBlyzer, 2006)
- *Mass Storage Class* (MSD): Esta clase permite la transferencia de inmensas cantidades de información entre los dispositivos que se realiza la comunicación (Technopedia, 2022)

2.5 Sensor DHT11

Es un sensor usado medir temperatura en un rango de 0°C a 50° C y humedad en un rango de 20% a 90% con una resolución de 16 bits. Este tiene un NTC con el cual toma la medida de la temperatura y un microcontrolador con 8 bits de salida para los datos medidos con una exactitud entre ±1°C y ±1% (Components101, 2021)

2.6 Dimmer Digital

Es un dispositivo usualmente usado para regular la intensidad de la luz con bombillos que utilizan corriente alterna (AC). Estos regulan el voltaje RMS recibido por el bombillo para de esta manera modular su brillo. La mayoría incluyen un encendido "paso por cero" de la tensión mediante la cual se referencia el corte de la señal de salida obteniendo una onda de entre 60 voltios y 200 voltios según la intensidad de luz que se desee. (Aitor, 2021)

2.8 Raspberry Pi

Es un computador de pequeñas dimensiones que se adapta al uso de periféricos de entrada y salida como un mouse, teclado y pantalla para interactuar con el usuario (Rodríguez, 2018). Sirve especialmente para la implementación de prototipos, utiliza sistemas operativos GNU/Linux como Raspbian y tiene una arquitectura de procesador ARM distinta a la de los ordenadores de mesa ya que puede realizar tareas con menor consumo de energía (programoergosum, 2021)

2.9 Logica Difusa

La lógica difusa parte de la imposibilidad de definir con total claridad una afirmación como verdadera o falsa teniendo en cuenta que según el punto de vista puede variar su estado y considerando los puntos intermedios que se puedan llegar a presentar en dicha afirmación. Por ende, los conjuntos a evaluar se toman de manera lingüística donde determinados elementos pueden pertenecer a más de un conjunto a la vez, de manera tal que dichos elementos tienen un rango de pertenencia al conjunto definido por un valor numérico entre 0 y 1 donde "0" significa que no pertenece al conjunto en absoluto y "1" significa una total pertenencia. (Ruvalcaba y Vermonden, 2015)

3. METODOLOGIA

Inicialmente se utiliza el programa propuesto en el foro TODOPIC (jfmateos2, 2011) en el cual, se presenta un ejemplo de comunicación USB-CDC® utilizando el microcontrolador PIC18F2455 y el compilador C18®.

Teniendo en cuenta el objetivo de implementar dicho programa en el compilador XC8® se realiza la búsqueda de otros ejemplos que permitan la adaptación de este. Entre estos se encuentra un ejemplo con dicho compilador utilizando el PIC16F1455 y el complemento de MPLAB® X, MPLAB® Code Configurator (MCC) (pablinza, 2019). De esta manera al explorar los dos ejemplos se realiza un primer programa con la estructura mostrada en la figura 4, el cual realiza la comunicación USB-CDC® entre el microcontrolador y el computador, siendo este la base del proyecto en general.

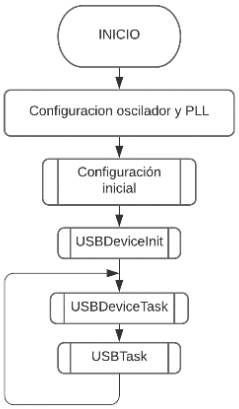


Fig. 4. Programa base. Fuente propia

Este programa utiliza las librerías generadas por el MPLAB® Code Configurator para los microcontroladores PIC18FxxKxx donde se destacan las funciones USBDeviceInit y USBDeviceTask las cuales se encargan de la inicialización de la comunicación y evitar la desconexión del dispositivo respectivamente. Por otra parte, se tiene en cuenta que la frecuencia de trabajo del microcontrolador para realizar dicha comunicación debe ser de 48MHz, por lo tanto, para este programa se utiliza el módulo PLL (*Phase Locked Loop*) del microcontrolador que permite elevar la frecuencia proporcionada por el cristal externo de 20MHz. De igual manera, Se tienen las funciones de configuración inicial con la cual se configuran los módulos a usar y la

función USBTask presentada en la figura 5, que es la encargada de realizar la comunicación de ejemplo.

