UACM

Universidad Autónoma
de la Ciudad de México

NADA HUMANO ME ES AJENO

COLEGIO DE CIENCIA Y TECNOLOGÍA
LICENCIATURA EN INGENIERÍA EN SISTEMAS
ELECTRÓNICOS Y DE TELECOMUNICACIONES

Comparación de Tecnologías para el IoT: PIC, Arduino y Raspberry Pi

T E S I S

QUE PARA OPTAR POR EL TÍTULO DE

LICENCIADAS EN INGENIERÍA EN SISTEMAS ELECTRÓNICOS Y DE TELECOMUNICACIONES

P R E S E N T A N :

VALERIA JENNY HERNÁNDEZ DE LA ROSA
DIANA LAURA PÉREZ SANTAMARÍA

D I R E C T O R: **M. EN I. LUIS ENRIQUE ARANDA MELO**

C O D I R E C T O R: **ING. ARIEL IVÁN LÓPEZ JUÁREZ**

Ciudad de México, junio de 2024.

UACM
Universidad Autónoma de la Ciudad de México
Nada humano me es ajeno

CA Coordinación Académica

SISTEMA BIBLIOTECARIO DE INFORMACIÓN Y DOCUMENTACIÓN

UNIVERSIDAD AUTÓNOMA DE LA CIUDAD DE MÉXICO
COORDINACIÓN ACADÉMICA

RESTRICCIONES DE USO PARA LAS TESIS DIGITALES

DERECHOS RESERVADOS©

Dale vida a los sueños

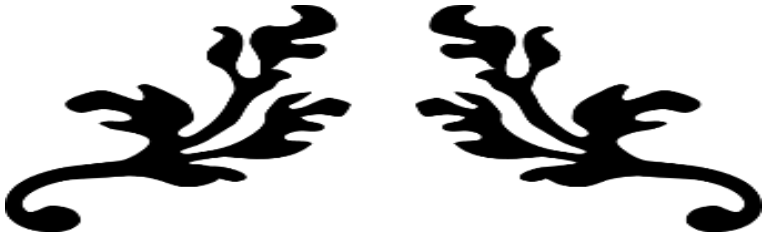
Dale vida a los sueños que alimentan el alma,
No los confundas nunca con realidades vanas.
Y aunque tu mente sienta necesidad, humana,
De conseguir las metas y de escalar montañas,
Nunca rompas tus sueños, porque matas el alma.

Dale vida a tus sueños aunque te llamen loco,
No los dejes que mueran de hastío, poco a poco,
No les rompas las alas, que son de fantasía,
Y déjalos que vuelen contigo en compañía.

Dale vida a tus sueños y, con ellos volando,
Tocaras las estrellas y el viento, susurrando,
Te contara secretos que para ti ha guardado
Y sentirás el cuerpo con caricias, bañado,
Del alma que despierta para estar a tu lado.

Dale vida a los sueños que tienes escondidos,
Descubrirás que puedes vivir estos momentos
Con los ojos abiertos y los miedos dormidos,
Con los ojos cerrados y los sueños despiertos.

Mario Benedetti



Agradecimientos

Quisiera expresar mi agradecimiento a la Universidad Autónoma de la Ciudad de México, pero sobre todo a cada profesor por el apoyo, la guía y las oportunidades que me han brindado a lo largo de mi trayectoria estudiantil, he tenido la oportunidad de expandir mis horizontes, explorar nuevas áreas de conocimiento y desarrollar habilidades que me serán de gran utilidad en el futuro.

Mi más profundo agradecimiento a mis padres, Ernesto y Rosa, por su amor, apoyo y sacrificio inquebrantable durante mi trayecto académico, ustedes han sido mis pilares y mis mayores motivadores. Su constante animo ha sido el impulso que necesitaba para superar los desafíos dando como resultado la culminación de esta tesis. Su ejemplo de dedicación, sacrificio y determinación ha sido una constante inspiración para mí, ya que, me han enseñado el valor del trabajo arduo, la importancia de ser perseverante y la gratitud por las oportunidades que se presentan en la vida.

No hay palabras suficientes para expresar mi agradecimiento por todo lo que han hecho por mí a lo largo de los años. Esta tesis no solo es el reflejo de mi esfuerzo, sino también del amor, dedicación y sacrificio que ustedes han invertido en mi educación y desarrollo personal. Por todo esto y más, les estoy y estaré eternamente agradecida.

A mis hermanas Norma y Monica por su constante apoyo, amor y animo a lo largo de mi trayectoria estudiantil. Su presencia ha dado luz a mi camino en algunos momentos difíciles y ha enriquecido cada etapa de mi vida, incluyendo este importante momento. Admiro la fortaleza, inteligencia y el carácter excepcional de cada una de ustedes. Su ejemplo de dedicación, trabajo y amor incondicional son una inspiración constante para mí.

Con gran estima y reconocimiento, quiero expresar mi más sincera gratitud a mi director y codirector, el M. en I. Luis Enrique Aranda Melo y a el Ing. Ariel Iván López Juárez, gracias a su dedicación docente y a su magnífica guía que han sido dos de los grandes pilares para la dirección y el enriquecimiento de esta tesis. Agradezco sinceramente el tiempo, energía y esfuerzo que han invertido en este trabajo, admiro profundamente su pasión por la enseñanza, así como su compromiso con el éxito de sus estudiantes.

De igual forma quiero agradecer a los lectores, la M. en I. Patricia Hong Cirión por sus consejos, por la confianza que nos brindó al prestarnos un lugar en el cual pudimos desarrollar nuestra tesis. Al M. en I. Víctor Manuel Macías Medrano por su amabilidad, por prestarnos la Raspberry Pi y además de otros elementos que fueron parte fundamental para la realización de este trabajo y al Lic. Jorge Mendoza Zavala por sus comentarios constructivos y las sugerencias que nos ha proporcionado.

Gracias por su tiempo, esfuerzo y conocimiento, que han dedicado para revisar y evaluar este trabajo, su compromiso con sus estudiantes y su apoyo durante este proceso han sido de gran ayuda.

Al Ing. Oscar Abraham Ocampo Rojas quiero expresar mi gratitud por su constante apoyo en estos últimos semestres de la carrera, pues me ha alentado a alcanzar mi máximo potencial. Su compromiso por la excelencia educativa y su pasión por la enseñanza han sido fuente de

inspiración para culminar este proyecto. Su influencia positiva a lo largo de la carrera será siempre fuente de inspiración y perdurará en mi formación como estudiante y como persona.

A mi compañera de tesis Diana Laura Pérez Santamaría gracias por todos los momentos que pasamos juntas, por tu colaboración, dedicación y apoyo durante el proceso de nuestra tesis.

Aprecio profundamente nuestra colaboración y la forma en que hemos podido completar nuestras fortalezas y debilidades para lograr la culminación de este trabajo. Además, quiero agradecerte por tu apoyo emocional durante los momentos difíciles que ocurrieron durante el desarrollo de nuestra tesis. Espero seguir contando con tu compañía y amistad en esta nueva etapa que está por comenzar en nuestras vidas.

A todos los mencionados, no hay palabras suficientes para expresar mi gratitud hacia todos aquellos que han hecho posible este logro, su apoyo ha sido la piedra angular de mi éxito y estoy profundamente agradecida por ello.

Valeria Jenny Hernández de la Rosa.

Primeramente, agradezco a Dios, por sus bendiciones, por darme una gran familia y permitirme vivir este momento tan especial.

A la Universidad Autónoma de la Ciudad de México, por ser mi casa de estudio durante mi preparación profesional.

A mi familia, mis abuelos, tíos, primos, por su comprensión y estimulo constante, además de su apoyo incondicional a lo largo de mis estudios, en especial a mis padres y hermanos.

A ti Laura, por ser una gran mujer que sabe motivarme y orientarme, gracias mamá por ser mi ejemplo de fortaleza y dedicación. Por tu sacrificio y esfuerzo, por darme una carrera para nuestro futuro, por creer en mi capacidad, por tu comprensión, cariño y amor. Todo lo que yo pueda escribir es poco para expresarte mi agradecimiento y sobre todo el amor que siento por ti. Te admiro y te amo infinitamente mamá.

A ti Juan, por ser un papá trabajador, carismático, sencillo, por el apoyo, por la confianza, por ir y venir por mi durante muchas mañanas y tardes, por siempre estar conmigo a pesar de la distancia. Y como siempre han dicho que el primer amor de una hija es su padre, sin duda tú lo eres, eres el mejor ejemplo de amor. Gracias papá te amo.

A ti Yirari, por ser tan única, tan especial, con sentimientos tan nobles y yo tan afortunada de poder tenerte en mi vida, como mi hermana, y al mismo tiempo como mi mejor amiga, mi confidente, mi cómplice, mi todo. Gracias por apoyarme y estar para mí, así como yo estaré para ti siempre. Gracias por el mejor de los regalos, mi sobrina.

A ti Juan, mi Junior gracias por ser un gran compañero de vida y no solo personal también de mi vida académica, por tantas risas, peleas, tantas anécdotas, gracias por estar, por apoyarme, por motivarme, por mi sobrina. Conmigo cuentas siempre.

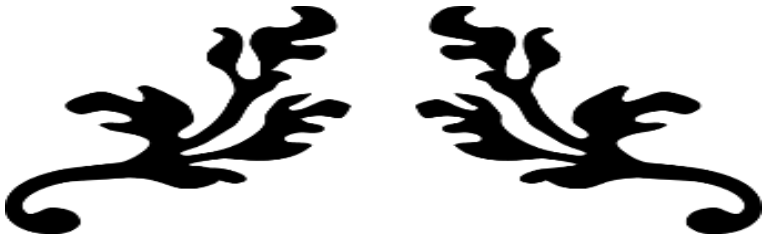
Con profunda estima y reconocimiento, mi más sincera gratitud a mi director, el M. en I. Luis Enrique Aranda Melo y codirector de tesis el Ing. Ariel Iván López Juárez. Su dedicación docente y su valiosa guía han sido pilares fundamentales en la dirección y enriquecimiento de la investigación. Gracias por la guía y todos los consejos que sin duda llevaré para mi futuro profesional.

Expreso mi agradecimiento a los lectores de tesis por su invaluable colaboración, M. en I. Patricia Hong, gracias por sus valiosos consejos, por su conocimiento a M. en I. Víctor Macías por su apoyo y consejos y Lic. Jorge Mendoza, gracias a los tres porque observaciones y constructivos comentarios han sido cruciales para la consolidación de este trabajo.

A mis maestros que formaron parte de mi vida universitaria, en particular al Ing. Oscar Ocampo y Lic. Alfonso León, gracias por trasmitirme sus conocimientos, por sus consejos y apoyo.

Mi más sincero agradecimiento a Valeria Jenny, mi compañera de tesis por su indudable apoyo y colaboración a lo largo de la tesis. Trabajar contigo ha sido una experiencia de aprendizaje enriquecedora, además de que te has convertido en una gran amiga, gracias Vale por todas las horas compartidas, por todas las experiencias vividas.

Diana Laura Pérez Santamaría.



Dedicatoria

Le dedico el resultado de esta tesis a mi familia. Principalmente a mis padres, Ernesto y Rosa, que me han apoyaron y me han alentado en los momentos buenos y malos a lo largo de mi carrera, gracias por enseñarme a superar mis miedos y a conseguir las metas que me propongo.

Me han enseñado a ser una persona, llena de principios y valores, a ser perseverante en cada una de mis metas y sueños, gracias por el gran apoyo, amor y compresión a lo largo de este largo camino, que hemos recorrido juntos.

A mis hermanas Norma y Monica que siempre me han apoyado incondicionalmente en mi formación académica. Gracias por sus consejos y por las noches de desvelo en las que me acompañaron para terminar mis deberes.

A los amigos que conocí a lo largo de la carrera Samara, Liz, David, Diana, Aparicio, que, aunque son pocos, son personas de excelente calidad.

De igual forma deseo dedicar este trabajo tesis a las personas que ya no se encuentran en este plano de existencia, a mis abuelos paternos Rafael y Cecilia, a mi abuela materna Consuelo, a mis tíos Rafael y Enrique.

Valeria Jenny Hernández de la Rosa.

"La mayor gloria no es caer nunca, sino levantarse siempre".

Nelson Mandela

A Dios por sus bendiciones sobre mí y llenarme de fuerza.

Dedico este trabajo a mis padres, Juan y Laura pues sin ustedes no lo habría logrado, me han enseñado a ser la persona que soy, mis principios, mis valores, mi perseverancia y mi empeño, por ayudarme a alcanzar el equilibrio que me permite dar todo mi potencial. Su bendición a diario me protege y me lleva por el camino del bien.

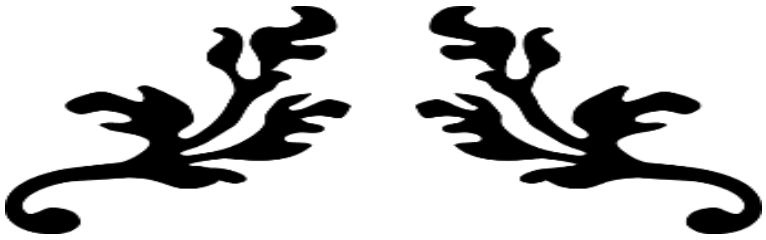
A mis hermanos, Juan y Yirari, por estar conmigo siempre, por ser los mejores hermanos, por darme el apoyo que necesito, por enseñarme el valor de muchas cosas, por sacarme muchas sonrisas, porque me hacen ser muy feliz.

A mis sobrinas, Samara y Alma Haydée, por ser los mejores regalos de vida.

A mis familiares, por estar y acompañarme en el proceso.

Con todo mi cariño y amor:

Diana Laura Pérez Santamaría.



Objetivos

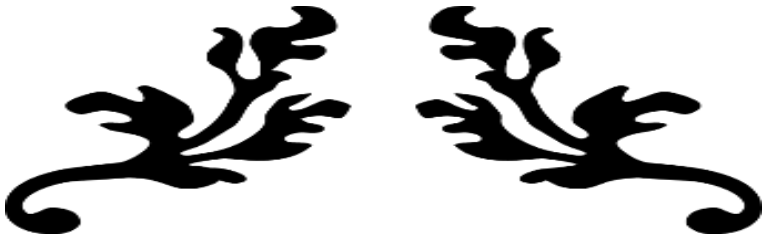
Objetivo

Objetivo general:

Analizar y comparar tres tecnologías: PIC, Arduino y Raspberry Pi, para poder identificar cuál se adapta mejor al IoT.

Objetivos específicos:

- Se realizarán tres prácticas enfocadas en IoT en cada una de las tecnologías.
- Mediante la realización de las prácticas se identificarán las principales diferencias entre cada una de las tecnologías.
- Comparar las tecnologías respecto a sus puertos.
- Comparar los puertos analógicos/digitales de las tecnologías.
- Seleccionar el protocolo de comunicación en el cuál trabajarán las tecnologías.
- Identificar qué tecnología nos proporciona mejor rendimiento referente a memoria.
- Identificar y argumentar cuál es la tecnología que más conviene aplicar a IoT.



Introducción

Introducción

El IoT (*Internet of Things*, Internet de las cosas), lo cambiará todo, incluso a nosotros mismos. Debemos tener en cuenta el impacto que Internet ha tenido en la educación, la comunicación, las empresas, la ciencia, el gobierno y la humanidad. Claramente Internet es una de las creaciones más importantes y poderosas de la humanidad, siendo Internet de las cosas su mayor auge, debido a su capacidad para reunir, analizar y distribuir datos que podemos convertir en información.

La información la obtenemos mediante sensores que interactúan con el mundo exterior, con respecto al procesamiento y transmisión de la información, se pueden emplear distintas tecnologías, es por eso, que estudiaremos tres tecnologías: PIC, Arduino y Raspberry Pi, cada una de ellas posee diferentes características particulares y otras que comparten. Debido a esto, identificaremos cuál de ellas brinda mejores características para llegar a la conclusión de cual tecnología es más eficiente para enfocarla en IoT.

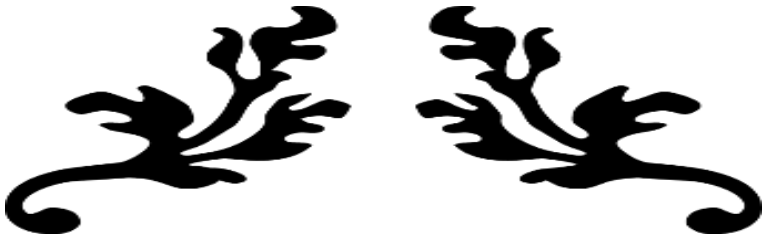
De las tecnologías, nos enfocaremos desde la capacidad de un puerto, hasta la forma en que se envía la información y bajo qué protocolo.

En el capítulo 1, veremos que son microcontroladores y microprocesadores, así como sus principales diferencias. Analizaremos las tecnologías PIC, Arduino y Raspberry Pi, en las que identificares sus principales particularidades, sus características, tales como puertos de E/S, protocolos de comunicación, etc. De tal forma que identificaremos cuál de las tecnologías tiene mayor eficiencia en el IoT.

En el capítulo 2, se aborda que es el IoT, la importancia de IoT, principales aplicaciones, arquitectura de IoT y protocolos de comunicación entre personas, procesos y cosas.

En el capítulo 3, se desarrollarán 3 prácticas propuestas, donde se compararán las 3 tecnologías. En la práctica 1 se compararán los puertos la forma en la que se activan. En la práctica 2 comparemos cómo funciona el módulo DAC en cada una de las tecnologías. Para la práctica 3 compararemos como es la conectividad en cada una de las tecnologías.

En el capítulo 4, se describirán los resultados obtenidos del capítulo de prácticas.



Justificación

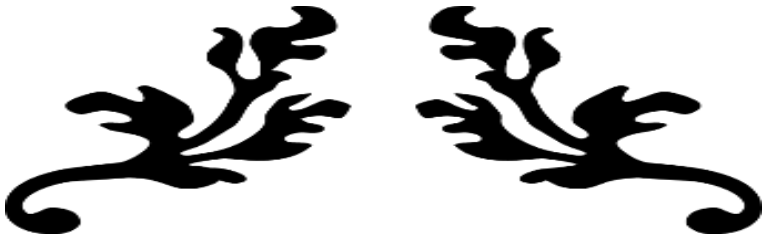
Justificación

El campo de la Ingeniería Electrónica es fundamental en los avances tecnológicos. Uno de sus objetivos es resolver tareas diversas, siendo los microcontroladores unos de los dispositivos con más potencial para la solución de dicha problemática, ya que, estos pequeños chips o dispositivos que pueden ser programados para realizar acciones o instrucciones que nosotros deseamos, son de bajo costo, prácticos y poderosos para circuitos que necesitan un ahorro de espacio.

La presente investigación se enfocará en el análisis de tres tecnologías de Microcontroladores y Microprocesadores: PIC, Arduino y Raspberry Pi. Existen infinidad de proyectos basados en tecnologías y cada día salen nuevas ideas que se pueden adaptar y mejorar, sin embargo, puede ser confuso elegir la mejor. Debido a esto, es que compararemos las tecnologías PIC, Arduino y Raspberry Pi, ya que, son tecnologías muy similares en funcionalidad y composición (Arduino y Raspberry Pi), tienen memoria. Las tres presentan entradas y salidas, con las tres se pueden desarrollar aplicaciones, pero la manera en la que cada una realiza estas funciones es donde está su mayor diferencia.