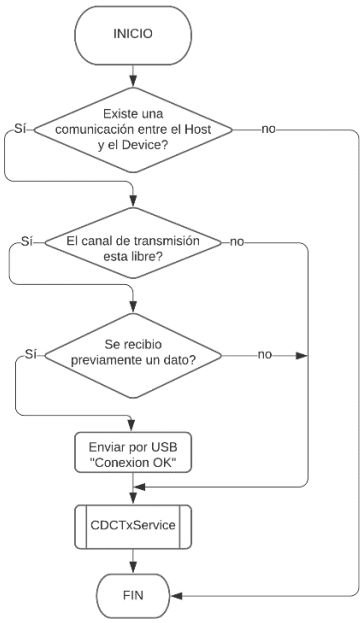


Fig. 4. Función USBTask. Fuente propia

En la función USBTask se tienen en cuenta dos funciones importantes getsUSBUSART y putrsUSBUSART que son las encargadas de enviar y recibir los datos de la comunicación.

Para la implementación de este primer programa se debe realizar el montaje de la Figura 6, donde se aprecia la unión del microcontrolador con el conector USB®, además de otros requerimientos como el cristal, la alimentación y el condensador del pin Vbus que se requiere para realizar la comunicación.

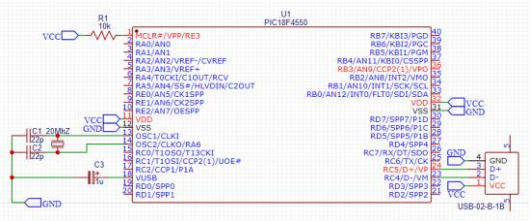


Fig. 6. Montaje de programa base. Fuente propia

Con el fin de mejorar la funcionalidad del programa base se cambia el modo de operación de POLLING a INTERRUPT para así evitar que se pueda interrumpir la comunicación entre el microcontrolador y el computador o la Raspberry Pi, véase figuras 7 y 8.

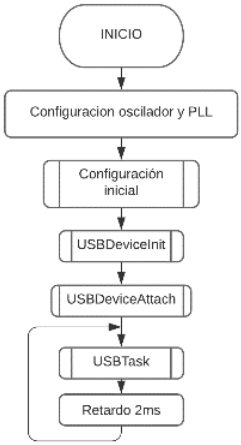


Fig. 7. Programa base por interrupción. Fuente propia

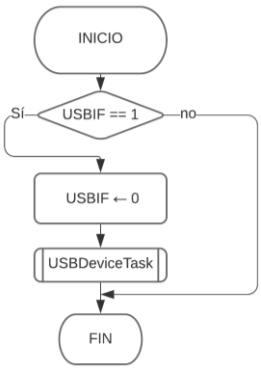


Fig. 8. Interrupción USB. Fuente propia

Una vez que empieza a funcionar el programa base se da una aplicación a este en el área de control utilizando lógica difusa para controlar la temperatura generada por un bombillo en una caja, para lo cual se utiliza un sensor DHT11 y un Dimmer digital y se realiza una visualización de la temperatura por medio de la comunicación USB® con una Raspberry Pi 4, véase figura 9.

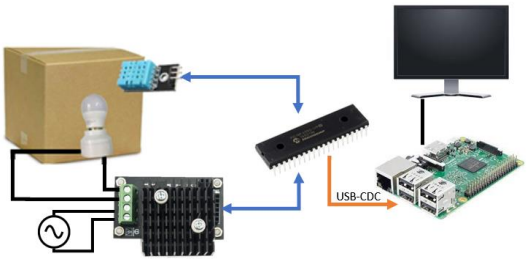


Fig. 9. Estructura ejercicio de Control difuso. Fuente propia

Para este ejercicio se plantea inicialmente una función que permita leer correctamente el sensor DHT11, véase figura 10, teniendo en cuenta que para realizar la comunicación con este dispositivo se requiere enviar una señal hacia el sensor que indique el inicio de la comunicación, en respuesta el DHT11 envía una señal de 40 bits que incluye los datos de temperatura y humedad.

Finalmente, se debe enviar otra señal que indique el recibido de la información e indique al sensor que debe volver al modo de espera. Además, al iniciar la comunicación se debe cambiar el nivel de voltaje del bus de alto a bajo en un tiempo de aproximadamente 18ms y en respuesta se recibirán pulsos de un tiempo entre 20us (0 lógico) y 40us (1 lógico) (Mouser Electronics, Inc., 2022)

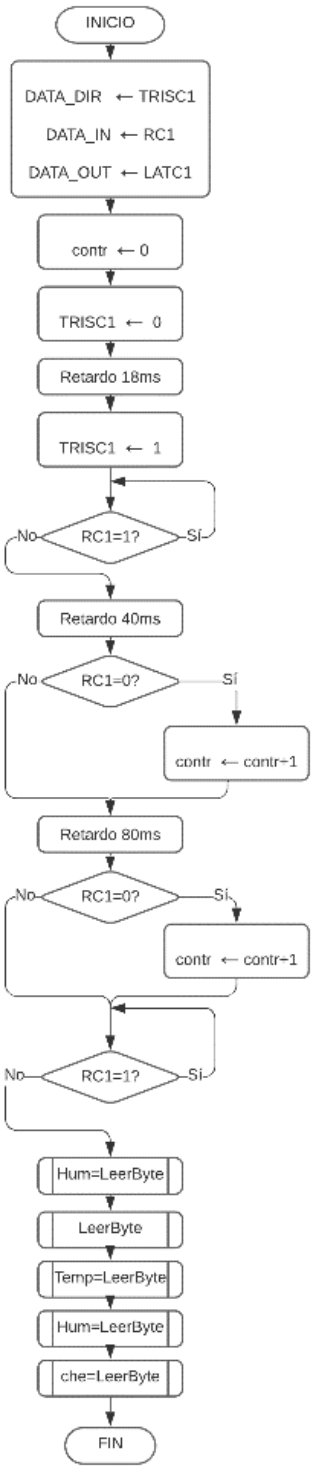


Fig. 10. Función de lectura del sensor DHT11. Fuente propia

Una vez planteada la función del sensor se procede a plantear dos interrupciones que en conjunto permiten controlar la intensidad del bombillo por medio del Dimmer digital. Para la primera función se presenta una interrupción de entradas y salidas que se activa por la señal del pin SYNC (cruce por cero) conectado al pin RB0 del microcontrolador, véase figura 11, en este se cuentan dos ciclos para determinar el inicio ya que se tiene un cruce por cero para el cambio entre la señal positiva y negativa y otro para la acción contraria.

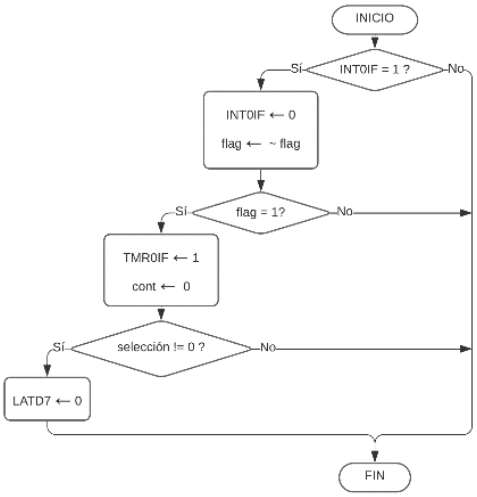


Fig. 11. Función de interrupción para Dimmer digital. Fuente propia

Además, se realiza una interrupción de TIMER0 llevando un periodo de 8,33 ms para subdividir el periodo de la señal análoga en 20 partes y así obtener 20 intensidades de luz diferentes, véase figura 12. Estas intensidades serán controladas por el valor de la variable "selección" que tomará un valor entre 0 y 19 según se establezca en el programa principal.