Así en una comparativa PIC, Arduino y Raspberry Pi podemos definir cuál es la mejor tecnología de acuerdo a las necesidades para el IoT.

Llegar a esta conclusión no es sencillo, se tienen que valorar muchos aspectos. Por eso es que, mediante el diseño de prácticas se distinguirán cuál de las tres tecnologías es más eficaz para el Internet de las Cosas. Posibilitando no solo la eficiencia dentro del internet de las cosas, sino también nos permitirá obtener mejores resultados en actividades más específicas.



Índice de Contenido

Índice de contenido

Capítulo 1

Capítulo 2

Capítulo 3

Capítulo 4

Índice de figuras y tablas

Índice de imágenes

Capítulo 1.

Capítulo 2.

Capítulo 3.

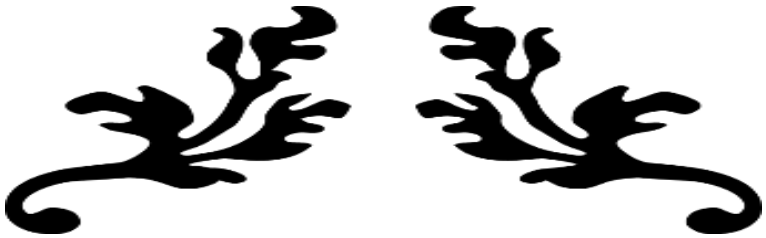
Índice de tablas

Capítulo 1.

Capítulo 2.

Capítulo 3.

Capítulo 4.



Capítulo 1

PIC, Arduino y Raspberry Pi

1.1 PIC´s

El nombre verdadero del microcontrolador es PICmicro (*Peripheral Interface Controller*, Controlador de interfaz periférica), conocido como PIC. Su ancestro fue creado en 1975 por la compañía General Instruments. Este PIC1650, fue diseñado para propósitos simples, como ser el control de sencillos periféricos de entrada y salida. Diez años más tarde al añadir una memoria EEPROM (*Electrically Erasable Programmable Read Only Memory*, Memoria de solo lectura programable y borrable eléctricamente) este circuito se convirtió en el microcontrolador PIC, que actualmente conocemos, debido a que permitía no solo controlar dispositivos externos, sino también la posibilidad de adaptarse y almacenar las instrucciones que el usuario establece según sus necesidades.

Estos microcontroladores PIC son circuitos electrónicos completos, todos sus componentes se organizan sobre un chip o pastilla semiconductora de silicio muy pequeña, encerrada en una cápsula plástica que contiene pines de acceso para comunicarse con el mundo exterior.

La industria informática acapara gran parte de los microcontroladores que se fabrican. Casi todos los periféricos de la computadora, desde el ratón o el teclado hasta la impresora, son regulados por el programa de un microcontrolador.

Los electrodomésticos de línea blanca (lavadoras, hornos, lavavajillas, etc.) y de línea marrón (televisores, videos, aparatos musicales, etc.) incorporan numerosos microcontroladores. Igualmente, los sistemas de supervisión, vigilancia y alarma en los edificios utilizan microcontroladores. También se emplean para optimizar el rendimiento de ascensores, calefacción, aire acondicionado, alarmas de incendio, robo, etc.

La instrumentación y la electromedicina son dos campos idóneos para la implementación de estos circuitos integrados. Una importante industria consumidora de microcontroladores es la automotriz que los aplica en el control de la climatización, la seguridad y los frenos ABS. [1]

1.2 Microprocesadores y Microcontroladores

Los microcontroladores son conceptualmente iguales a los microprocesadores, es decir, siguen el mismo esquema de funcionamiento. No obstante, la diferencia entre ambos reside en los elementos que se integran en el chip del microcontrolador, los cuales incluyen memoria ROM (*Read Only Memory*, Memoria de solo lectura), RAM (*Random Access Memory*, Memoria de acceso aleatorio), periféricos de entrada y salida, etc.

El área destinada a estos elementos disminuye la capacidad del circuito integrado y hace que los microcontroladores tengan un conjunto de instrucciones más reducido, menor capacidad de cálculo, etc. Por esta razón las áreas de aplicación de los microcontroladores y los microprocesadores son complementarias. [2]

1.2.1 Microprocesadores

Los microprocesadores constan de tres partes:

- La unidad central de procesamiento (CPU), la cual reconoce y ejecuta las instrucciones de un programa. La arquitectura general del CPU consta de la unidad lógica y aritmética (ALU: *Arithmetic and Logic Unit*), la unidad de control y los registros. [3]
- Las interfaces de entrada y salida, para manejar las comunicaciones entre la computadora y el mundo exterior.
- La memoria, donde se almacenan instrucciones de programas y datos.

Los microprocesadores que contienen memoria y varios arreglos de entradas/salidas (E/S) en un mismo chip se llaman microcontroladores.

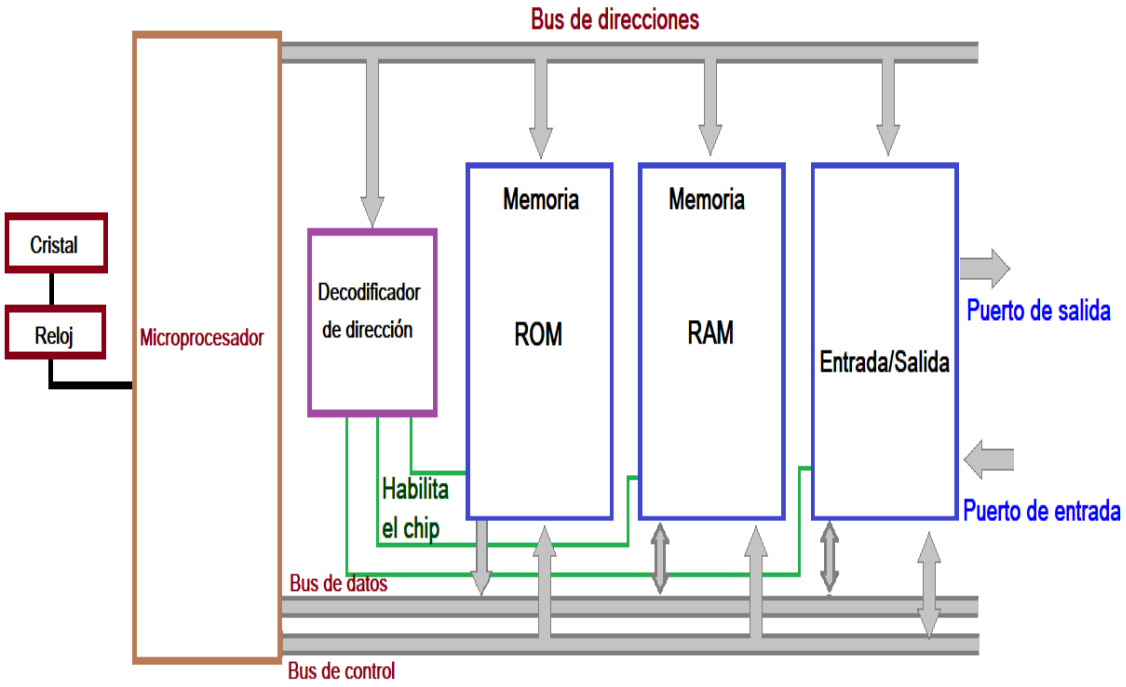


FIGURA 1. 1. ESQUEMA GENERAL DE UN MICROPROCESADOR Y SUS BUSES

Bolton, W. (2013). *Mecatrónica: Sistemas de control electrónico en la ingeniería mecánica y eléctrica*. Alpha Editorial.

1.2.1.1 Buses

Hay tres tipos de buses en un microprocesador:

1. *Bus de datos*
   Los datos asociados con las funciones de procesamiento de la CPU son transportados a través del bus de datos. De esta manera, se utiliza para transportar palabras hacia o desde la CPU y la memoria o las interfaces de E/S. En cada línea del bus viaja una señal binaria, es decir, un 0 o un 1. Entre más líneas tenga el bus de datos, más larga podrá ser la palabra que se utilice. El intervalo de valores que puede emplear un elemento de datos está restringido al espacio correspondiente a la longitud de la palabra. Los primeros

microprocesadores eran dispositivos de 4 bits, en la actualidad existen microprocesadores de 16, 32 y 64 bits, sin embargo, los microprocesadores de 8 bits aún se utilizan mucho en controladores.

2. *Bus de direcciones*
   El bus de direcciones transporta señales que indican dónde se pueden encontrar los datos y hace la selección de alguna localidad de memoria de los puertos de entrada y salida. Cada localidad en la memoria tiene una identificación única, denominada dirección, de modo que los sistemas son capaces de seleccionar una instrucción o datos específicos en la memoria. Cada interfaz entrada/salida tiene también una dirección. Cuando una dirección dada se selecciona, colocándola en el bus de direcciones, dicha localidad será la única que estará abierta a la comunicación que se envía desde el CPU, ya que, ésta solo puede comunicarse con una localidad a la vez. Entre más memoria direccionable haya, mayor es la cantidad de datos que es posible guardar, así como mayor y más complejo el programa que se puede utilizar.
3. *Bus de control*
   Las señales referentes a las acciones de control se transportan en el bus de control. El bus de control también se usa para transportar las señales de reloj del sistema que deben sincronizar todas las acciones del sistema microprocesador. El reloj es un oscilador controlado por un cristal y produce pulsos de periodos regulares. [4]

En general se hace referencia al microprocesador como la unidad de procesamiento central (CPU). Esta es la parte del procesador en la que se procesan los datos, se extraen instrucciones y datos. La estructura interna, conocida como arquitectura de un microprocesador se simplifica en la figura 1.2, las funciones que realizan los elementos que componen al microprocesador son las siguientes:

1. *Unidad lógica y aritmética (ALU)*
   Es la responsable de llevar a cabo la manipulación de los datos, en ella se realizan operaciones aritméticas y lógicas con números enteros y números reales tanto en punto fijo como en punto flotante. [3], [4]
2. *Registros*
   Los datos internos que la CPU utiliza, están contenidos en un grupo de registros mientras se ejecutan las instrucciones, éstas son localidades de memoria dentro del microprocesador que se usan para almacenar información involucrada en la ejecución de un programa. Un microprocesador contiene un grupo de registros, cada tipo de registro tiene una función diferente.
   Los registros se clasifican en: registros de propósito general, de instrucción, de acceso a memoria, de estado y de control. [3]
3. *Unidad de control*
   La unidad de control determina la temporización y secuencia de las operaciones. Ésta genera señales de temporización utilizadas para extraer de la memoria una instrucción del programa y ejecutarla. Las operaciones pertenecientes a los microprocesadores se reconocen por la cantidad de ciclos que se requieren para ejecutarlas. Existen diversos tipos de registros; la cantidad, la dimensión y el tipo de registros varía de un microprocesador a otro. Los registros más comunes son:

- Registro acumulador (A o Acc) almacena temporalmente los resultados aritméticos y lógicos intermedios que después serán tratados por la ALU.
- Registro de estado o registro de código de condición o registro de banderas. Es un registro de memoria donde se almacena el estado del procesador, y se indica mediante bits o banderas, que son modificados por el microprocesador como resultado de la ejecución de instrucciones aritméticas o lógicas, o por acciones como pedidos de interrupción.
- Contador del programa (PC) o apuntador de instrucciones (IP) contiene la dirección de la siguiente instrucción por ejecutar en el programa. Cada vez que una instrucción se ejecuta, el valor de PC se incrementa, de modo que siempre contiene la dirección de memoria en la que se almacena la siguiente instrucción por ejecutar. Esta ejecución se puede interrumpir mediante instrucciones como *jump* o ramificación (*branch*).
- Registro de direccionamiento de memoria se relaciona con las operaciones de lectura o escritura de la memoria. Los dos más importantes son: el registro de direcciones de memoria (*Memory Address*), en el que se almacena la dirección de la memoria a la que se desea acceder, y el registro de datos de memoria (*Memory Buffer Register*)
- Registro de instrucciones (RI) se relaciona con el acceso a las instrucciones.
- Registro de propósito general se emplea como operandos en las instrucciones en lenguaje ensamblador. Puede almacenar datos o direcciones de memoria.
- Registro de apuntador de la pila (SP) [3], [4]

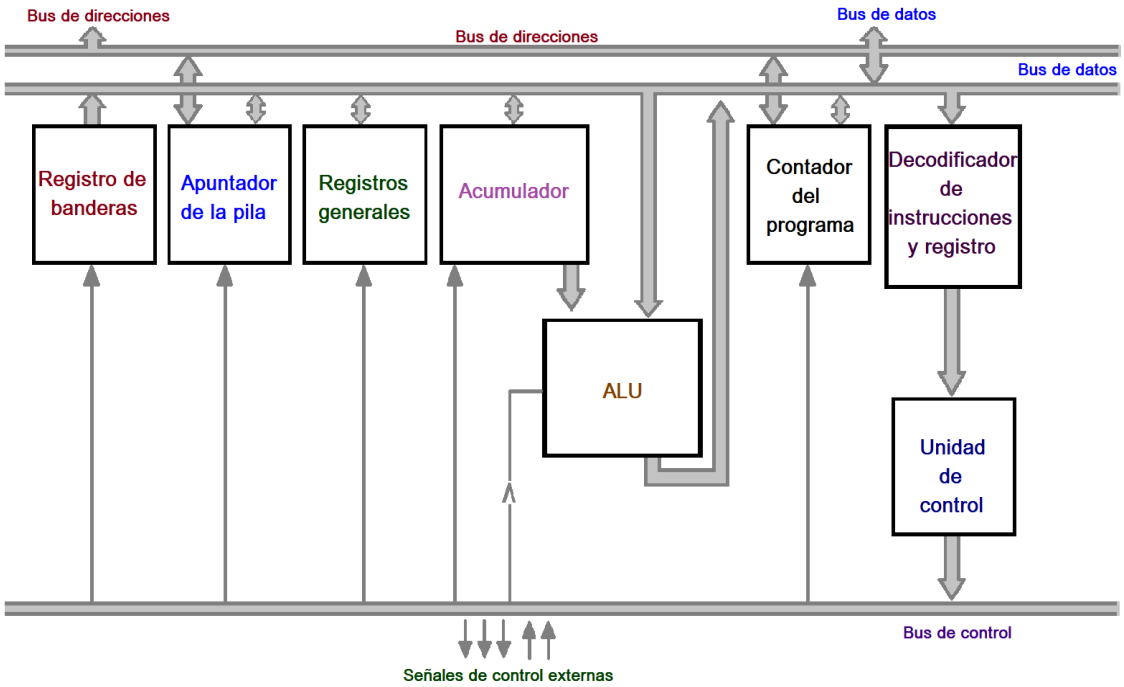


FIGURA 1. 2. ARQUITECTURA INTERNA DE UN MICROPROCESADOR

Bolton, W. (2013). *Mecatrónica: Sistemas de control electrónico en la ingeniería mecánica y eléctrica*. Alpha Editorial.

1.2.1.2 Memoria

La unidad de memoria de un microprocesador guarda datos binarios que pueden ser códigos de instrucciones de un programa, o números con los que se realizan operaciones. El tamaño de la memoria depende de la cantidad de líneas del bus de direcciones. Los elementos de la unidad de memoria están formados por grandes cantidades de celdas de memoria, cada una guarda un bit 0 ó 1.

Las celdas de memoria se agrupan por localidades, cada localidad puede guardar una palabra. Para acceder a la palabra almacenada, se identifica cada localidad por una dirección única.

La capacidad de la unidad de memoria se especifica por la cantidad de localidades de memoria disponibles. Existen varios tipos de memoria, entre los cuales se encuentran:

1. *ROM*
   La memoria ROM también conocida como memoria de solo lectura, guarda datos de forma permanente, ya que, es programada con el contenido que se requiere durante su fabricación. Mientras el chip de memoria esté en el circuito no es posible escribirle datos y solo se permite la lectura. Esta memoria se utiliza para programas fijos, como el sistema de arranque o "*boot*" y programas para aplicaciones. Aun cuando se suspenda la alimentación eléctrica la memoria no pierde su contenido.

2. *PROM*
   Una memoria PROM (*Programmable* ROM) es un tipo de ROM que puede programarse una única vez. Ésta construida con un fusible, que se quema de manera irreversible haciendo circular por ellos una corriente excesiva. Los chips nuevos tienen todos los fusibles intactos, por lo cual todo su contenido se encuentra en estado alto.

3. *EPROM*
   ROM borrable y programable (EPROM) es una memoria que es posible programar y modificar. Para almacenar se aplican voltajes a las terminales del circuito integrado el cual produce una configuración de celdas cargadas. Esta configuración queda permanente en el chip y se puede borrar mediante luz ultravioleta.

4. *EEPROM*
   PROM eléctricamente borrable es similar a la anterior solo que para borrar los datos se utiliza un voltaje alto.

5. *RAM*
   Esta memoria también conocida como memoria de acceso aleatorio, se encarga de almacenar datos temporales los cuales se utilizan para realizar operaciones. [4]

1.2.1.3 Interfaces de Entrada / Salida

La operación de entrada/salida se define como la transferencia de datos entre el microprocesador y el mundo exterior. Una de las funciones más importantes de estos circuitos es sincronizar la transferencia de datos entre el microprocesador y el dispositivo periférico. En las operaciones de entrada, el dispositivo de entrada coloca los datos en el registro de datos del circuito de interfaz; estos datos permanecen ahí hasta que los lee el microprocesador. En las operaciones de salida, el microprocesador coloca los datos en el registro hasta que los lee el dispositivo periférico. [4][5]

Otra forma es mediante interrupción, ya que, la interfaz le envía los datos al microprocesador cuando estos son validados, se suspende la ejecución del programa principal para ejecutar la rutina asociada con la interrupción y así leer los datos.

Las E/S están asociados con sistemas de interconexión tanto internos como externos. Los sistemas de interconexión internos utilizan interrupciones como técnicas de acceso a los periféricos, de modo que el microprocesador no tenga que esperar a que un dispositivo esté libre para realizar la transferencia. También utilizan Acceso Directo a Memoria (DMA / *Direct Memory Access*) el cual transfiere bloques de datos entre el periférico y la memoria sin la intervención del microprocesador. [3]

1.2.2 Microcontroladores

Componentes de un microcontrolador

Un microcontrolador combina los recursos fundamentales de un microprocesador, es decir, la unidad central de procesamiento (CPU), la memoria y los recursos de E/S, en un único circuito integrado.