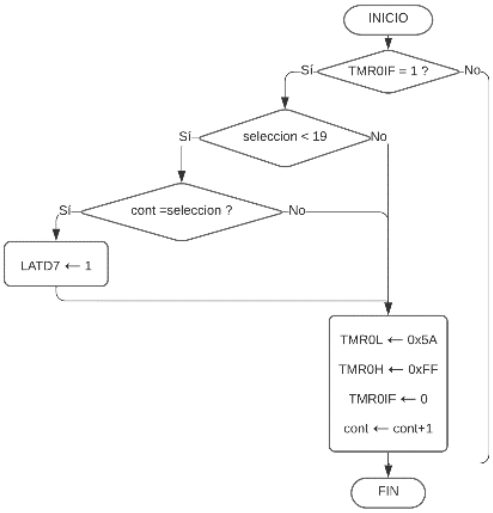


Fig. 12. Interrupción de TIMER0 para Dimmer digital. Fuente propia

Finalmente, se plantean las funciones de control difuso en el cual su única entrada será el valor leído por el sensor de temperatura que una vez implementado el control generará una salida correspondiente a la intensidad necesaria en el bombillo para llegar a la referencia, para esto se plantea el programa de la figura 13.

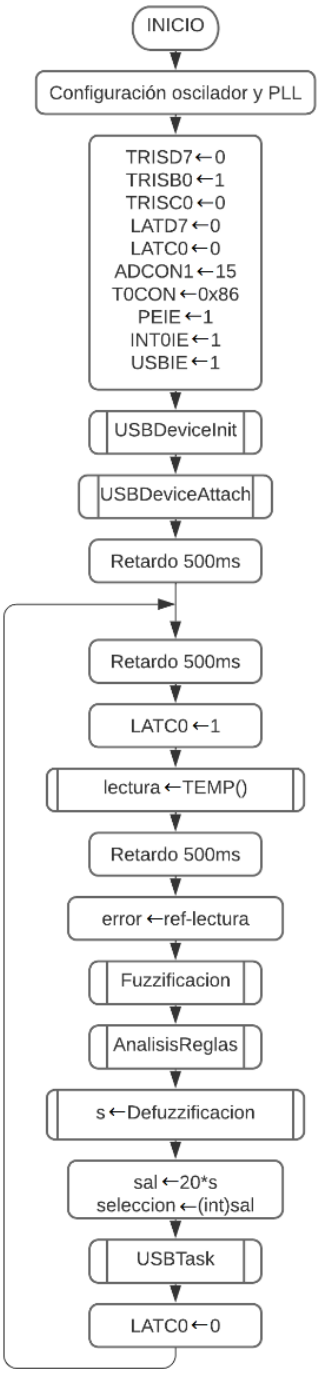


Fig. 13. Programa de control difuso. Fuente propia

En este programa la función de Fuzzificación está dada por las funciones mostradas en la figura 14 donde función azul corresponde a un valor negativo del error, la función verde a un valor cero y la función roja a un valor positivo. Teniendo en cuenta estas funciones, en el análisis de reglas se hace una asignación a cada una de las tres reglas establecidas para cada una de las tres funciones mencionadas anteriormente.

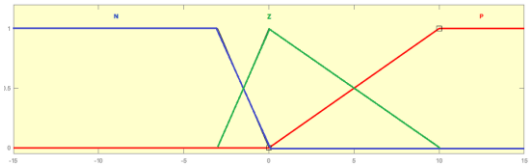


Fig. 14. Función de Fuzzificación. Fuente propia

Por otro lado, en la Defuzzificación se da un valor correspondiente a la acción de control que se realizará según la temperatura evaluada, véase figura 15.

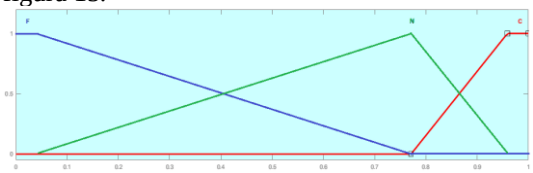


Fig. 15. Función de Defuzzificación. Fuente propia

Por último, la función USBTask realiza la comunicación con la Raspberry para enviar la información requerida para la interfaz de usuario y recibir de igual manera la referencia de la temperatura., véase figura 16.

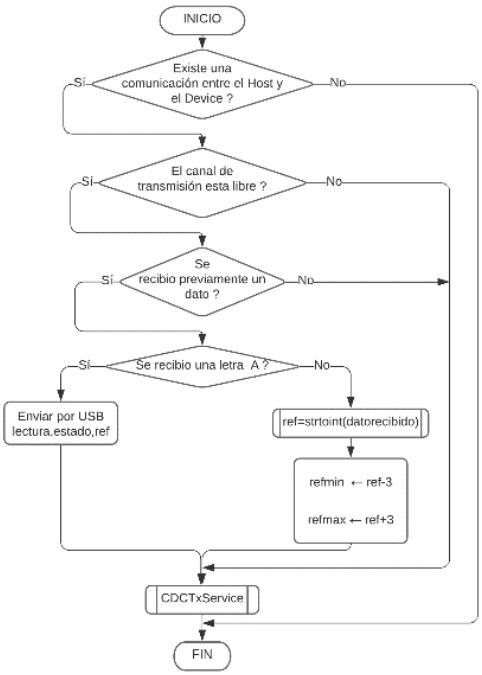


Fig. 16. Función USBTask para programa de control. Fuente propia

Seguidamente, se realiza una adaptación al programa para obtener una mayor precisión teniendo como variables de entrada la

temperatura y su derivada, Ecuación 1, teniendo un tiempo de muestreo de 1 segundo.

$$dE = \frac{e_{actual} - e_{anterior}}{t} \quad (1)$$

Para la función de Fuzzificación se utilizan las funciones mostradas en las figuras 17 y 18 donde la función azul indica un error negativo, la verde un error cero y la roja un error positivo; de igual manera, la función morada indica una derivada del error negativo, la amarilla una derivada cero y la anaranjada una derivada positiva.

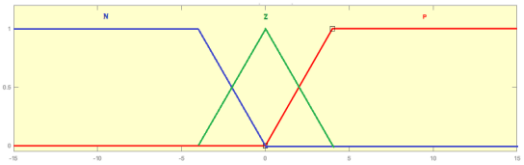


Fig. 17. Entrada de error. Fuente propia

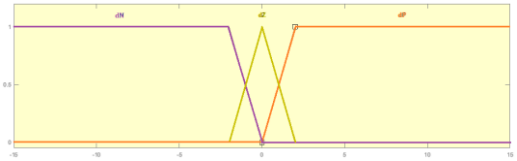


Fig. 18. Derivada del error. Fuente propia

Teniendo en cuenta lo anterior para el análisis de reglas se aplica la Tabla 2, donde según el valor indicado del error y la derivada se aplicará una de las reglas establecidas, teniendo nueve posibles opciones. Finalmente, la Defuzzificación se aplica bajo las mismas funciones mostradas para el programa anterior, véase figura 15.

Tabla 2: Análisis de reglas para control difuso de dos entradas.

		DERIVADA		
		N	Z	P
ERROR	N	N	N	Z
	Z	N	Z	P
	P	Z	P	P

Para la aplicación de la comunicación USB-CDC® a este ejercicio se realiza una interfaz utilizando el Software Python y su librería Tkinter, además de las librerías serial para la comunicación con el PIC®, numPy para el uso de vectores, math para las operaciones matemáticas y time para funciones relacionadas con el tiempo, véase figura 19.