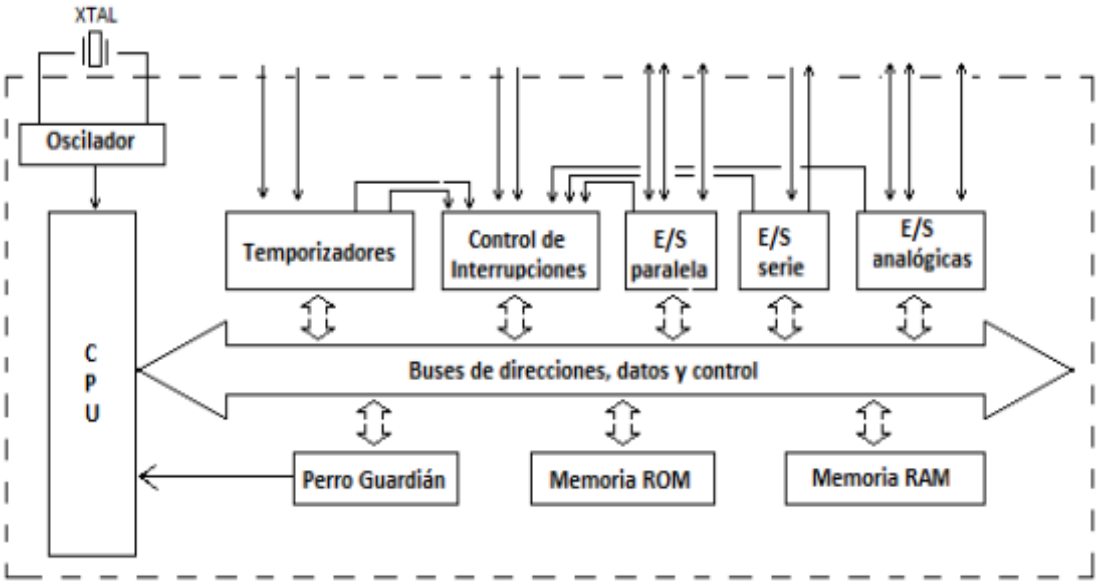


FIGURA 1. 3. DIAGRAMA DE BLOQUES GENERAL DE UN MICROCONTROLADOR.

VALDES, F., & ARENY, R. P. (2007). *MICROCONTROLADORES FUNDAMENTOS Y APLICACIONES CON PIC*. MARCOMBO.

Los microcontroladores disponen de un oscilador que genera los pulsos que sincronizan todas las operaciones internas. El oscilador puede ser del tipo RC o del tipo XTAL (cristal de cuarzo), debido a su gran estabilidad de frecuencia. La velocidad de ejecución de las instrucciones del programa está en relación directa con la frecuencia del oscilador del microcontrolador.

El CPU es el cerebro del microcontrolador. Esta unidad tiene las instrucciones del programa, una a una, desde la memoria donde están almacenadas, las interpreta y hace que se ejecuten. En la CPU se incluyen los circuitos de la ALU para realizar operaciones aritméticas y lógicas.

La CPU de un microcontrolador dispone de diferentes registros, algunos de propósito general y otros para propósitos específicos. Entre ellos están el Registro de Instrucción, el Acumulador, el Registro de Estado, el Contador de Programa, el Registro de Direcciones de Datos y el Puntero de la Pila.

La memoria del microcontrolador es el lugar donde son almacenadas las instrucciones del programa y los datos que manipula. En un microcontrolador hay dos tipos de memoria: la memoria RAM y la memoria ROM. Tanto la memoria RAM como la memoria ROM son de acceso aleatorio, esto quiere decir, que el tiempo necesario para localizar un dato no depende del lugar de la memoria donde esté almacenado. En las memorias de acceso secuencial cuanto más alejado esté un dato de la posición a la que se ha accedido por última vez, más se tarda en localizarlo.

La entrada y salida en los microcontroladores es de suma importancia, ya que, a través de ellas el microcontrolador interactúa con el exterior. Forman parte de la entrada y salida de los puertos en paralelo y serie, los temporizadores y la gestión de las interrupciones. El microcontrolador puede incluir entradas y salidas analógicas asociadas a convertidores A/D y D/A. De igual forma cuenta con el perro guardián (WDT, *Watchdog Timer*) que garantiza un funcionamiento seguro del microcontrolador.

Los puertos paralelos se organizan en grupos de hasta 8 líneas de entradas y salidas digitales. Los puertos serie pueden ser de varios tipos, según la norma de comunicación que implementen: RS-232C, I^2C, USB, Ethernet, etc.

Un requisito para que un microcontrolador se pueda utilizar en un gran número de aplicaciones es que tenga muchos recursos de entrada y salida.

1.3 Arquitectura de un microcontrolador

La arquitectura de un microcontrolador permite definir la distribución de su funcionamiento, en su fabricación se utiliza la arquitectura Von Neumann o Harvard, de igual forma que la arquitectura RISC o CISC, la selección de cada una de ellas depende del número de instrucciones que se necesite y la versatilidad de ejecución.

1.3.1 Arquitectura RISC y CISC

Las arquitecturas CISC (*Complex Instruction Set Computer* / Computadora con conjunto de instrucciones complejas) y RISC (*Reduced Instruction Set Computer* / computadora con conjunto de instrucciones reducidas) son los dos modelos generales de ordenadores.

La arquitectura CISC es una estructura interna de diseño de procesadores en la cual el bus de datos se comparte con el de direcciones, esta arquitectura se desarrolló por la necesidad de tener un amplio conjunto de instrucciones. Esta arquitectura tiene instrucciones más elaboradas las cuales realizan operaciones aritméticas complejas en comparación a la arquitectura RISC. [6], [7], [8]

Algunas de las características de la arquitectura CISC son:

- Tiene instrucciones de longitud variable que dependen del modo de direccionamiento.
- Dichas instrucciones requieren de varios ciclos de reloj para ejecutarse.
- Soportan cero, uno o más instrucciones.
- Direccionamiento de registro a registro, de registro a memoria y de memoria a registro.

La arquitectura RISC fue investigada por IBM en los años 70, en esta arquitectura tanto el bus de datos como el bus de direcciones se encuentran separados, lo cual permite que las instrucciones sean ejecutadas en un solo ciclo de máquina, esta arquitectura dispone de instrucciones reducidas, cada una de ellas puede realizar una operación muy simple a alta velocidad. [6], [8] y [9]

Algunas características de la arquitectura RISC son:

- La mayoría de sus instrucciones tienen la misma longitud.
- Tiene pocos modos de direccionamiento de datos y son aplicables a todas las celdas de la memoria de datos.
- La mayoría de las instrucciones se ejecutan en un solo ciclo, lo cual permite ejecutar varias instrucciones al mismo tiempo, implementando la segmentación (*pipelining*).

1.3.2 Arquitectura Von Neumann

Fue desarrollada en 1949 por el profesor John Von Neumann. Según esta arquitectura, existe un bus de datos que liga la CPU con la memoria de datos y de programa, por el cual viajan datos e instrucciones. Este concepto fue muy útil en las primeras décadas de las computadoras, pero, al incrementarse la cantidad de datos por procesar, la velocidad de procesamiento se redujo, rápidamente la arquitectura se saturó, ya que, el bus de datos debía compartirse con los datos y las instrucciones, lo cual generaba un cuello de botella. Algunos problemas que presentó eran el ancho de bus de datos que era de 8 bits y, como por él viajan los datos y las instrucciones, el ancho de los datos limitaba la longitud de las instrucciones. Como consecuencia las instrucciones con más de 8 bits debían ser enviadas en varias partes, lo cual resultaba en un sistema lento. [10]

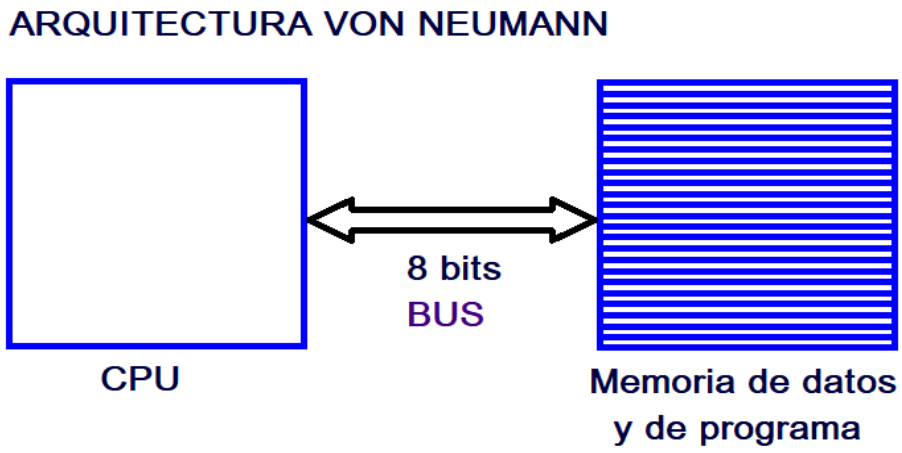


FIGURA 1. 4. DIAGRAMA DE LA CONEXIÓN ENTRE LA CPU, LA MEMORIA DE DATOS Y LA DEL PROGRAMA, UTILIZANDO EL BUS DE DATOS PARA LAS INSTRUCCIONES Y DATOS.

Benchimol, D. (2011). *Microcontroladores*. USERSHOP.

1.3.3 Arquitectura Harvard

Esta arquitectura conectó la CPU hacia su memoria mediante dos buses distintos: uno de datos y otro de instrucciones. De este modo el ancho de bus de instrucciones no ésta limitado por el de datos, y el procesador puede recibir instrucciones por buses diferentes, aprovechando el tiempo del ciclo de máquina. [6]

Ciclo de máquina

Los microprocesadores realizan una serie de operaciones básicas: búsqueda de la instrucción, decodificación, ejecución y almacenamiento de los resultados. Estas cuatro operaciones conforman el ciclo de máquina. Todas se encuentran sincronizadas con un reloj general (reloj del sistema). [5] [11]

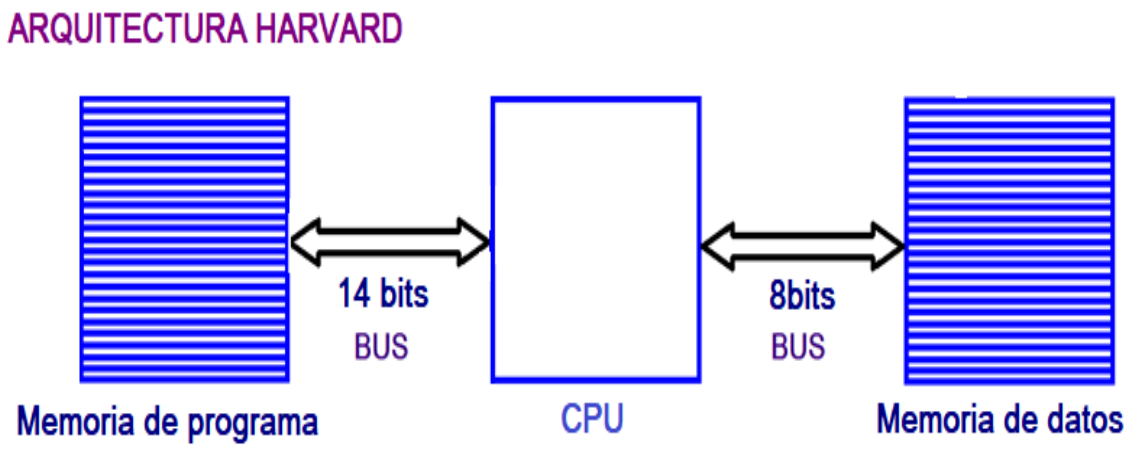


FIGURA 1. 5. BUSES SEPARADOS DE DATOS E INSTRUCCIONES DE LA ARQUITECTURA HARVARD.

Benchimol, D. (2011). *Microcontroladores*. USERSHOP.

1.4 Características generales de los microcontroladores PIC

Uno de los componentes fundamentales de la CPU de un microcontrolador es la unidad aritmética y lógica (ALU) la cual realiza las operaciones aritméticas y lógicas previstas en la memoria de las instrucciones del microcontrolador. ALU tiene asociada un registro que almacena temporalmente uno de los datos que intervienen en la operación de la ALU y eventualmente el resultado de la operación realizada. También se asocia a la ALU algunos bits que indican determinadas características del resultado de la operación. Estos bits indicadores usualmente forman parte del llamado registro de estado (STATUS). En muchos microprocesadores y microcontroladores, el registro asociado a la ALU recibe el nombre de Acumulador (ACC: *Accumulator*), como se muestra en la figura 1.6 el ACC está en la salida de la ALU, de modo que el resultado de cualquier operación aritmética o lógica siempre es depositado en el ACC.

En los microcontroladores PIC se denomina Registro de Trabajo (W: *Working Register*) y hace funciones semejantes al Acumulador de los microprocesadores y microcontroladores tradicionales, pero su posición respecto a la ALU es distinta a la que tiene el ACC, de tal forma que los registros ACC y W no se comportan exactamente igual, como se muestra en la figura 1.6 el resultado de una operación aritmética o lógica puede depositarse en W o puede llevarse directamente a cualquier registro de la memoria de datos, esto le proporciona mayor flexibilidad y potencia.

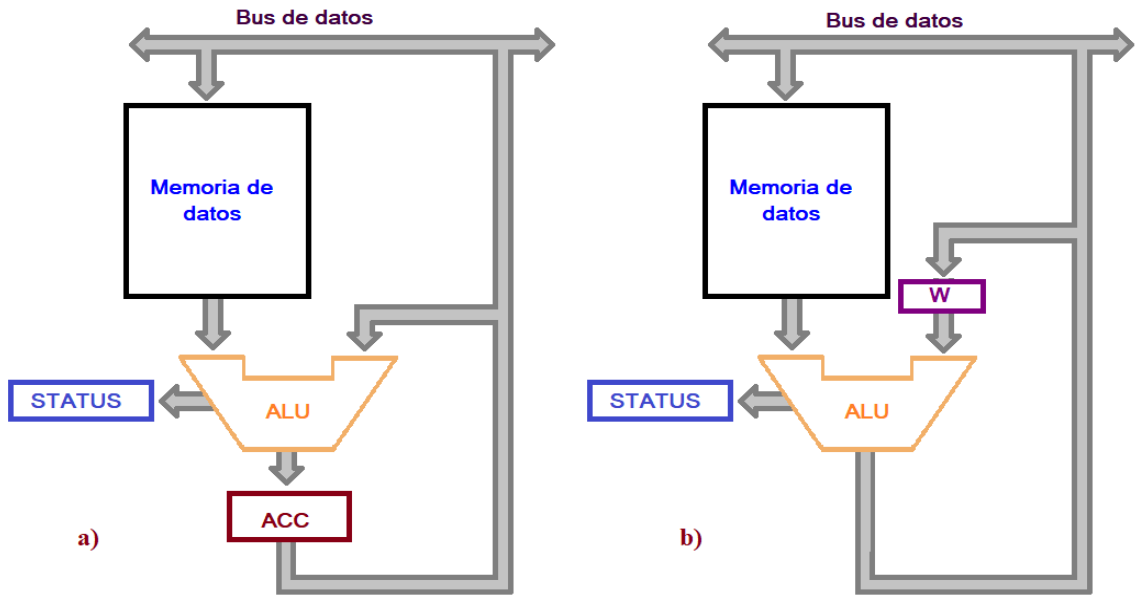


FIGURA 1. 6. A) MICROPROCESADOR CON UN REGISTRO DE NOMBRE ACC EN EL SE GUARDA EL RESULTADO DE CUALQUIER OPERACIÓN REALIZADA. B) MICROCONTROLADOR PIC QUE TRABAJA CON UN REGISTRO DE TRABAJO W EN EL CUAL SE PUEDE GUARDAR EL RESULTADO DE LA OPERACIÓN O NO.

VALDES, F., & ARENY, R. P. (2007). *MICROCONTROLADORES FUNDAMENTOS Y APLICACIONES CON PIC*. MARCOMBO

1.4.1 Ciclos máquina y ejecución de instrucciones

Los PIC tienen un oscilador principal que dicta la cadencia de las operaciones internas. Los pulsos generados por este oscilador son divididos internamente para generar cuatro señales denominadas Q1, Q2, Q3 y Q4, que sincronizan todo el trabajo interno del microcontrolador. Cada cuatro pulsos del oscilador principal se tiene un ciclo de máquina (CM).

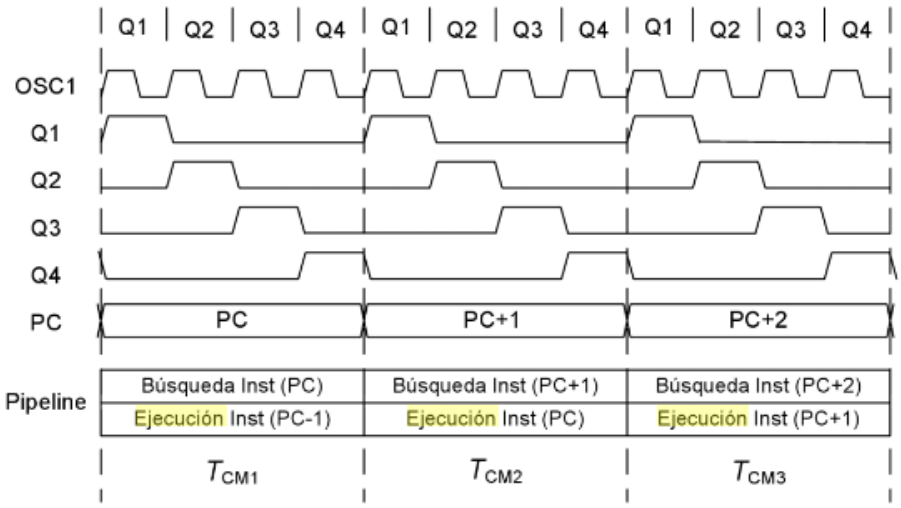


FIGURA 1. 7. SEÑALES DE RELOJ EN LOS MICROCONTROLADORES PIC OSC1 ES LA SEÑAL DEL OSCILADOR PRINCIPAL DE LA CUAL SE DERIVAN LAS SEÑALES INTERNAS Q1, Q2, Q3 Y Q4 QUE SE SINCRONIZAN LA BÚSQUEDA DECODIFICACIÓN Y EJECUCIÓN DE LAS INSTRUCCIONES TCM ES EL TIEMPO QUE DURA UN CICLO DE MÁQUINA Y EQUIVALE A CUATRO PULSOS DEL OSCILADOR PRINCIPAL.

VALDES, F., & ARENY, R. P. (2007). *MICROCONTROLADORES FUNDAMENTOS Y APLICACIONES CON PIC*. MARCOMBO.

Durante el tiempo Q1 de cada ciclo de máquina, el Contador de Programa (PC) se incrementa, apuntando hacia la instrucción que debe ser buscada, lo que ocurre durante el tiempo Q4. Paralelamente, durante todo el CM (desde Q1 a Q4) se está ejecutando la instrucción anterior.

La ejecución de una instrucción cualquiera se realiza en tres fases: búsqueda, decodificación y ejecución. En la fase de búsqueda (*fetch*), el microcontrolador lee la instrucción que está en la memoria de programa y la lleva a la CPU. En la fase de decodificación, la CPU determina cuál es la operación indicada en la instrucción. En la fase de ejecución, se ejecuta la operación prevista por la instrucción.

La ejecución de una instrucción se realiza en dos ciclos de máquina. En el primer ciclo de máquina se busca la instrucción en la memoria de programa y en el segundo ciclo de máquina se decodifica y ejecuta la instrucción.

Debido al mecanismo de segmentado (*pipeline*) ese segundo ciclo de máquina se traslapa, en el tiempo, con el primer ciclo de máquina de la siguiente instrucción, resultando que en promedio se ejecute una instrucción por ciclo de máquina.

1.4.2 Pila en los PIC

La pila en los PIC es una zona de memoria que se encuentra separada tanto de la memoria de programa como la de datos. Tiene una estructura LIFO (*Last In First Out*), por lo que el último valor que se guarda es el primero que sale. La única manera de cargar la pila es a través de la instrucción CALL (llamada) o por una interrupción que hace que, con cada una de ellas, se cargue el contenido de la PC en el valor superior de la pila. Esta acción es necesaria realizarla, porque no se dispone de ningún *flag* (identificador o bandera) que indique un desbordamiento o agotamiento de la pila.