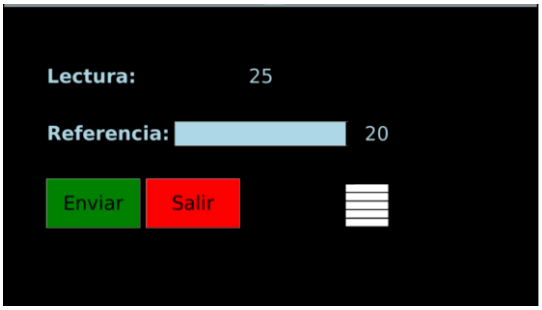


Fig. 19. Interfaz de Python. Fuente propia

Esta interfaz incluye un espacio donde se muestra el dato de lectura de la temperatura actual, un espacio para la escritura de la referencia deseada en caso de que este sea el método de entrada del programa seguido por un cuadro de texto que muestra la referencia actual, además de dos botones que permiten enviar la referencia digitada al microcontrolador, véase figura 20 y salir del programa, véase figura 21 y finalmente un indicador para saber la intensidad de encendido del bombillo por medio de barras y un procedimiento que define el color de cada una de ellas teniendo en cuenta 4 tonos (blanco, amarillo, naranja y rojo). Los datos de temperatura y referencia actual se actualizan en cada siclo del programa al igual que el indicador de estado, véase la figura 22.