Se utiliza la técnica de segmentación (*pipeline*) en la ejecución de las instrucciones.

La segmentación permite al procesador dividir la ejecución de las instrucciones en varias fases, para permitir atender instrucciones en cada fase, de manera simultánea. De esta forma, se puede ejecutar cada instrucción en un ciclo (en los PIC cada ciclo de instrucción son cuatro ciclos de reloj) Durante la fase de búsqueda, la dirección de la instrucción la proporciona la CPU, la cual normalmente se autoincrementa en la mayoría de las instrucciones, excepto en las de salto.

Las instrucciones de los microprocesadores más sencillos tienen una longitud de palabra de 12 bits; los medianos tienen una longitud de 14 bits, y los de mayor complejidad poseen más longitud.

La gran variedad de modelos de microprocesadores PIC permite que el usuario pueda seleccionar el más conveniente para su proyecto.

- El número de patillas de E/S varía de 4 a 70, esto dependerá de cada modelo.
- Casi todos disponen de una memoria EEPROM de 16 a 1024 bytes para almacenar datos y recuperarlos después de haber eliminado la alimentación.
- Las frecuencias más habituales de funcionamiento máximas, según el modelo son 4 MHz y 10 MHz llegando algunos a 48 MHz.
- Además de las entradas/salidas digitales, temporizadores y contadores, según el modelo, podemos disponer de entradas/salidas analógicas (convertidores A/D, D/A) comparadores analógicos, amplificadores operacionales, puerto serie, I2C, USB.
- Según la versión, la pila o *stack* dispone de un cierto número de niveles lo que supone poder encadenar más o menos subrutinas.
- Los microprocesadores más sencillos no admiten interrupciones.

1.4.3 Segmentación

El segmentado o *pipeline* es una técnica mediante la cual se consiguen que dos o más instrucciones se traslapen durante su ejecución. En los microcontroladores PIC, las instrucciones se ejecutan a través de un *pipeline* de dos etapas, la primera etapa corresponde a la búsqueda, para lo cual se requiere el tiempo correspondiente a un ciclo máquina. En la segunda etapa se decodifica y ejecuta la instrucción, lo cual toma otro ciclo de máquina, mientras se busca la instrucción siguiente. Es decir, que en cada ciclo de máquina se busca una instrucción y a la vez se ejecuta la instrucción anterior. Entonces, con cada nuevo ciclo de máquina se ejecuta una

nueva instrucción, excepto cuando haya instrucciones de transferencia de control, que requiere de dos ciclos máquina.

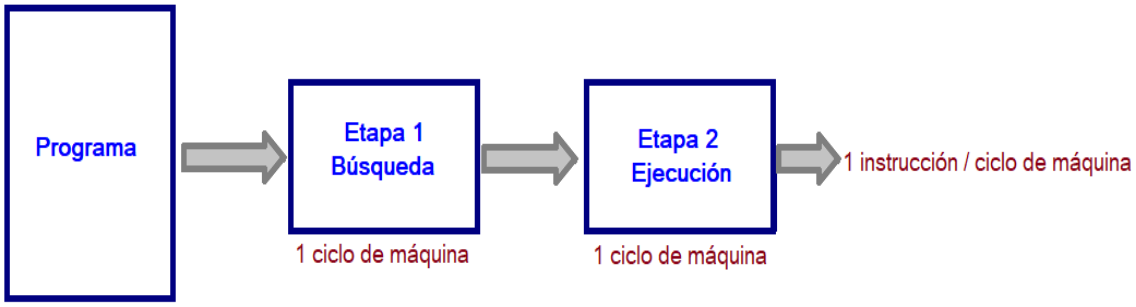


FIGURA 1. 8. PIPELINE DE DOS ETAPAS.

1.4.4 Osciladores

Los microcontroladores PIC disponen de las siguientes opciones para el oscilador principal:

- Oscilador XT (cristal de cuarzo)
- Oscilador RC (oscilador con resistencia y capacitor)
- Oscilador HS (cristal de alta velocidad)
- Oscilador LP (cristal para baja frecuencia y bajo consumo de corriente)
- Oscilador externo (la señal de reloj se aplica de manera externa al circuito)

Algunos dispositivos disponen de un oscilador RC interno de unos 4 [MHz]. Al aumentar la frecuencia del oscilador principal, se acorta la duración de los ciclos de máquina y con ello el tiempo de ejecución de las instrucciones, pero aumenta el consumo de energía.

El tipo de oscilador a utilizar se selecciona mediante los bits de configuración del PIC. En estos bits se especifican varias configuraciones de osciladores de cristal de cuarzo o resonador cerámico.

El oscilador de cristal LP se selecciona en aplicaciones de muy baja potencia, en el rango de frecuencias de 32 [kHz] a 200 [kHz]. Así mismo el oscilador de cristal XT se ajusta a las aplicaciones a frecuencias medias de 100 [kHz] a 4 [MHz]. Por otro lado, el oscilador de cristal HS es apropiado en aplicaciones de alta frecuencia, entre 8 [MHz] y 20 [MHz]. Así pues, el oscilador RC es una opción de bajo costo para el oscilador principal del microcontrolador, apropiado cuando la presión y la estabilidad en el valor de frecuencia no es esencial. [1], [11]

1.4.5 Modo de bajo consumo

En el modo o estado de bajo consumo o de reposo *(sleep*), el microcontrolador suspende casi todas sus funciones, incluso el oscilador principal deja de funcionar. En esas condiciones el microcontrolador consume muy poca corriente de la fuente de alimentación. Esto es principalmente útil cuando se alimenta con baterías o en caso de que se requiera reducir al mínimo el consumo de potencia. Cuando se coloca al PIC en modo *sleep*, será necesario despertarlo para que pueda continuar con la ejecución del programa. Existen varias formas de hacer esto:

- Al dar un reset al sistema.
- Por el desbordamiento del WDT
- Por una interrupción

Si se despierta por reset, el microcontrolador va directo a ejecutar la instrucción que está en la dirección 0 de la memoria del programa.

Cuando el WDT se desborde, provocará que el PIC salga del modo de bajo consumo y continué la ejecución del programa en la siguiente instrucción después del *sleep*.

Si es despertado por interrupción y el sistema de interrupción está habilitado se ejecuta la instrucción que sigue a la instrucción de *sleep* y se salta a la dirección 4 de la memoria de programa en busca de la rutina de atención a la interrupción. Si ocurre una interrupción mientras el microcontrolador está en modo de bajo consumo y las interrupciones no están habilitadas, el microcontrolador despierta, ejecuta la instrucción que sigue a la instrucción *sleep* y continúa la secuencia de instrucciones del programa, pero no salta a la dirección 4 de la memoria de programa. [12], [13]

1.4.6 Perro guardián (WDT)

El WDT se realiza mediante un oscilador independiente del oscilador principal del microcontrolador, de modo que funciona incluso durante el modo de bajo consumo, y un contador de pulsos que produce ese oscilador independiente. Si el contador se desborda mientras el microcontrolador está operando normalmente, es decir, no en modo de bajo consumo, se genera un reset al microcontrolador. Si el desbordamiento ocurre mientras el microcontrolador está en modo de bajo consumo, el microcontrolador despierta y ejecuta la instrucción que está a continuación de la instrucción *sleep*.

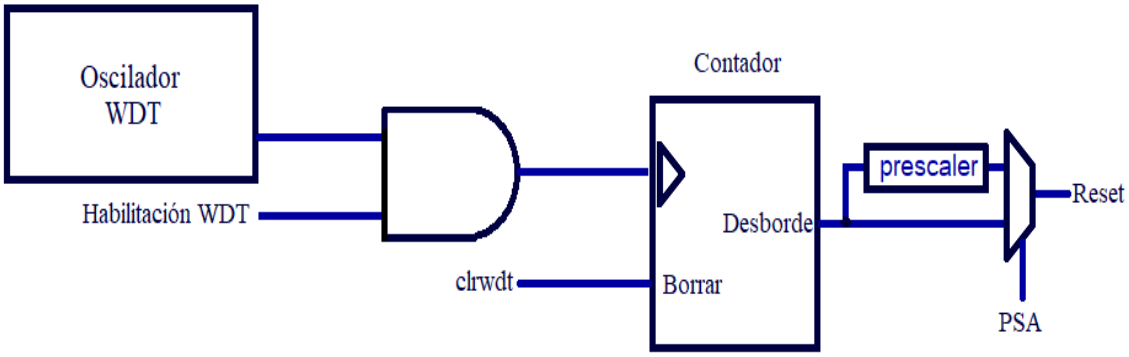


FIGURA 1. 9. DIAGRAMA DE BLOQUES DE LA HABILITACIÓN DEL WDT.

Rossano, V. (2009). *Electrónica & microcontroladores PIC*. USERSHOP.

El desbordamiento del perro guardián ocurre cada 18 [ms], aproximadamente. Para evitarlo hay que poner a 0 el contador del perro guardián antes de que transcurra ese tiempo y con ello provoque el reset del sistema, se debe utilizar la instrucción clrwdt, la cual pondrá a cero el contador del WDT y éste tendrá que contar de nuevo desde el principio.

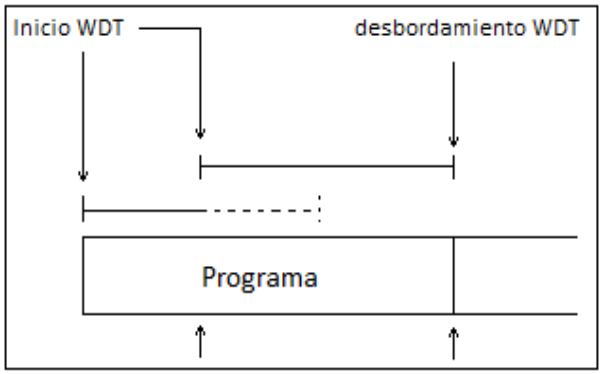


FIGURA 1. 10. INSTRUCCIÓN CLRWDT DEL WDT.

Rossano, V. (2009). *Electrónica & microcontroladores PIC*. USERSHOP.

Estos 18 [ms] se pueden ampliar hasta 2.3 [s] mediante la asignación de un contador adicional al perro guardián.

Se debe colocar la instrucción clrwdt en lugares estratégicos de nuestro programa para evitar que el WDT se desborde mientras el funcionamiento sea normal, y así el WDT no tendrá efecto alguno. Sólo hasta que se produzca una situación no deseada, el WDT se desbordará y provocará el reset obligando al sistema a iniciar de nuevo todo el funcionamiento desde el principio. Solo se acostumbra utilizar el WDT en programas avanzados, donde se requiere mayor seguridad en el funcionamiento de los circuitos. [1], [11] y [14]

1.5 Entradas y Salidas

Un periférico es un dispositivo externo conectado al microcontrolador, todos los dispositivos periféricos deben incluir una interfaz necesaria para conectarlo a los puertos del microcontrolador.

Un puerto (*port*) es un circuito que forma parte del microcontrolador y sirve de interfaz con algún dispositivo externo (un periférico). En general esta conexión dispone de *n* líneas para transportar el dato y de *m* líneas adicionales para controlar la transferencia de los datos entre el periférico y el puerto. Las líneas de control pueden no ser necesarias, como sucede en la E/S simple sin sincronización. Los puertos se identifican por sus direcciones, ubicadas por lo general en la memoria de datos. Para hacer referencia a los datos que entran o salen por un puerto, se necesita al menos una dirección.

El manejo de las señales de control puede requerir algunos bits adicionales, repartidos en una o dos direcciones más. En los microcontroladores PIC el acceso a los puertos se realiza a través de los registros de funciones especiales de la memoria de datos.

1.5.1 Entrada y Salida en Serie

1.5.1.1 Conexión entre equipos: interfaz RS-232C

La Alianza de Industrias Electrónicas (EIA, *Electronics Industries Association*) definió esta interfaz en 1962. Esta interfaz de comunicación a distancia requiere de equipos que pueden agruparse en:

Equipos terminales de Datos (DTE: *Data Terminal Equipment*) los cuales se encargan de enviar y recibir datos a través de la interfaz y los Equipos de Comunicación de Datos (DCE: *Data Communication Equipment*) los cuales son dispositivos que facilitan la comunicación.

La interfaz RS-232C (*Recommended Standard* número 232 revisión C) se utiliza para la conexión entre equipos de datos a corta distancia, ya que, está limitada a 15 [m], la velocidad máxima de datos es de 20 [kbit/s].

Las señales de la interfaz RS-232C tiene los siguientes niveles lógicos:

Nivel lógico 0: entre +3 [V] y +15 [V] con carga, hasta +25 [V] sin carga

Nivel lógico 1: entre -3 [V] y -15 [V] con carga, hasta -25 [V] sin carga

1.5.1.2 Bús I^2C

El Bus I^2C (*Inter-Integrated Circuit* / Intercomunicación con Circuitos Integrados) fue diseñado por Philips para la comunicación entre circuitos integrados o módulos, además de que utiliza muy pocas líneas para la conexión. Este Bus se utiliza para interconexión y transferencia síncrona de datos en serie entre diferentes dispositivos tales como: microcontroladores, memorias, convertidores A/D y D/A, etc. Los dispositivos utilizan solo dos líneas, una de ellas para la transferencia de datos (SDA: *Serial Data Line*) y la otra para la señal de reloj (SCL: *Serial Clock Line*).

El Bus puede alcanzar velocidades de transferencia de datos de hasta 100 [kbit/s] en el modo de baja velocidad (*low-speed mode*), 400 [kbit/s] en el modo rápido (*fast-mode*) y 3.4 [Mbit/s] en el modo de alta velocidad (*high-speed mode*). En la comunicación uno de los dispositivos se comportará como maestro (*master*) y los restantes como esclavos (*slave*). El maestro y los esclavos pueden funcionar como transmisor o receptor.

El dispositivo que actúa como maestro es el que inicia la comunicación, genera la señal de reloj y cierra la comunicación. El bus I^2C es multimaestro (multi-master) esto quiere decir que pueden existir varios servidores conectados al bus, pero en determinado momento solo uno actuará como tal. Cada dispositivo cuenta con una dirección única, con la cual se identifica durante la comunicación.

1.5.1.3 Puerto serie USART

El microcontrolador PIC posee un puerto llamado USART (*Universal Synchronous Asynchronous Receiver Transmitter*) o SCI (*Serial communication Interface*) el cual puede ser configurado como una comunicación asíncrona (*full-duplex*) o una comunicación síncrona (*half-duplex*). El puerto USART utiliza dos terminales TX/CK y RX/DT. En el modo asíncrono TX/CK es la terminal de transmisor y RX/DT es la terminal de receptor.

Por otra parte, en el modo síncrono si el dispositivo está configurado como servidor TX/CK actúa como terminal de salida de reloj por el contrario si el dispositivo está configurado como cliente TX/CK actúa como terminal de entrada de reloj. La terminal RX/DT es bidireccional por ello transmite o recibe los datos.

El puerto USART utiliza los registros de funciones especiales TXREG y RXREG para almacenar los datos que se van a transmitir o los que se van a recibir, de igual forma utiliza los registros TXSTA y RXSTA para controlar el puerto y el registro SPBRG para establecer la velocidad de transmisión.

1.6 Familias de microcontroladores PIC.

El microcontrolador PIC es una familia de microcontroladores fabricados por Microchip Technology Inc. Actualmente, el PIC es uno de los microcontroladores más populares utilizados en educación y en aplicaciones comerciales e industriales. La familia consta de más de 140 dispositivos, que van desde simples dispositivos duales en línea de 4 pines con 0.5 [K] de memoria hasta dispositivos complejos de 80 pines con 32 [K] de memoria.

Aunque la familia se compone de una gran cantidad de dispositivos, todos los dispositivos tienen la misma estructura básica, ofreciendo las siguientes características fundamentales:

- Conjunto de instrucciones reducido (RISC) con solo 35 instrucciones.
- Puertos de E/S digitales bidireccionales.
- Memoria de datos RAM.
- Flash reescribible o memoria de programa programable de una sola vez.
- Temporizador en chip con escalador previo.

- Temporizador de vigilancia; reinicio de encendido.
- Operación de cristal externo; capacidad de fuente/sumidero de corriente de 25 [mA].
- Modo de suspensión de ahorro de energía.

Los dispositivos más complejos ofrecen las siguientes funciones adicionales:

- Canales de entrada analógica.
- Comparadores analógicos.
- USART en serie.
- Memoria EEPROM no volátil.
- Temporizadores adicionales en el chip.
- Externo e interno (temporizador).
- Interrupción.
- Salida PWM (*Pulse Width Modulated*, Modulación por ancho de pulso).
- Interfaz de bus CAN (*Controller Area Network*, Red de área del controlador).
- Interfaz de bus I^2C.
- Interfaz USB.
- Interfaz LCD.

La familia de microcontroladores PIC se puede clasificar, atendiendo al tamaño de sus instrucciones, en tres grandes grupos o gamas. [11], [15]

Gama	
Baja	12 bits
Media	14 bits
Alta	16 bits

TABLA 1. 1. CLASIFICACIÓN DE INSTRUCCIONES DE ACUERDO A LA GAMA DE PIC.

1.6.1 Microcontroladores de gama baja

Los Microcontroladores PIC de gama baja disponen de un repertorio de 33 instrucciones de 12 bits cada una. La memoria de programa tiene una capacidad de hasta 2048 palabras de 12 bits y está organizada en páginas de 512 palabras cada una. La memoria de datos está formada por registros de 8 bits y se organiza en bancos de hasta 32 registros cada uno.

Los PIC de gama baja tienen una pila (*stack*) de dos niveles, para guardar direcciones de la memoria de programa. No tienen interrupciones. Su entrada y salida tiene un pequeño número de dispositivos, que comprenden hasta tres puertos de entrada y salida de hasta 8 bits cada uno, un temporizar y un comparador (según el modelo del PIC). [11]

PIC de Gama Baja					
PIC	Encapsulado	Memoria	Consumo de Corriente	Modo de bajo consumo	Recursos de E/S
PIC16X5x	18, 20 o 28 terminales	EEPROM, OTP o FLASH	2 [mA] a 5 [V]	3 [µA] a 3 [V]	n/a
PIC12X5x	8 terminales	OTP o FLASH	2 [mA] a 5 [V]	2 [µA] a 3 [V]	Puerto paralelo de 6 bits. Temporizador. Convertidor A/D.
PIC10	6 o 8 terminales	FLASH	350 [µA] a 2 [V]	100 [nA] a 2 [V]	Puerto paralelo de 6 bits. Temporizador Comparador

TABLA 1. 2. MICROCONTROLADORES PIC DE GAMA BAJA.

1.6.2 Microcontroladores de gama media

Estos dispositivos tienen un repertorio de 35 instrucciones de 14 bits cada una. La memoria de programa puede llegar a los 8k (8192) palabras de 14 bits y se organiza en páginas de 2k (2048) palabras cada una. La memoria de datos está formada por registros de 8 bits y está organizada en bancos de 120 registros cada uno, con un máximo de cuatro bancos. En general, los PIC de gama media poseen algo de memoria de EEPROM, todos tienen una pila de 8 niveles, donde almacenan las direcciones de la memoria de programa.

Los PIC de gama media tienen una amplia variedad de dispositivos de entrada y salida. Cuentan con varios puertos paralelos (puertos A, B, C, etc.) Disponen también de hasta tres temporizadores, dos módulos de captura, comparación y modulación de ancho de pulsos (PWM), varios tipos de puertos serie para la comunicación síncrona y asíncrona, un convertidor A/D de 10 bits asociado con multiplexor con varias entradas analógicas. [11], [15]

PIC Gama Media				
PIC	Voltaje	Consumo de Corriente	Modo de bajo consumo	Encapsulado
PIC16	2 [V]	100 [µA]	1 [nA] a 2 [V]	8 terminales
PIC12X6xx	2 [V]	100 [µA]	1 [nA] a 2 [V]	8 terminales

TABLA 1. 3. MICROCONTROLADORES PIC DE GAMA MEDIA.

1.6.3 Microcontroladores de gama alta

Los microcontroladores de gama alta se distinguen por instrucciones de 16 bits, mayor profundidad en la pila y un sistema de interrupciones más elaborado que incluye, además de interrupciones internas de los dispositivos integrados en el microcontrolador, varias entradas de interrupciones externas. Algunos PIC de gama alta tienen arquitectura abierta, que admite la aplicación de las memorias de programa y de datos. El número de dispositivos de entrada y salida es mucho más amplio que en los PIC de gama media. [4]

PIC de Gama Alta.						
PIC	Repertorio	Memoria de programa	Memoria de Datos	Consumo de corriente	Memoria	Pila
PIC 17	58 instrucciones de 16 bits	64k (65536) palabras de 16 bits	1k (1024) registros de 8 bits	100 [µA]	EPROM ROM OTP	16 niveles
PIC 18	77 instrucciones de 16 bits	2 MB palabras de 16 bits	4k (4096) registros de 8 bits	2 [mA]	FLASH	31 niveles

TABLA 1. 4. MICROCONTROLADORES PIC DE GAMA ALTA.

1.7 Memoria en los Microcontroladores

La memoria de un microcontrolador es el lugar donde se almacenan el programa que se ejecuta y los datos o variables utilizados por el programa. La memoria es un conjunto de celdas o localizaciones que se identifican por su dirección. En cada celda se almacena una palabra. La capacidad de celdas de la memoria define su tamaño y se mide en palabras, ya sean bits o bytes.

1.7.1 Organización lógica de la memoria

La memoria de los microcontroladores se organiza normalmente como un todo (organización lineal) o por bloques llamados páginas. En la organización lineal, las direcciones de las celdas son números binarios consecutivos. En este caso, cada celda se identifica por su dirección lineal, formada por un número binario único.

Una página es una porción de memoria de tamaño fijo. Las páginas están una a continuación de la otra sin traslaparse. Cada página se puede identificar por un número consecutivo denominado número de página. Dentro de una página, las celdas se identifican por su posición respecto al comienzo de la página, llamada desplazamiento.

En una memoria organizada en páginas, la dirección de una celda se compone de dos elementos: el número de la página y el desplazamiento. El conjunto de estos dos elementos constituye la

dirección lógica de la celda en el esquema paginado de la memoria. La dirección lógica de una celda se representa en la figura 1.11.

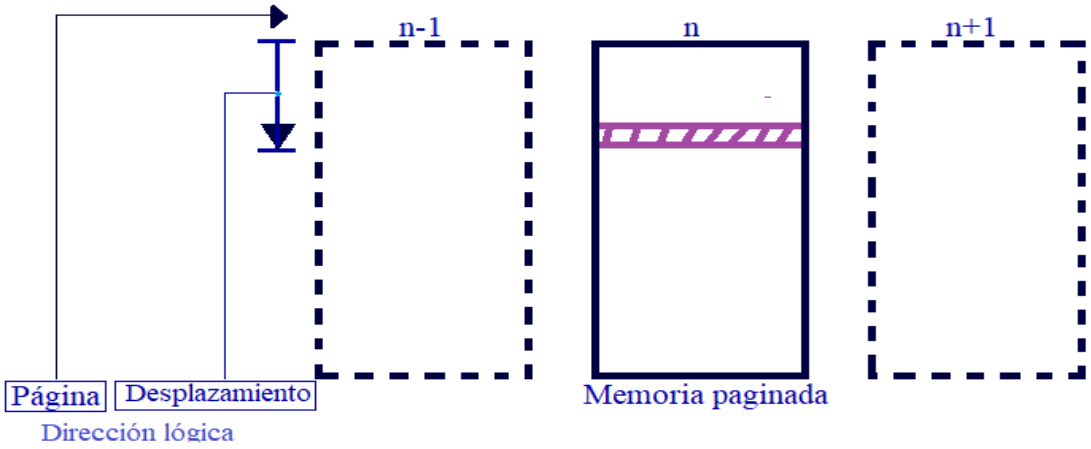


FIGURA 1. 11. ORGANIZACIÓN DE LA MEMORIA EN PÁGINA EL NÚMERO DE LA PÁGINA Y EL DESPLAZAMIENTO.

La dirección lineal se puede obtener de la dirección lógica, se obtiene multiplicando el número de la página por su tamaño y sumando a este resultado el desplazamiento que tiene la celda dentro de la página.

1.8 Temporizadores

El microcontrolador PIC de clase media tiene hasta tres módulos básicos para temporizar, que se identifican con los nombres Timer0, Timer1, y Timer2. Todos los PIC disponen al menos del Timer0. Algunos PIC tienen uno o dos módulos adicionales para temporizar, que amplían las posibilidades de los módulos básicos, llamados módulos de Comparación Captura y PWM, o módulos CCP (Capture/Compare/PWM), que comparten componentes y funciones con el Timer1 y el Timer2.

Cada uno de los temporizadores disponibles en un PIC de clase media tienen, como elemento, un contador síncrono ascendente de 8 ó 16 bits. Estos contadores se pueden programar para contar pulsos internos o externos. El número almacenado en cada contador se puede leer o modificar mediante la lectura o escritura de registros de funciones especiales asociados al temporizador. El desbordamiento de los contadores queda reportado en bits indicadores disponibles en esos registros, y puede generar también una solicitud de interrupción al microcontrolador.

Los temporizadores pueden disponer de un contador asíncrono auxiliar. Este contador auxiliar se inserta en el camino de los pulsos, antes del contador principal, en cuyo caso funciona como un pre- divisor (*prescaler*), o después del contador principal funcionando entonces como post-

divisor (*postscaler*). Los temporizadores Timer0 y Timer1 tienen solamente un pre-divisor; el Timer2 dispone de un pre-divisor y un post-divisor. [11], [12] y [13]

1.9 Interrupciones

Las interrupciones permiten a cualquier suceso interior o exterior interrumpir la ejecución del programa principal en cualquier momento. En el momento de producirse la interrupción, el PIC ejecuta un salto a la rutina de atención a la interrupción, donde se atenderá la demanda de la interrupción. Cuando se termina de ejecutar dicha rutina, el PIC retorna a la ejecución del programa principal en la misma posición de la memoria de programa donde se produjo la interrupción.

En un microcontrolador hay varias fuentes de interrupción, unas internas y otras externas. Las interrupciones internas tienen su origen en los módulos de entrada y salida del microcontrolador, la memoria o el CPU. Las interrupciones externas se originan en un periférico y llegan al microcontrolador por alguna de sus terminales.

Aunque los microcontroladores disponen de un bit para el control global del sistema de interrupción. Con este bit se permite o impide el paso de cualquier interrupción hacia la CPU. Los bits de control utilizados para permitir o no el paso de las solicitudes de interrupción hacia la CPU se denominan máscaras; de ahí que las interrupciones que se puedan habilitar o inhabilitar por programa se llamen interrupciones enmascarables (*maskable interrupts*) y las interrupciones que no se pueden inhabilitar por programa se denominen interrupciones no enmascarables (*non-maskable interrupts*). En general cuando una solicitud de interrupción que llega a la CPU es atendida, el sistema de interrupción queda inhabilitado (el bit de control global es puesto a 0), por tal motivo no llegará a la CPU ninguna nueva solicitud de interrupción. Para poder atender otras solicitudes de interrupción se debe habilitar nuevamente el sistema.

Al producirse una interrupción, el PIC salta automáticamente a la dirección del vector de interrupción de la memoria de programa y ejecuta la porción de programa, correspondiente a la atención de la interrupción, hasta encontrar la instrucción RETFIE (*Return From Interrupt*). Las fuentes de interrupción dependen del PIC utilizado.

Los PIC de gama baja y media tiene un único vector de interrupción situado en la dirección 04h de programa, mientras que los de gama alta tienen dos vectores de interrupción de distinta prioridad, alta y baja, situados en la posición 08h y 18h de la memoria de programa.

Al poseer un único vector de interrupción (dos en la gama alta), el PIC posee unos registros de control donde mediante la utilización de banderas, o *flags*, el usuario puede determinar qué es lo que ha producido la interrupción; además en estos registros, se pueden habilitar o no las distintas fuentes de interrupción e incluso permite una habilitación general. Cuando ésta se activa, los *flags* se activan. [3], [11], [12] y [13]

1.9.1 Tipos de interrupciones

Las interrupciones pueden dividirse en internas o externas, y en enmascarables o no enmascarables. Las internas son disparadas por el hardware interno del microcontrolador, por ejemplo, el convertidor analógico/digital, los temporizadores, etc. Las externas son disparadas mediante la aplicación de un pulso o un estado sobre un pin del microcontrolador, denominado INT (interrupción).

En las del tipo enmascarable o no enmascarable, la primera necesita tener activo un bit de habilitación para generarse, aunque hayan sido solicitadas; en tanto las segundas ocurren no importando que se encuentre ejecutando una interrupción. [11], [16]

1.9.2 Vector de Interrupciones

Cuando la interrupción se genera, el procesador ejecuta la rutina que se encuentra a partir de una posición de memoria fija, conocida como vector de interrupción. En los microcontroladores puede existir más de un vector de este tipo, uno para la interrupción enmascarable y otro para la no enmascarable. [11]

1.10 Arduino

La placa de desarrollo más popular es Arduino, la idea principal fue entregar acceso a MCU embebidos, pensando en proyectos de diseño interactivo. Fue diseñado en el año 2005 en el Instituto de diseño Interactivo de Ivrea Italia.

Arduino es una plataforma de hardware libre basada en una placa con un microcontrolador y un entorno de desarrollo de *open source* (código abierto) que basa su funcionamiento en una placa con entradas y salidas (analógicas y digitales), incorpora un microprocesador que permite la programación con un lenguaje de alto nivel, el cual se encarga de efectuar los procesos matemáticos y lógicos de igual forma gestiona los recursos para cada componente externo que sea conectado a la placa principal.

Debido a que Arduino incorpora una serie de entradas analógicas y digitales se pueden conectar distintos sensores y otras placas de expansión o *shields*, kits y accesorios. [18]

1.11 Hardware de Arduino

Una plataforma *hardware* puede referirse a la arquitectura de una computadora o de un procesador. En Arduino la plataforma *hardware* consiste, principalmente en una placa electrónica que contiene una unidad de control, así como diferentes entradas y salidas que pueden ser analógicas o digitales, que conectan el mundo físico real con el virtual. De esta forma mediante sensores y actuadores y junto con las instrucciones dadas a la unidad de control de la placa, se puede interactuar de diferentes maneras con el medio. [19]

El *hardware* de Arduino, tiene una placa de circuito impreso con un microcontrolador (Atmel AVR).

Una de las principales características de Arduino se trata de una plataforma *open source* o libre. El *hardware* de código abierto consiste en compartir los diferentes diseños o desarrollos elaborados con distintos elementos de hardware para que otras personas puedan aprender, fabricar, modificar e incluso utilizar estos elementos con propósitos comerciales.

Para considerar que un *hardware* es libre, se debe considerar lo siguiente:

- Es necesario publicar la documentación incluyendo los archivos de los diseños, para efectuar su modificación y distribución.
- Se debe definir qué parte del diseño es abierta.
- Debe entregar el *software* necesario para leer el archivo del diseño y la documentación adecuada que se relaciona con sus funcionalidades, así se podrá escribir el código necesario en forma sencilla.
- Ofrecer una licencia que permita producir derivados y modificaciones.
- La licencia no debe discriminar ni restringir campos o actividades.
- La licencia no debe restringir otro *hardware* ni otro *software*, además debe ser neutral, sin basarse en tecnologías específicas, partes o componentes, materiales o interfaces de su uso. [20]

La placa Arduino es *hardware* libre porque sus archivos esquemáticos están disponibles para descargar de la página web del proyecto con la licencia *Creative Commons Attribution Share-Alike*.

1.12 Software de Arduino

Software libre es aquel *software* que da a los usuarios la libertad de poder ejecutarlo, copiarlo y distribuirlo, estudiarlo, cambiarlo y mejorarlo, sin tener que pedir ni pagar permisos al desarrollador ni a una entidad en específico.

Según la *Free Software Foundation.* organización encargada de fomentar el uso y desarrollo del *software* libre a nivel mundial, un software para ser considerado libre ha de ofrecer a cualquier persona u organización cuatro libertades básicas e imprescindibles:

1. Libertad 0: la libertad de usar el programa con cualquier propósito y en cualquier sistema informático
2. Libertad 1: la libertad de estudiar cómo funciona internamente el programa, y adaptarlo a las necesidades particulares. El acceso al código fuente es un requisito previo para esto.
3. Libertad 2: la libertad de distribuir copias.
4. Libertad 3: la libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. El acceso al código fuente es un requisito previo para esto. [18]

Un programa es *software* libre si los usuarios tienen todas estas libertades.

El *software* Arduino es libre porque se publica con una combinación de la licencia GPL (*General Public License*, Licencia Pública General) para el entorno visual de programación, y la licencia LGPL (*GNU Lesser General Public License*, Licencia Pública General Reducida de GNU) para los códigos fuente de gestión y control del microcontrolador a nivel interno. [18] [25]

1.13 Características generales de Arduino

Arduino es libre y extensible: cualquiera que desee ampliar y mejorar tanto el diseño *hardware* de las placas como el entorno de desarrollo *software* y el propio lenguaje de programación puede hacerlo.

Su entorno de programación es multiplataforma: se puede instalar y ejecutar en sistemas Windows, Mac OS y Linux.

Su entorno de lenguaje de programación es simple y claro: es muy fácil de aprender y de utilizar, a la vez que es flexible y completo para que los usuarios avanzados puedan aprovechar y exprimir todas las posibilidades del *hardware*.

Las placas de Arduino son de bajo costo.

Las placas de Arduino son reutilizables y versátiles: reutilizables porque se puede aprovechar la misma placa para varios proyectos y versátiles porque las placas Arduino proveen varios tipos de entradas y salidas de datos, los cuales permiten capturar información.

Los bloques básicos del *hardware* Arduino se observan en la figura 1.12, se representa la placa Arduino UNO, la más utilizada y que sirve como referencia para otras placas. [20]

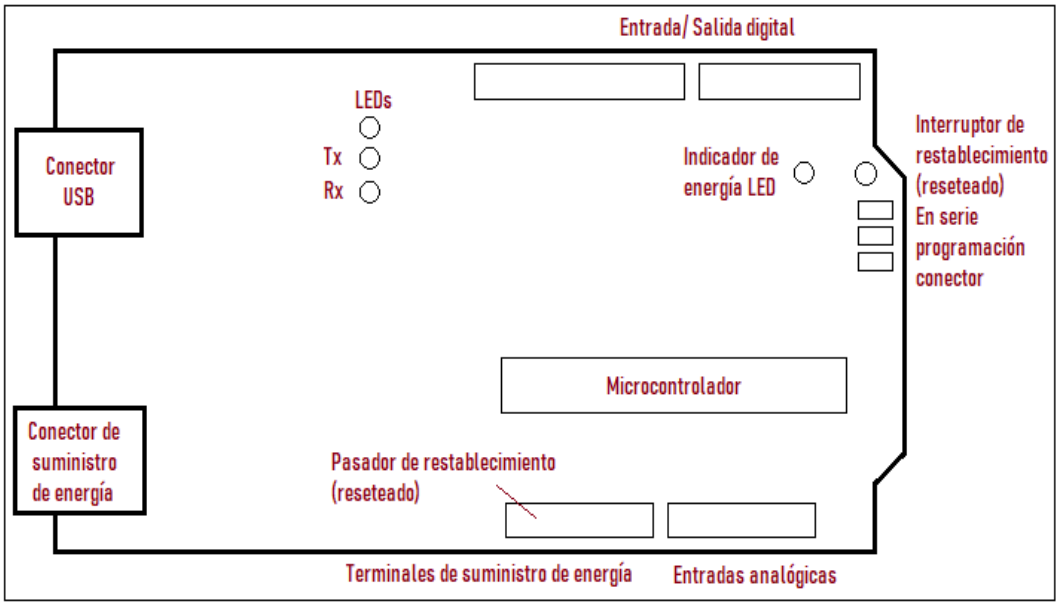


FIGURA 1. 12. DIAGRAMA A BLOQUES QUE CONFORMAN LA PLACA ARDUINO UNO.

Bolton, W. (2013). *Mecatrónica: Sistemas de control electrónico en la ingeniería mecánica y eléctrica*. Alpha Editorial.

- Interruptor de restablecimiento (reseteado) en los conectores de energía de sección.
  Esto restablece el microcontrolador de modo que inicie el programa desde el arranque. Para hacer un restablecimiento es necesario que este interruptor se conecte a 0V.
- Otros interruptores en los conectores de energía
  Estos suministran voltajes diferentes, es decir 3.5[V], 5[V], GND y 9[V].
- Entradas analógicas
  Están etiquetadas de A0 a A5 y pueden usarse para detectar las señales de voltaje.
- Conexiones digitales
  Etiquetadas de 0 a 13, pueden usarse tanto para entradas como para salidas. Las dos primeras de estas conexiones están etiquetadas Rx y Tx para la recepción y la transmisión en la comunicación.
- Conector USB
- Conector de programación en serie
  Medio para la programación de Arduino sin usar el puerto USB
- LED
  La tarjeta está equipada con tres LED, uno para indicar la transmisión en serie (Tx), otro para la recepción (Rx) y un LED extra para usarse en proyectos.

- Conector de suministro de energía
- Microcontrolador

  El microcontrolador ATmega328 viene pre-programado con un cargador de arranque que permite la carga del código nuevo de programa sin el uso de un programador externo de *hardware*.

  Para instalar un nuevo *software* es necesario instalar el software Arduino y cargar los controladores USB en su computadora. [24]

1.14 Familias de Arduino

Existen varias placas Arduino, cada una con características específicas que se deben conocer para poder elegir el modelo que más convenga según sea al caso.

1.14.1 Familia Nano

La familia Nano tiene un conjunto de sensores integrados, como temperatura / humedad, presión, micrófono y más. Va desde el económico y básico, hasta un nano con módulos de radio *Bluetooth/Wi-Fi*. Se pueden programar con *MicroPython* y es compatible con *Machine Learning*. [21] [23]

Placas Arduino de la familia nano y sus principales características.

Modelo		Arduino Nano 33 IoT	Arduino Nano RP2040 Connect	Arduino Nano 33 BLE Sense	Arduino Nano 33 BLE	Arduino Nano Every	Arduino Nano	Arduino Nano Motor Carrier
Microcontrolador		SAMD21 Cortex-M0 +32bit low power ARM MCU	Raspberry Pi RP2040	nRF52840	nRF52840	ATMega4 809	ATmega3 28	Atsamd11 (Arm Cortex-M0 48 [MHz])
Pines	Analógicos	8	8	8	A través de PWM	8	8	4
	Digitales	14	20	14	14		22	4
Memoria	FLASH	256 [kB]	16 [MB]	1 [MB]	1 [MB]	48 [kB]	32 [kB]	
	EEPROM	No	No	No	No	256 bytes	1 kB	
	SRAM	32 [kB]	250 [kB]	256 [kB]	256 [kB]	6 [kB]	2 [kB]	
Frecuencia		48 [MHz]	133 [MHz]	64 [MHz]	64 [MHz]	20 [MHz]	16 [MHz]	

TABLA 1. 5. FAMILIA NANO.