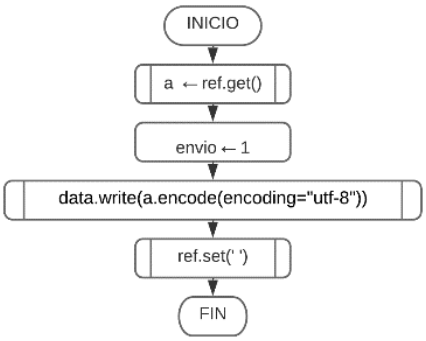


Fig. 20. Función de botón Enviar. Fuente propia

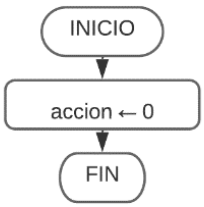


Fig. 21. Función de botón Salir. Fuente propia

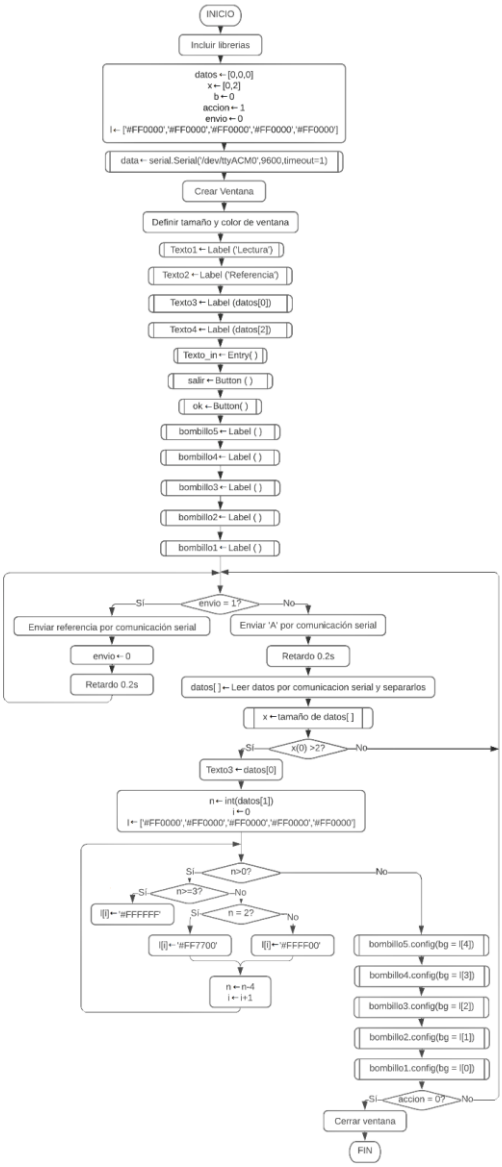


Fig. 22. Programa Interfaz Python. Fuente propia

4. RESULTADOS

Inicialmente, se obtiene un programa ejemplo con el microcontrolador PIC18F45K50 generado mediante el MPLAB® Code Configurator, de manera tal que haciendo uso de las librerías generadas para el módulo USB-CDC® y tomando como ejemplo el programa implementado en C18® se obtiene el programa base para este proyecto el cual utiliza un puerto COM para realizar la comunicación, véase figura 23 e ingrese a https://youtu.be/aiisQ7gHnjI. A partir de este programa se comprende la estructura y funcionamiento de las diferentes funciones aplicadas.

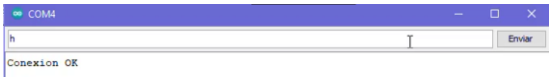


Fig. 23. Funcionamiento programa base. Fuente propia

En el ejemplo de control difuso se observa un buen funcionamiento en cuanto a la variación de la intensidad del bombillo respecto a la temperatura presentada en la caja, pero al inicio con el uso de una sola entrada (error) se presenta una pequeña desestabilidad de la temperatura, por lo cual se opta por añadir una segunda entrada (derivada del error) que le da una mayor estabilidad al control del sistema. En cuanto a la interfaz, su tiempo de respuesta no es tan rápido, pero es apropiado para lo requerido frente a la visualización de la información, ingrese a https://youtube.com/playlist?list=PLg3WXOmGn8LUXlR3ml6kZ9Wwz_ObvTbtQ.

5. RECONOCIMIENTO

Agradecemos el apoyo del programa de Ingeniería Mecatrónica de la Universidad ECCI sede Bogotá

6. CONCLUSIONES

- Los microcontroladores PIC18 son una herramienta muy funcional a la hora de realizar un proyecto de ingeniería ya que tienen un bajo costo y una gran variedad de funciones que permiten la interacción con múltiples dispositivos electrónicos.
- La clase USB-CDC® es una buena opción para realizar un primer acercamiento a la programación de este protocolo de comunicación, ya que de manera general es la más sencilla de aplicar debido a la manera de uso.
- Se vuelve muy importante aprender cómo funciona la comunicación USB® para tener un mayor alcance en los diferentes proyectos a implementar, debido a su alta cantidad de aplicaciones además de su funcionalidad y velocidad.
- La comunicación USB® es más optima utilizándola por el método de interrupción (INTERRUPT) ya que hay más control sobre el tiempo de respuesta. Sin embargo, el modo sondeo (POLLING) permite otras aplicaciones más sencillas y requiere menos instrucciones de inicio.
- El control difuso presenta una mayor precisión al utilizar más entradas (factores a evaluar) ya que se tiene más certeza del funcionamiento del sistema.

REFERENCIAS

Microchip Technology Inc. (2006). *PIC18F2455/2550/4455/4550 Datasheet*.

Ruvalcaba F. J. y Vermonden A. (2015*). Lógica difusa para la toma de decisiones y la selección de personal, Universidad del Rosario. Universidad & Empresa, vol. 17, nº 29, pp. 239-256*, Colombia.

Mouser Electronics, Inc. (2022). *DHT11 Humidity & Temperature Sensor*, Mansfield, Texas USA

SUTAGO (2022) *Dimmer Digital TRIAC BTA312*

Axelson, J. (2005). *USB Complete: Everything You Need to Develop Custom USB Peripherals. Third Edition,* Madison: Lakeview Research, U.S.