1.14.2 Familia MKR

La familia MKR se basan en un procesador de bajo consumo, con protectores y soportes están diseñados para ampliar las funciones: como sensores ambientales, GPS, Ethernet, control de motores y matriz RGB. Cada placa está equipada con un módulo de radio (excepto MKR Zero), que permite la comunicación Wi-Fi, Bluetooth, LoRa, Sigfox, NB-IoT.

Placas Arduino de la familia MKR y sus principales características.

Modelo		Arduino MKR 1000 WiFi	Arduino MKR WiFi 1010	Arduino MKR WAN 1300	Arduino MKR WAN 1310	Arduino MKR GSM 1400	Arduino MKR 4000	Arduino MKR Zero
Microcontrolador		SAMD21 Cortex-M0+32BIT low power ARM MCU	SAMD21 Cortex-M0+32BIT low power ARM MCU	SAMD21 Cortex-M0+32BIT low power ARM MCU	SAMD21 Cortex-M0+32BIT low power ARM MCU	SAMD21 Cortex-M0+32BIT low power ARM MCU	Intel Cyclone 10CL016	SAMD21 Cortex-M0+32BIT low power ARM MCU
Pines	Analógicos	7	7	7	7	7	No	7
	Digitales	8	8	8	8	8	22	22
Memoria	FLASH	256 [kB]	256 [kB]	256 [kB]	256 [kB]	256 [kB]	2 [MB]	256 [kB]
	EEPROM	No	No	No	No	No	No	No
	SRAM	32 [kB]	32 [kB]	32 [kB]				
Frecuencia		48 [MHz]	48 [MHz]	48 [MHz]	48 [MHz]	48 [MHz]	48 [MHz] 200 [MHz]	48 [MHz]

TABLA 1. 6. FAMILIA MKR.

1.14.3 Familia Clásica

En la familia clásica esta la placa mítica Arduino UNO y otros clásicos como Leonardo y Micro. [21] [23]

Placas Arduino de la familia clásica y sus principales características.

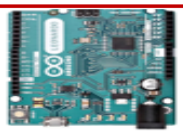
Modelo	 Arduino UNO R3	 Arduino Mega 2560 Rev3	 Arduino Leonardo	 Arduino Mini	 Arduino Due	 Arduino Micro	Arduino Zero
Microcontrolador	ATemega328 P	ATmega2 560	ATmega3 2u4	ATmega32 8P	ATmega32 8P	ATmega32u 4	ATSAMD2 1G18
Pines Analógicos	6	16	12	6	12	12	7
Pines Digitales	14	54	20	14	54	20	20
Memoria FLASH	32 [kB]	256 [MB]	32 [kB]	32 [kB]	512 [kB]	32 [kB]	256 [kB]
Memoria EEPROM	1 [kB]	4 [kB]	1 [kB]	1 [kB]	No	1 [kB]	No
Memoria SRAM	2 [kB]	8 [kB]	2.5 [kB]	2 [kB]	96 [kB]	2.5 [kB]	32 [kB]
Frecuencia	16 [MHz]	16 [MHz]	16 [MHz]	16 [MHz]	84 [MHz]	16 [MHz]	48 [MHz]

TABLA 1. 7. FAMILIA CLÁSICA.

1.15 El IDE de Arduino

El IDE (*Integrated Development Environment*) es un programa que se compone de una serie de herramientas que ayudan en tareas de programación. Existen IDEs que nos permiten trabajar con varios lenguajes de programación y plataformas, así también algunos específicos, como Arduino IDE.

Entre las características más destacadas de Arduino IDE, se encuentran:

Gratuito y libre: por lo que solo necesitamos acceder al sitio web oficial.

El IDE de Arduino se distribuye como un programa empaquetado, con lo necesario para programar: Editor de código, Compilador y Depurador. [22]

1.16 Comunicación en Arduino

Cuando se desea transmitir un conjunto de datos desde un componente a otro, se puede hacer de múltiples formas.

Una comunicación en serie/serial, en este tipo de comunicación la información es transmitida bit a bit, por un único canal, enviando un solo bit en cada momento.

Una comunicación en paralelo, en la que se envían varios bits simultáneamente, cada uno por un canal separado y sincronizado con el resto.

Existen muchos protocolos y estándares basados en la transferencia de información en serie.

1.16.1 Protocolo de comunicación I^2C

I^2C (*Inter-Integrated Circuit*) también conocido con el nombre de TWI (*TWo-wIre*), comunica circuitos integrados entre sí. Su principal característica es que utiliza dos líneas para transmitir la información: una llamada línea SDA, sirve para transferir datos y otra llamada línea SCL, sirve para enviar la señal de reloj.

Por señal de reloj se entiende una señal binaria de una frecuencia periódica muy precisa que sirve para coordinar y sincronizar los elementos integrantes de una comunicación para saber cuándo empieza, cuánto dura y cuándo acaba la transferencia de información.

Cada dispositivo conectado al bus I^2C tiene una dirección única que lo identifica y puede estar configurado como maestro o como esclavo. Un dispositivo maestro es el que inicia la transmisión de datos y además genera la señal de reloj. [26]

FIGURA 1. 13. DIAGRAMA PROTOCOLO I^2C

TORRENTE, ÓSCAR. (2013). ARDUINO: CURSO PRÁCTICO DE FORMACIÓN. ALPHA EDITORIAL.

1.16.2 Protocolo de comunicación SPI

SPI (*Serial Peripheral Interface*), al igual que I^2C, un dispositivo conectado al bus SPI puede ser maestro o esclavo, donde el primero es el que inicia la transmisión de datos y además genera la señal de reloj.

La mayor diferencia entre el protocolo SPI y el I^2C, es que el primero requiere de cuatro líneas en vez de dos. Una línea llamada SCK envía a todos los dispositivos la señal de reloj generada por el maestro actual, otra llama SS utilizada por ese maestro para elegir en cada momento con qué dispositivo esclavo se quiere comunicar, otra llamada MOSI es la línea utilizada para enviar los datos y otra llamada MISO utilizada para enviar los datos en sentido contrario. [24]

1.17 Tipos de Entradas / Salidas (puertos)

Las entradas y salidas son el medio por el cual el microcontrolador se comunica con el mundo exterior, emiten señales lógicas, una señal lógica solo puede tomar dos valores, un valor alto igual a 1 y un valor bajo igual a 0.

Las placas Arduino facilitan el acceso a estas entradas/salidas distribuyéndolas alrededor del circuito. Usando los conectores hembra estándares de 2.45 [mm], se conectan los *shields* y los diferentes circuitos adicionales utilizados. Normalmente, estos conectores funcionan con un voltaje de 5[V] y solo puede recibir y emitir 40 [mA] de corriente. [19] [27]

1.17.1 Entradas digitales

Las placas de Arduino disponen de diferentes números de pines-hembra de acuerdo al modelo. Es en estos pines donde conectamos sensores para que la placa pueda recibir datos del entorno, y también donde se conectan los actuadores para que la placa pueda enviar las ordenes, además de conectar cualquier otro componente que necesite comunicarse con la placa de alguna manera. A estos tipos de pines-hembra digitales de "propósito general" se les llama pines GPIO (*General Purpose Input/Output*). [20] [27]

Una señal digital solo puede tener dos estados:

0 (*LOW*, bajo, false): 0[V], enviados desde la placa.

1 (*HIGH*, alto, true): 5[V], enviados desde la placa.

Por lo tanto, cuando ponemos un pin digital a valor *HIGH* la placa suministra 5[V] por la salida que hayamos indicado, y si ponemos el valor a *LOW* suministrará 0[V].

Todos los pines-hembra digitales funcionan a 5 [V], pueden proveer o recibir un máximo de 40 [mA] y disponen de una resistencia "*pull-up*" interna entre 20 [KΩ] y 50 [KΩ] que inicialmente está desconectada (a menos que se indique lo contrario mediante programación *software*).

1.17.2 Entradas analógicas

Los pines analógicos pueden recibir un rango de valores de voltaje, a diferencia de los digitales que solo entienden dos valores: 0 y 1, ó 0[V] ó 5[V].

Con los pines analógicos se pueden leer valores intermedios entre 0[V] y 5[V], representados con un valor entero comprendido entre 0 y 1023, ya que la información se representa en números de 10 bits. El símbolo "~" en la placa indica que puede ser utilizados como pines analógicos.

No obstante, la electrónica de la placa tan solo puede trabajar con valores digitales, por lo que es necesaria una conversión previa del valor analógico recibido a un valor digital lo más aproximado posible. Se realiza mediante un circuito analógico/digital incorporado en la propia placa.

1.18 Raspberry Pi

Las placas fueron desarrolladas por la empresa *Raspberry Pi Foundation* en 2012 para fomentar el aprendizaje de la informática. A diferencia de otras plataformas, Raspberry Pi conserva un *hardware* propio, con ello mantiene el control sobre la creación y la fabricación de sus placas. Raspberry Pi está diseñada en una arquitectura de microcontrolador ARM, comúnmente utilizado en teléfonos inteligentes. Los primeros diseños de Raspberry Pi se basan en el microcontrolador Atmel ATmega644.

Desde su lanzamiento en febrero de 2012 la placa de Raspberry Pi ha pasado por una serie de revisiones y ha estado disponible en dos modelos, el Modelo A y el Modelo B. El Modelo A es más barato y simple, mientras que el Modelo B incluye soporte para conectividad Ethernet.

Raspberry Pi es una computadora que está compuesta por un CPU, memoria RAM, conectividad de red, puertos de entrada y de salida. Sin embargo, no cuenta con un botón de encendido apagado. Solo se tiene que conectar a la corriente para su funcionamiento.

Raspberry Pi cuenta con sistemas integrados, como Linux para facilitar el desarrollo de dispositivos que faciliten los desafíos en edificios inteligentes, Internet de las cosas (IoT), ciudades inteligentes. [29]

1.18.1 Hardware de la Raspberry Pi

Raspberry Pi utiliza un microprocesador con arquitectura ARM, memoria RAM y tarjeta gráfica o GPU (*Graphics Processing Unit*, Unidad de Procesamiento Gráfico). El diseño no incluye disco duro, para su almacenamiento utiliza una tarjeta SD externa.

1.18.1.1 GPU

La unidad de procesamiento de gráficos (GPU) es un chip especializado utilizado para mejorar las matemáticas complejas necesarias para representar gráficos.

1.18.1.2 GPIO

Los pines GPIO (E/S de propósito general), también conocidos como BCM o *Broadcom* son los elementos que actúan como interfaz a conectarse con otros módulos, cuenta con 40 pines GPIO en la placa Raspberry Pi 4 modelo B, cualquiera de estos pines se puede habilitar en el software como pin de E/S.

La numeración de los pines no está en orden numérico. Por ejemplo, los pines GPIO 0 y 1 se encuentran presentes en la placa con el número 27 y 28, pero estos están reservados para uso avanzado. [29] [30]

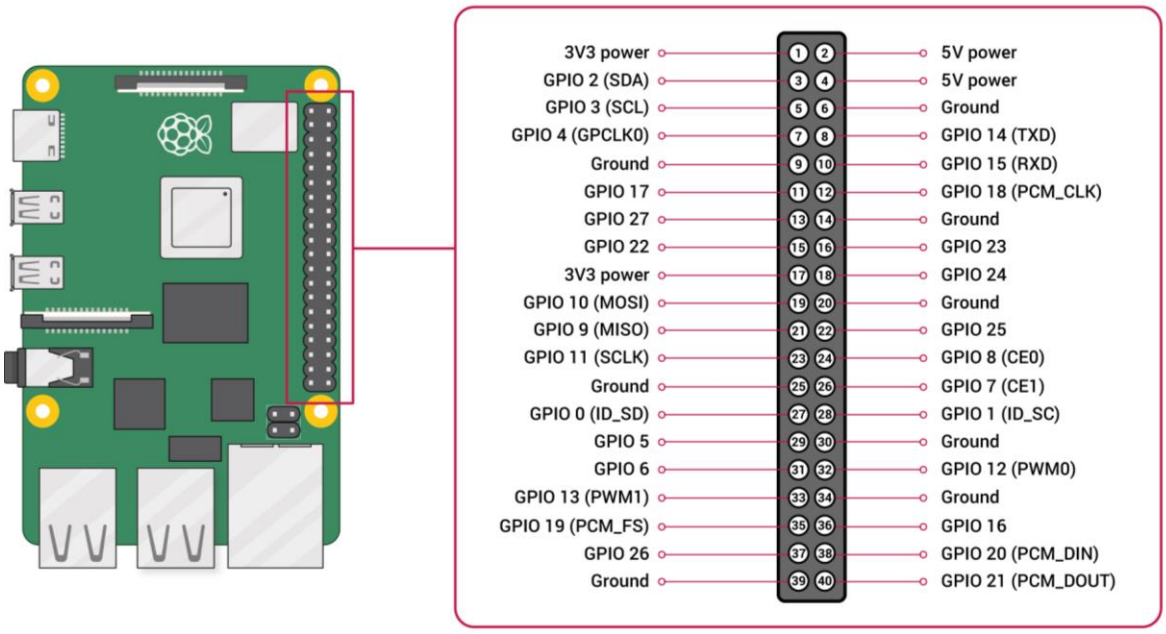


FIGURA 1. 14. GPIO DE LA PLACA RASPBERRY PI 4 DONDE SE MUESTRAN LOS 40 PINES CON LOS QUE CUENTA LA PLACA.

RASPBERRY PI DOCUMENTATION - RASPBERRY PI HARDWARE. (S. F.).
HTTPS://WWW.RASPBERRYPI.COM/DOCUMENTATION/COMPUTERS/RASPBERRY-PI.HTML

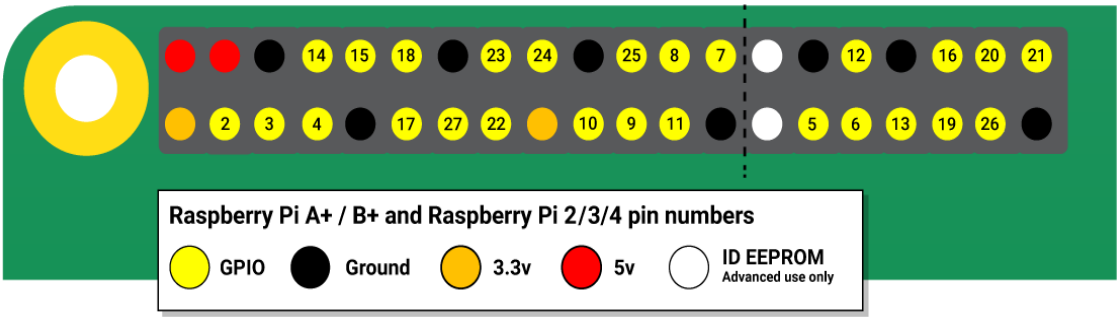


FIGURA 1. 15. NUMERACIÓN DE LOS PINES GPIO DE RASPBERRY PI 4.

Raspberry Pi Documentation - Raspberry Pi Hardware. (s. f.). https://www.raspberrypi.com/documentation/computers/raspberry-pi.html

1.18.1.3 Voltaje

La Raspberry Pi cuenta con dos pines de 5 [V] y dos de 3.3 [V] en la placa, de igual manera cuenta con varios pines de tierra, los cuales no se pueden configurar. Los pines restantes son de uso general con un voltaje de 3.3 [V].

1.18.1.4 Entradas y Salidas

Un pin GPIO habilitado como pin de E/S puede configurarse en Alto (3.3 [V]) o en Bajo (0 [V]). Esta acción se realiza por medio de resistores *pull-up* o *pull-down* internas. Los GPIO2 y GPIO3 cuentan con resistores *pull-up* fijas, en cuanto a otros pines se pueden configurar por medio del software.

1.18.2 Software de Raspberry Pi

Los Sistemas Operativos para PC o computadoras más utilizados son: Windows, Mac y Linux. Sin embargo, la placa Raspberry Pi está basada en el sistema operativo Linux.

La Raspberry Pi cuenta con un sistema operativo propio llamado Raspberry Pi OS anteriormente llamado Raspbian, el cual es gratuito, este sistema cuenta con 35,000 paquetes de software pre-compilado. El sistema operativo Raspberry Pi OS se encuentra en constante cambio, ya que se realizan mejoras en la estabilidad y el rendimiento de los paquetes Debian.

1.18.3 Familias de Raspberry Pi

La familia Raspberry Pi ha tenido 4 versiones, de las cuales el primer Modelo tuvo versiones A, B y B+, el Modelo 2 solo cuenta con la versión B, el Modelo 3 con versiones A, B y B+ y el Modelo 4 la versión B.

En la siguiente tabla se pueden observar características generales de cada uno de los Modelos de las familias Raspberry Pi.

	SoC	CPU	GPU	RAM	USB	V/A	Boot	Red	Alimentación
Modelo A	Broadcom BCM2835	700MHz ARM1176JZF-S	VideoCore IV	256 MB	1	RCA Jack HDMI	SD	No	300mA 1.5w/5v MicroUSB GPIO
Modelo A+	Broadcom BCM2835	700MHz ARM1176JZF-S	VideoCore IV	256 MB	1	Jack HDMI	uSD	No	400mA 2W / 5v MicroUSB GPIO
3 Modelo A+	Broadcom BCM2837B0	1,4GHz QUAD ARM Cortex-A53	VideoCore IV	512 MB	1	Jack HDMI	uSD	Dual-band WiFi, BT	2, 5A 12,5w / 5v MicroUSB GPIO
Modelo B	Broadcom BCM2835	700MHz ARM1176JZF-S	VideoCore IV	512 MB	2	RCA Jack HDMI	SD	ETH 10/100	700mA 3.5w/5v MicroUSB GPIO
Modelo B+	Broadcom BCM2835	700MHz ARM1176JZF-S	VideoCore IV	512 MB	4	Jack HDMI	uSD	ETH 10/100	500mA 2.5w/5v MicroUSB GPIO
2 Modelo B	Broadcom BCM2836	900MHz QUAD ARM Cortex-A7	VideoCore IV	1 GB	4	Jack HDMI	uSD	ETH 10/100	800mA 4w / 5v MicroUSB GPIO
3 Modelo B	Broadcom BCM2837	1,2GHz QUAD ARM Cortex-A53	VideoCore IV	1 GB	4	Jack HDMI	uSD	ETH 10/100 WiFi, BT	2,5A 12,5w / 5v MicroUSB GPIO
3 Modelo B+	Broadcom BCM2837B0	1,4GHz QUAD ARM Cortex-A53	VideoCore IV	1 GB	4	Jack HDMI	uSD	ETH 10/100/300(USB) Dual-band WiFi, BT	2,5A 12,5w / 5v MicroUSB GPIO PoE (HAT)
4 Modelo B	Broadcom BCM2711	1,5GHz QUAD ARM Cortex-A72	VideoCore IV	1,2 o 4 GB	2(2.0) 2(3.0)	Jack 2 micro HDMI	uSD	ETH 1000 Dual-band WiFi, BT	2,5A 12,5w / 5v USB-C GPIO PoE (HAT)
Zero	Broadcom BCM2835	1GHz ARM1176JZF-S	VideoCore IV	512 MB	1 Micro	Mini HDMI	uSD	No	160mA 0.8w / 5v MicroUSB GPIO
Zero W	Broadcom BCM2835	1GHz ARM1176JZF-S	VideoCore IV	512 MB	1 Micro	Mini HDMI	uSD	WiFi, BT	160mA 0.8w / 5v MicroUSB GPIO

Tabla 1. 8. Familias de Raspberry Pi.

Jecrespom. (2022, 16 abril). Hardware Raspberry Pi. Aprendiendo Arduino. https://aprendiendoarduino.wordpress.com/2022/04/16/hardware-raspberry-pi-2/

1.18.4 Tarjeta de Memoria

La tarjeta de memoria tiene dos características principales que son su capacidad de almacenamiento y la velocidad de transferencia. Las tarjetas de memoria se clasifican en diversos tipos de memoria.

Raspberry Pi utiliza la tarjeta microSD la cual actúa como almacenaje, el software, el sistema operativo, además de los archivos que se crean se almacenan en la tarjeta.

1.19 Comunicación en Raspberry Pi

Los puertos y los buses son los elementos que nos permiten tener una comunicación entre los dispositivos del microcontrolador.

Los puertos paralelos envían información de forma paralela y por lo tanto se tienen un mayor número de cables, pistas, etc. Los puertos serie son los más utilizados tales como: USB, I^2C, $SPI, UART, CAN\ bus, RS232$, etc.

Los buses que tiene Raspberry Pi son I^2C, SPI y UART.

1.19.1 Bus de serie I^2C

El bus I^2C (*Inter Integrated Circuit*) consta de dos señales: SCL y SDA, donde SCL es la señal de reloj y SDA es la señal de datos. La señal de reloj siempre la genera el maestro, aunque suceden algunas excepciones donde el esclavo puede modificar el reloj según sus necesidades. Este protocolo se puede utilizar con tramos cortos de cable de hasta 3 metros, la comunicación de I^2C es *half-duplex*.

El uso de I^2C tanto la línea SDA y SCL tienen resistencias *pull-up* de 2.2 [kΩ] a el voltaje de alimentación de 5 [V].

La comunicación es maestro esclavo por lo cual en cualquier instante uno de los dispositivos toma el papel de maestro y el resto de esclavos

El puerto I^2C permite conectar sensores, pantallas u otros dispositivos. [32] [33]

1.19.2 Bus serie SPI

El bus SPI (*Serial Peripheral Interface*) se utiliza para la transferencia de información entre circuitos electrónicos, el cual es regulado por reloj.

Al igual que el bus I^2C, el bus SPI se comunica por medio de una relación maestro esclavo, donde la comunicación es bidireccional (*full-duplex*).

La sincronización y la transmisión de datos se realiza por medio de 4 líneas o señales las cuales son, SCLK (*Clock*), MOSI (*Master Output Slave Input*), MISO (*Master Input Slave Output*) y SS/*Select*/CS (*Chip Select*).

El puerto SPI permite conectar sensores atmosféricos, EEPROMS, *display*, etc. Al contrario que

I^2C, el bus SPI no requiere de resistencias *pull-up* lo cual da como resultado un menor consumo de energía. Una de sus desventajas en que necesita una línea por cada esclavo lo cual se traduce en más pines. [32] [33]

1.19.3 Bus UART

El módulo UART (*Universal Asynchronous Receiver Transmiter,* transmisor/receptor asíncrono universal) para comunicaciones.

Los pines de la UART en Raspberry Pi son el pin 6, el 8 para la transmisión y el 10 para la recepción. La comunicación serie a través de UART se trata de envío de datos a través de dos líneas una para la transmisión y la otra para la recepción. En UART la información es transmitida de bit a bit, agrupados en un paquete de datos (*frame*). Este tipo de comunicación es asíncrona ya que no requiere de una línea de reloj externa para sincronizar los dispositivos. [32] [33]

Referencias de PIC, Arduino y Raspberry Pi

PIC

[1] Budris, Paula. (2013). Microcontroladores PIC. Buenos Aires: Colección Técnico en electrónica.

[2] Budris, Paula. (2013). Microprocesadores y Microcontroladores. Buenos Aires: Colección Técnico en electrónica.

[3] Rivamar, Alfredo. (2014). Fundamentos de microprocesadores conocimientos y herramientas indispensables. Cuidad Autónoma de Buenos Aires: Manuales USERS.

[4] Bolton, William. (2013). Mecatrónica: Sistemas de control electrónico en la ingeniería mecánica y eléctrica. Un enfoque multidisciplinario. México: ALFAOMEGA.

[5] Benchimol, Daniel. (2011). Microcontroladores. Buenos Aires: Manuales USERS.

[6] Aranda, Diego. (2014). Electrónica: técnicas digitales y microcontroladores. Cuidad Autónoma de Buenos Aires: Manuales USERS.

[7] Zuloaga, Aitzol; Astarloa, Armando. (2008). Sistemas de Procesamiento Digital. Delta: Madrid (España).

[8] Orduña, Manuel; Arnau, Vicente. (1996). Arquitectura y programación de microcontroladores. Colección Ingeniería Informática: España.

[9] Benchimol, Daniel. (2011). Microcontroladores. Buenos Aires: Colección desde cero USERS.

[10] Galeano, Gustavo. (2009). Sistemas Embebidos en C teoría y prácticas aplicadas a cualquier microcontrolador. Alfaomega Grupo Editor: México.

[11] Valdés, Fernando; Pallas, Ramon. (2007). Microcontroladores: Fundamentos y aplicaciones con PIC. Primera Edición. México: ALFAOMEGA Grupo Editor.

[12] Angulo, Usategui, José; Angulo, Martínez, Ignacio. (2003). Microcontroladores PIC diseño práctico de aplicaciones primera parte. El PIC16F84 lenguaje PBASIC y Ensamblador. España: McRAW-HILL

[13] Angulo, Usategui, José; Angulo, Martínez, Ignacio; Romero, Yesa, Susana. (2003). Microcontroladores PIC diseño práctico de aplicaciones segunda parte. El PIC16F87x lenguaje PBASIC y Ensamblador. España: McRAW-HILL.

[14] A. Reyes, Carlos. (2008). Microcontroladores PIC Programación en Basic. Quito-Ecuador: RISPERGRAF.

[15] Dogan, Ibrahim. (2006). Microcontroller Based Applied Digital Control. John Wiley y Sons, Ltd: USA.

[16] Rossano, Victor. (2009). Electrónica & Microcontroladores PIC. Manuales USERS: Argentina

ARDUINO

[18] Torrente, Óscar. (2013). Arduino. Curso práctico de formación. ALFAOMEGA.

[19] Goilav, Nicolas; Loi, Geoffrey. (2016). Arduino aprender a desarrollar para crear objetos inteligentes. Ediciones ENI.

[20] Lajara, José; Pelegrí, José. (2014). Sistemas integrados con Arduino. ALFAOMEGA.

[21] Arduino (12 de mayo del 2023). Hardware de Arduino. Arduino. https://www.arduino.cc/en/hardware.

[22] Caicedo P., Antonio. (2017). Arduino para principiantes.

[23] Beiroa, Rubén. (2019). Aprender Arduino, electrónica y programación con 100 ejercicios prácticos. ALFAOMEGA.

[24] López, Eugenio. (2016). Arduino. Guía práctica de fundamentos y simulación. Ra-Ma.

[25] Users, Staff. (2020). Descubriendo Arduino. Creative Andina Corp.

[26] Novillo, Johnny; Hernández, Dixys; Mzón, Bertha; Molina Jimmy; Cárdenas Oscar. (2018). Arduino y el internet de las cosas. Ciencias.

[27] Porcuna L., Pedro. (2016). Robótica y domótica básica con Arduino. Ra – Ma. Ediciones de la U.

[28] Bolton, William. (2013). Mecatrónica. Sistemas de control electrónico en la ingeniería mecánica y eléctrica. Un enfoque multidisciplinario. ALFAOMEGA.

RASPBERRY

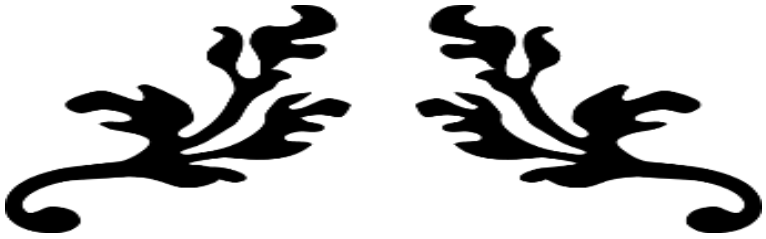
[29] López, Eugenio. (2017). Raspberry Pi Fundamentos y aplicaciones. Ra-Ma. Madrid.

[30] Molloy, Derek. (2012). Exploring Raspberry Pi Interfacing to the Real World with Embedded Linux. WILEY.

[31] Dennis, Andrew k. (2016). Raspberry Pi Computer Architecture Essentials. PACKT PUBLISHING.

[32] López, Juan Pedro. (2019). Guía práctica para Raspberry Pi y Beaglebone. México: Alfaomega Grupo Editor.

[33] López, Eugenio. (2017). Raspberry Pi. Fundamentos y aplicaciones. Madrid: RA-MA Editorial.



Capítulo 2

Internet de las cosas IOT

2.1 Internet de las cosas (IoT)

El internet de las cosas (*Internet of things*, IoT) está definido como una red de objetos físicos conectados a internet, los cuales logran interactuar mediante sistemas embebidos, redes de comunicación, aplicaciones en la nube, etc. Permitiendo a los objetos comunicarse entre sí, con IoT podemos acceder, almacenar, recuperar e interactuar con la información. [1]

Actualmente más personas están utilizando Internet, tras tener conectividad para todas las personas en cualquier momento. Lo que se tiene que hacer es emitir una señal, que el objeto la reconozca, conecte y envíe la información. De esta manera, está tecnología permitirá el crecimiento de la información, el incremento de interconexión entre máquinas y dispositivos inteligentes personales. [2], [3], [4]

De acuerdo con *Cisco Systems*, la evolución de internet se divide en cuatro fases distintas, teniendo cada una de ellas un efecto mayor sobre la otra:

I. Conectividad: en el año de 1990.
II. Transformación del proceso empresarial Red de la Agencia de Proyectos de Investigación Avanzados (ARPANET).
III. Digitalización cooperativa.
IV. IoT, es la fase cuatro de Internet. [5]

El término Internet de las Cosas fue utilizado por primera vez por Kevin Ashton en 1999 cuando estaba trabajando en el campo de la tecnología RFID en (Red Identificación por Radio Frecuencia) y tecnologías de detección emergentes. Sin embargo, la IoT inicio en algún momento entre 2008 y 2009. [6], [7]

IoT se refiere a una tecnología basada en la conexión de objetos cotidianos a Internet, los cuales están provistos de sensores, actuadores y tecnología de comunicación, que interactúan, recolectan y procesan información sobre el entorno físico para proporcionar servicios a los usuarios finales. Internet puede ser también una plataforma para dispositivos que se comunican electrónicamente y comparen información y datos específicos del mundo que los rodea.

IoT es de gran importancia, ya que, ha logrado que internet sea sensorial (temperatura, presión, vibración, luz, humedad, etc.). [5]

El termino IoT no tiene una definición única y universal por ello varias organizaciones tratan de definir dicho término, dando una visión particular de lo que significa el IoT.

Definiciones de Internet de las cosas:

Consejo de Arquitectura de internet (IAB). N

El término “Internet de las cosas” (IoT) denota una tendencia en que un gran número de dispositivos embebidos utilizan los servicios de comunicación que ofrecen los protocolos de Internet. A estos dispositivos suelen llamarles “objetos inteligentes” y no son operados directamente por un ser humano, sino que existen como componentes en edificios o vehículos o están distribuidos en el entorno. [6]

Unión Internacional de Telecomunicaciones (UIT, ITU).

Internet de las Cosas (IoT) es una serie de dispositivos conocidos como “cosas” inteligentes que pueden comunicarse entre sí directamente a través de la red. Estas tecnologías no solo aplican para hogares inteligentes, sino también en sectores de la cibersalud, el cibergobierno, la industria automotriz, los sistemas de información geográfica, la teledetección, las redes en el hogar, el comercio electrónico y la atenuación de los efectos de cambio climático. [8]

Internet de las cosas (IoT) es una infraestructura global para la sociedad de la información, que permite servicios avanzados mediante la interconexión de cosas (físicos y virtuales) basadas en tecnologías de información y comunicación interoperables en evolución. [9]

Organización Internacional de Normalización (ISO).

Internet de las cosas (IoT) es una infraestructura de objetos, personas, sistemas y recursos de información interconectados junto con servicios inteligentes para permitirles procesar información del mundo físico y virtual, y reaccionar. [10]

Instituto de Ingenieros Eléctricos y Electrónicos (IEEE).

Internet de las cosas (IoT) es una estructura en el que todas las cosas tienen una representación y una presencia en Internet. Más específicamente, el IoT tiene como objetivo ofrecer nuevas aplicaciones y servicios que una los mundos físico y virtual, en que las comunicaciones máquina a máquina (M2M) representan la comunicación de referencia que permite las interconexiones entre las cosas y las aplicaciones en la nube. [6]

Internet de las cosas incluye cualquier objeto o "cosa" que pueda conectarse de manera inalámbrica a una red de internet. Pero hoy en día, IoT se refiere más específicamente a cosas conectadas que estén equipadas con sensores, software y otras tecnologías que permiten transmitir y recibir datos con el propósito de informar a los usuarios o automatizar una acción.

Una vez que los dispositivos de IoT recopilen y transmitan datos, el punto final es aprender tanto como sea posible de ellos y hacer que brinden resultados e información estratégica cada vez más precisos y sofisticados. Aquí es donde entran en juego las tecnologías IA (Inteligencia Artificial): aumentar las redes de IoT con capacidades de analíticas avanzadas y *machine learning*. (forma de aprender mediante los datos y no de la programación explícita).

FIGURA 2. 1. INTERNET DE LAS COSAS (IOT).

2.2 Características de IoT

Los dispositivos de IoT tienen una función específica que es recopilación, almacenamiento y procesamiento de datos de sensores.

Y cuenta con una serie de características que son:

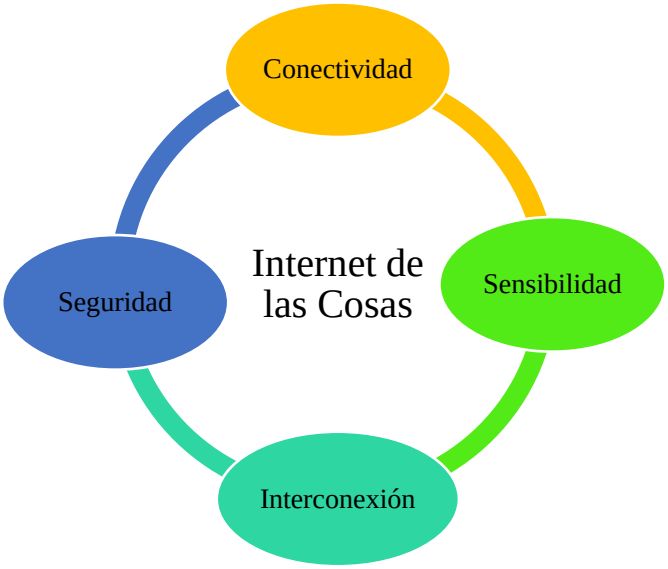


FIGURA 2. 2. CARACTERÍSTICAS DE IOT.

Conectividad: Los dispositivos deben conectarse a una red, se puede utilizar Wifi, Ethernet o Bluetooth, para interrelacionarse con otros dispositivos y usuarios.

Sensibilidad: A través de sensores los dispositivos se pueden obtener diferentes parámetros, como temperatura, movimiento, etc.

Interconexión: IoT utiliza un sistema de comunicación para establecer la comunicación de las personas, dispositivos y el mundo físico.

Seguridad: Los dispositivos que están conectados a la red deben de disponer de medios de seguridad que protejan y garanticen la integridad y privacidad de los mismos. [11]

2.3 Arquitectura de IoT

Internet de las cosas es una combinación de sensores y actuadores que proporcionan y reciben información, capaces de transmitir datos para ser utilizados. No hay un modelo de IoT establecido aún, por lo tanto, podemos encontrar varias propuestas de arquitecturas, las cuales coinciden en áreas específicas. [11]

Por lo que, IoT integra una arquitectura que se divide en cuatro capas:

Arquitectura de IoT	
Capa de detección	Sensores, los objetos físicos y la obtención de datos.
Capa de intercambio de datos	Transmisión de datos a través de redes de comunicación.
Capa de integración de la información	El procesamiento de la información adquirida de las redes, filtrado de datos no deseados e integración de información principal útil para los servicios y los usuarios finales.
Capa de servicio de aplicación	Servicios de contenido a los usuarios.

TABLA 2. 1. ARQUITECTURA DE IOT.

El Internet no cuenta con la capacidad de detección, ya que, solo interconecta los dispositivos. Por otra parte, IoT tiene la capa de detección, lo cual reduce los requisitos sobre la capacidad de los dispositivos, permitiendo así la interconexión entre los dispositivos. En el IoT se presentan nuevos requisitos para el intercambio de datos, la integración de la información y los servicios. [12]

2.4 Modelos de conectividad

La implementación del IoT se divide en 4 modelos de conectividad, los cuales están descritos por el Consejo de Arquitectura de internet (*Internet Architecture Board*, IAB): *Device to Device* (dispositivo a dispositivo), *Device to Cloud* (dispositivo a la nube), *Device to Geteway* (dispositivo a puerta de enlace) y *Back End Data Sharing* (intercambio de datos a través del *back end*). [6]

2.4.1 Comunicaciones dispositivo a dispositivo (*Device to Device Model*)

En este modelo se conectan dos o más dispositivos que se comunican entre sí, sin la intervención de un tercero. Estos dispositivos se comunican por medio de redes IP o de Internet. Para poder establecer una comunicación directa de dispositivo a dispositivo se utilizan diferentes protocolos como lo son: Bluetooth, Zigbee, Z- Wave, etc.

Por lo general este tipo de comunicación es utilizada en aplicaciones como sistemas de automatización en el hogar, ya que, la tasa de transmisión de datos es muy baja. [6] [13]

En la figura 2.1 se observa un interruptor que se comunica con un foco con el requisito de que cada uno es fabricado por empresas diferentes, identificados por fabricante A (foco) y fabricante B (interruptor). Dichos fabricantes deben ponerse de acuerdo sobre la pila de protocolos que utilizaran el foco y el interruptor. [14]

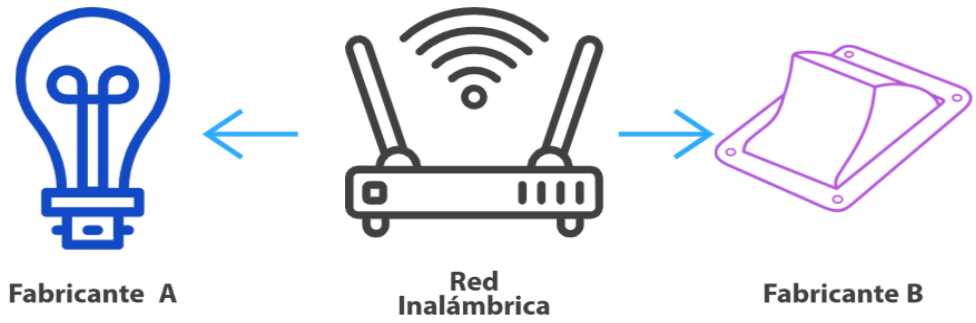


FIGURA 2. 3. MODELOS DE COMUNICACIÓN DE DISPOSITIVO A DISPOSITIVO.