SITIOS WEB

Tecnologia&Informatica (2019). *Historia y evolución USB.* https://www.tecnologia-informatica.com/historia-evolucion-usb/

Gómez, B. (2021). *Descúbrelo todo sobre USB 4, el estándar basado en Thunderbolt, Profesional Review.* https://www.profesionalreview.com/2021/04/25/usb-4/

MICRO CONTROLADORES. (2022). *Microchip PIC18F4550.* https://microcontroladoress.com/pic18f4550/

Weis, O. (2021). *What is called "USB"?, Flexihub.* https://www.flexihub.com/all-about-usb.html#usb

Crisross (2013). *PROTOCOLO USB, ARQUITECTURA DE COMPUTADORAS.* https://arquitecturadecomputadora.wordpress.com/2013/05/21/protocolo-usb/

Electronics notes. (2022). *USB Operation: Protocol, Data Transfer & Packets, Radio-Electronics.* https://www.electronics-notes.com/articles/connectivity/usb-universal-serial-bus/protocol-data-transfer.php

Akpan, V. (2022). *¿Qué es un USB OTG y qué cosas increíbles puedes hacer con él?, suntrustblog*. https://suntrustblog.com/es/what-is-a-usb-otg/

Roca, J. (2021). *¿Qué tipos de conectores USB existen y en qué se diferencian?, hardzone*. https://hardzone.es/tutoriales/rendimiento/conector-usb-tipos/

Profesional Review. (2019). *USB: Qué es, tipos, formatos y velocidades*. https://www.profesionalreview.com/conectores-pc/usb/#Como_funciona_un_USB

Noguera, B. (2021). *Cuál es la diferencia entre un micro USB y un mini USB, Culturación.* https://culturacion.com/cual-es-la-diferencia-entre-un-micro-usb-y-un-mini-usb/

López, P. (2020). *¿Qué es Micro USB y para qué sirve?, Geeknetic.* https://www.geeknetic.es/Micro-USB/que-es-y-para-que-sirve

WhistleOut MX (2021). *¿Qué es USB Tipo C y qué beneficios tiene?* https://www.whistleout.com.mx/CellPhones/Guides/usb-tipo-c

CMD ltd (2022). *USB DATA TRANSFER GUIDE*. https://www.cmd-ltd.com/advice-centre/usb-chargers-and-power-modules/usb-and-power-module-product-help/usb-data-transfer-guide/

USB-IF (2022). *Defined Class Codes*. https://www.usb.org/defined-class-codes

armKeil (2021). *CDC: Communication Device Class.* https://www.keil.com/pack/doc/mw/USB/html/_c_d_c.html

USBlyzer (2006). *Human Interface Device (HID) Class Decoder*. http://www.usblyzer.com/usb-human-interface-device-hid-class-decoder.htm

Technopedia (2022). *What Does Mass Storage Device (MSD) Mean?* https://www.techopedia.com/definition/11901/mass-storage-device-msd

Velasco, R. (2021). *Exprime al máximo tu Raspberry Pi con estos sistemas operativos*. https://www.softzone.es/programas/sistema/mejores-sistemas-operativo-raspberry-pi/

BeJob (2016). *7 RAZONES PARA PROGRAMAR EN PYTHON*. https://www.bejob.com/7-razones-para-programar-en-python/

Componentes101 (2021). *DHT11–Temperature and Humidity Sensor*. https://components101.com/sensors/dht11-temperature-sensor

Aitor (2021). *DIMMER- QUE ES, TIPOS Y COMO FUNCIONA UN ATENUADOR DE LUZ, ERENOVABLE*. https://erenovable.com/que-es-un-dimmer/

Rodríguez de Luis, E. (2018). *De cero a maker: todo lo necesario para empezar con Raspberry Pi, Xakata*. https://www.xataka.com/makers/cero-maker-todo-necesario-para-empezar-raspberry-pi

Programoergosum (2021). *Introducción a Raspberry Pi*. https://www.programoergosum.es/tutoriales/introduccion-a-raspberry-pi/

jfmateos2 (2011). *TODOPIC*. http://www.todopic.com.ar/foros/index.php?PHPSESSID=qcrpvrhug17lm510penrim03m3&topic=33699.0

pablinza (2019). *ENTRENAMIENTO PIC CON XC8*. https://pablinza.blogspot.com/2019/07/usb-serial-cdc-con-pic16f.html