Según el artículo del IETF Journal “muchas veces estos dispositivos se relacionan en forma directa, en general tienen mecanismos de seguridad y confianza integrados; además de que utilizan modelos de datos específicos para cada dispositivo “.

Lo cual da como resultado que dichas empresas desarrollen la forma de implementar los formatos de datos específicos, los cuales permitan el formato de estándares que logren la interoperabilidad entre dos dispositivos.

2.4.2 Comunicaciones dispositivo a la nube (*Device to Cloud Model*)

En este modelo los dispositivos del IoT se conectan directamente a un servicio en la nube, aprovechando así los mecanismos de comunicación existentes para establecer una conexión entre el dispositivo y la red IP, misma que se conectara al servicio en la nube. Este modelo de comunicación es utilizado en algunos dispositivos electrónicos de consumo para la IoT. Este modelo suele utilizar protocolos tales como TCP (*Transfer Control Protocol*), UDP (*User Datagram Protocol*), HTTP (*Hypertext Transfer Protocol*) y TLS (*Transport Layer Security*) para intercambios basados en seguridad. [6] [13]

Al intentar integrar dispositivos de diferentes fabricantes suelen surgir problemas de compatibilidad. En la figura 2.2 se muestra el modelo de comunicación para cargar datos de sensores a un proveedor de servicios de aplicaciones. A menudo el proveedor de servicios de aplicación también vende objetos inteligentes de tal forma que la comunicación ocurre internamente al proveedor, y no surgen problemas de incompatibilidad.

Cuando esta situación no sucede y existe una incompatibilidad con los elementos es necesario que las empresas den la posibilidad de utilizar otro sistema operativo en el dispositivo, es por eso que se necesitan protocolos de internet. [14]

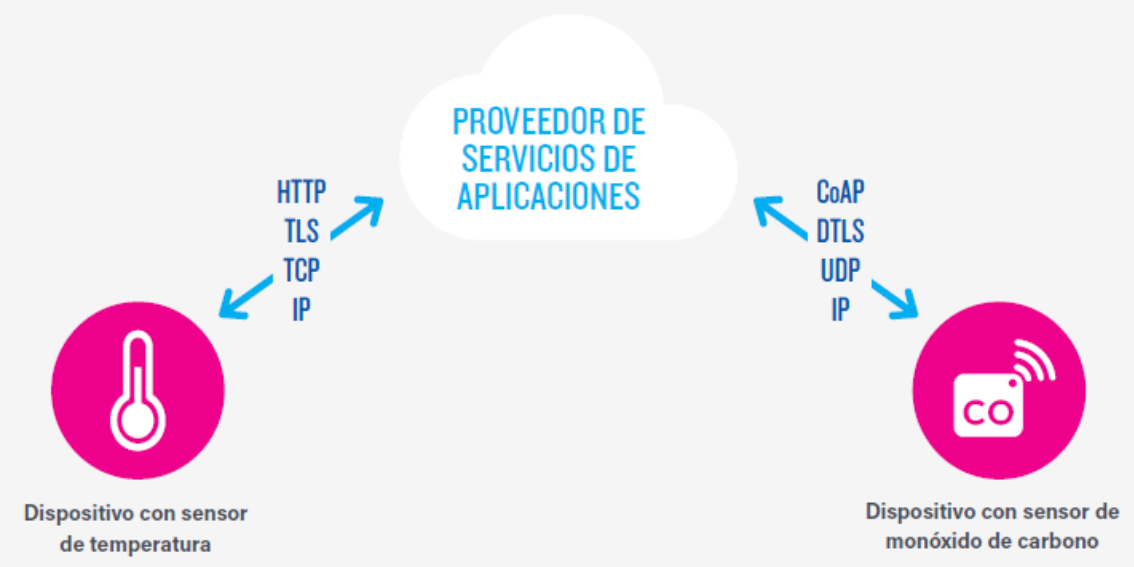


FIGURA 2. 4. MODELOS DE COMUNICACIÓN DISPOSITIVO A LA NUBE.

INTERNET SOCIETY. (2015). LA INTERNET DE LAS COSAS – UNA BREVE RESEÑA. INTERNET SOCIETY: HTTPS://WWW.INTERNETSOCIETY.ORG/WP-CONTENT/UPLOADS/2017/09/REPORT-INTERNETOFTHINGS-

2.4.3 Modelo dispositivo a puerta de enlace (*Device to Gateway Model*)

En el modelo dispositivo a puerta de enlace de la capa de aplicación (ALG), los dispositivos de IoT se conectan a través de un servicio de ALG como una forma de llegar al servicio en la nube. Esto quiere decir que un software de aplicación se encuentra ejecutándose en un dispositivo de puerta de enlace local, el cual actúa como intermediario entre el dispositivo y el servicio en la nube, que provee de seguridad, y de otras funciones.

El IETF *Journal* explica que "*Este modelo de comunicación se usa en situaciones donde los objetos pequeños deben interoperar con dispositivos que no utilizan el protocolo de Internet. A veces se adopta este enfoque para integrar dispositivos que solo soportan IPv6, lo que significa que se necesita una puerta de enlace para los dispositivos y servicios existentes que solo soportan IPv4."* [6], [14]

Este modelo de comunicación se utiliza para integrar nuevos dispositivos a un sistema heredado el cual contiene dispositivos que no son interoperables, la desventaja de este tipo de modelo es un elevado costo y la complejidad en el software, y en el sistema para ALG.

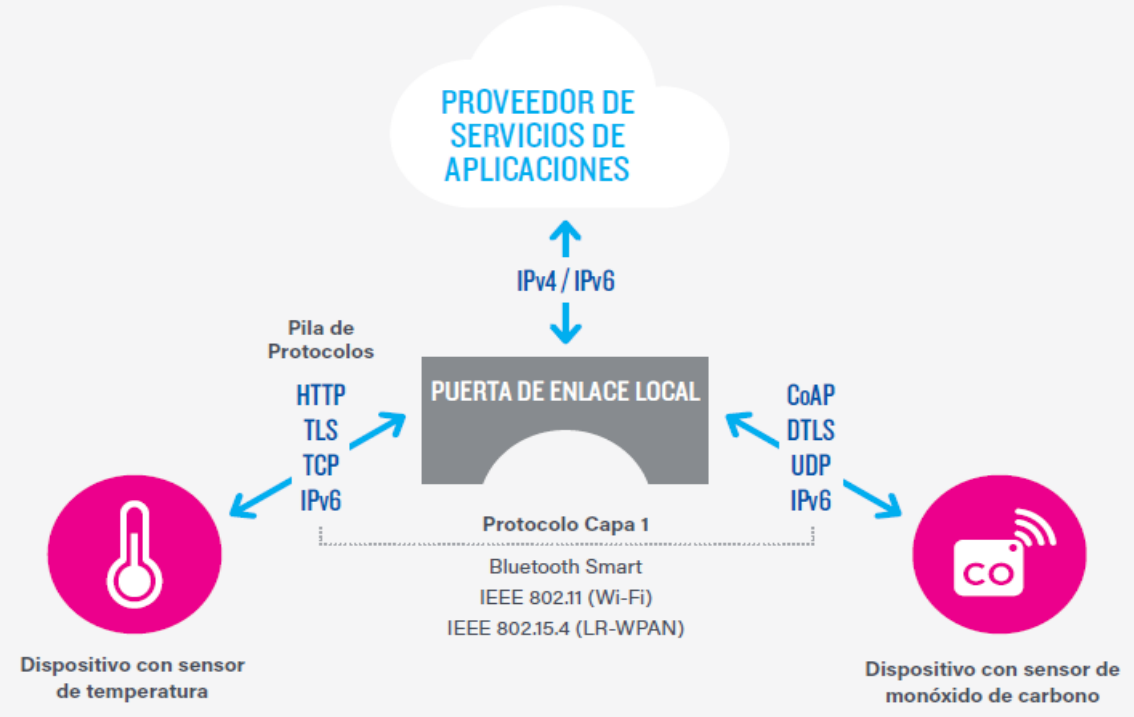


FIGURA 2. 5. MODELO DE COMUNICACIÓN DE DISPOSITIVO A PUERTA DE ENLACE.

INTERNET SOCIETY. (2015). INTERNET SOCIETY. (2015). LA INTERNET DE LAS COSAS – UNA BREVE RESEÑA. INTERNET SOCIETY: HTTPS://WWW.INTERNETSOCIETY.ORG/WP-CONTENT/UPLOADS/2017/09/REPORT-INTERNETOFTHINGS-

2.4.4 Modelo de intercambio de datos a través del back –end (*Back End Data Sharing Model*)

El modelo de intercambio de datos a través del *back-end*, se trata de una arquitectura de comunicación que permite que los usuarios exporten y analicen datos de los sensores de un servicio en la nube en combinación con datos de otras fuentes.

Este tipo de modelo propone que para lograr la interoperabilidad de los datos de los sensores alojados en la nube se requiere de una combinación de los diferentes proveedores de los servicios de nube o de interfaces de programación de aplicaciones en la nube.

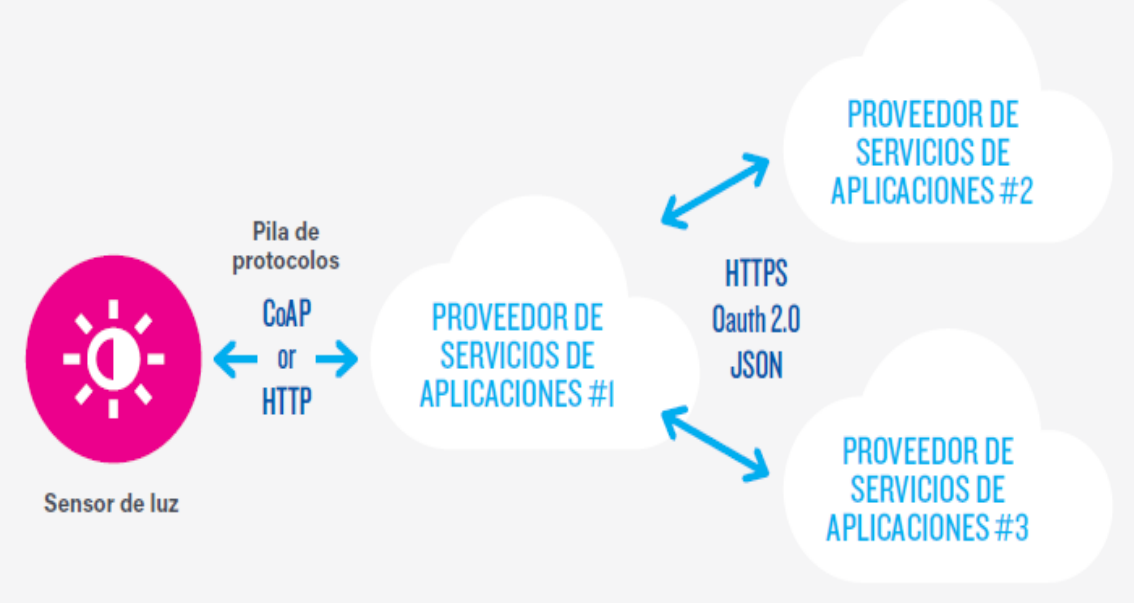


FIGURA 2. 6. MODELO DE INTERCAMBIO DE DATOS A TRAVÉS DE BACK-END

INTERNET SOCIETY. (2015). INTERNET SOCIETY. (2015). LA INTERNET DE LAS COSAS – UNA BREVE RESEÑA. INTERNET SOCIETY: HTTPS://WWW.INTERNETSOCIETY.ORG/WP-CONTENT/UPLOADS/2017/09/REPORT-INTERNETOFTHINGS-

2.5 Categorías en el Internet de las Cosas (IoT)

El Internet de las Cosas está siendo implementado en diferentes campos de nuestra vida diaria algunos de ellos son la industria, en la medicina, en las ciudades, en nuestros hogares, etc.

2.5.1 Internet industrial de las cosas (IIoT)

Este término fue introducido en 2012, pensado en conseguir una mayor eficacia en el funcionamiento de las máquinas, de tal forma se optimizarían los servicios y la producción de la industria.

En el Internet industrial de las cosas (*Industrial Internet of Things*, IIoT) las máquinas y sensores trabajan en una red donde hacen uso de la información obtenida, logrando así una reducción de costos en su producción, de igual forma en la selección de materias primas, etc.

CISCO define IoT como la conexión de personas, procesos, datos y cosas. En la industria es el fusionar de IT (Tecnología de la información, *Information Technology*) y OT (Tecnología de la Operación, *Operational Technology*), y de cómo los datos que se recolectan de las máquinas, las fábricas los analizan para ganar conocimiento, con ello optimizar y crear procesos.

IIoT es una extensión del IoT, pero enfocada a los entornos industriales y empresariales, la cual incluye dos componentes clave:

- La conexión de sensores y actuadores de máquinas industriales a internet.
- La futura conexión a otras redes industriales que puedan generar análisis de manera independiente.

Con ello el IIoT hace posible no solo la comunicación entre las máquinas y sus operadores, sino que también permite la comunicación externa entre suministro, servicio, diseño y logística. [15], [16], [17]

FIGURA 2. 7. INTERNET INDUSTRIAL DE LAS COSAS (IIOT).

2.5.2 Internet de las cosas médicas (IoMT)

El internet de las cosas médicas (IoMT**:** *Internet of Medical Things*), también llamado internet de la salud es una extensión del IoT, enfocado para fines médicos y de salud. Con ella se busca tener una recopilación y análisis de los datos más eficiente. El IoMT busca conectar los recursos médicos disponibles con los servicios de salud. Dichos dispositivos se utilizan para habilitar el monitoreo remoto de la salud y los sistemas de notificación de emergencia, tales como monitores de presión arterial y frecuencia cardiaca, dispositivos capaces de monitorizar implantes especializados, etc. Con estos dispositivos se recopilan y envían los datos obtenidos de los pacientes, a través de redes a los especialistas en la salud. [16], [18]

FIGURA 2. 8. INTERNET DE LAS COSAS MÉDICAS (IOMT).

2.5.3 Ciudades inteligentes

Las ciudades inteligentes son aquellas que tienen un desarrollo y mejora en la sostenibilidad y eficacia energética, movilidad y transporte, etc. Dichas ciudades buscan incrementar la calidad de vida de sus ciudadanos a través de las tecnologías inteligentes. [15]

La Unión Internacional de Telecomunicaciones define a una ciudad inteligente y sostenible como: "*Una ciudad innovadora que aprovecha las Tecnologías de la Información y la Comunicación (TIC) y otros medios para mejorar la calidad de vida, la eficiencia del funcionamiento, los servicios urbanos y la competitividad, al tiempo que se asegura que responde a las necesidades de las generaciones presentes y futuras en lo que respecta a los aspectos económicos, sociales, medioambientales y culturales".* [16]

Es por esto que las ciudades inteligentes utilizan la conectividad de los sensores distribuidos en el medio ambiente, para así dar solución a problemas, de tal forma garantizando una ciudad sostenible que goce de buenos servicios e informando siempre a sus ciudadanos.

Los sensores son los que proporcionan la información, asegurando así el correcto funcionamiento de aspectos sociales y ambientales.

FIGURA 2. 9. CIUDADES INTELIGENTES.

2.5.4 Hogares inteligentes

Los dispositivos de IoT que son utilizados en hogares y edificios inteligentes son agrupados habitualmente en la categoría de Hogar inteligente. Estos dispositivos tienen beneficios a largo plazo, tales como ahorros de energía al tener un encendido / apagado automático de luces, de igual forma en aparatos electrodomésticos inteligentes.

Estos dispositivos IoT se pueden conectar a una red y controlar de forma remota a través del internet mediante un dispositivo móvil o una computadora. [16], [18]

FIGURA 2. 10. HOGARES INTELIGENTES.

2.6 Tipología de conexiones entre elementos IoT

Los tres tipos de conexiones principales que existen entre ellos son:

- P2P persona a persona (*people to people*). El origen y destino de la comunicación se realiza de persona a persona.
- M2P máquina a persona (*machine to people*). El origen de la comunicación es una máquina y el destino es una persona.
- M2M máquina a máquina (*machine to machine*). El origen y el destino de la comunicación es una máquina. [19]

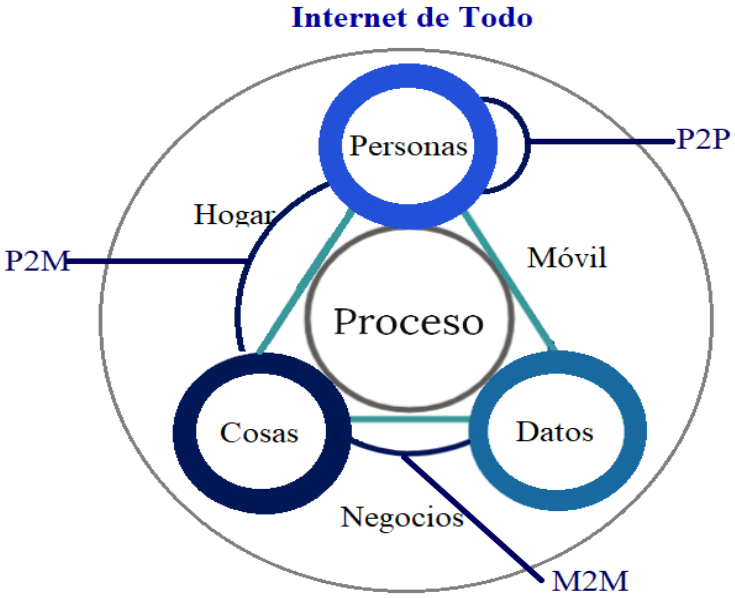


FIGURA 2. 11. TIPOS DE CONEXIONES ENTRE ELEMENTOS DE IOT.

SEUMA, M. L. I (2019). INTERNET DE LAS COSAS. RAMA EDITORIAL.

2.7 Ventajas y desventajas de IoT

Ventajas	Desventajas
Introducción de nuevos productos y servicios y hacer que muchas ofertas existentes queden completamente obsoletas.	En el futuro se pueden introducir nuevos tipos de delitos, armas y guerras.
La tecnología eliminará puestos de trabajo, pero introducirá nuevas líneas de trabajo.	También crear importantes problemas políticos y sociales.
Los sistemas conectados se extraerán a través de la educación, el gobierno y las empresas.	Fundamentalmente reasignar y reconfigurar las acciones, el comportamiento y las normas sociales.
La tecnología afectará todo, desde la forma en que la gente vota hasta la forma en que comemos en los restaurantes y tomamos vacaciones.	Producirá que la sociedad examine más de cerca la noción de privacidad y seguridad.

TABLA 2. 2. VENTAJAS Y DESVENTAJAS DE IOT

Referencias de Internet de las cosas (IoT).

[1] Congressional Research Service. (2015). The Internet of Things: Frequently Asked Questions. Congressional Research Service: https://crsreports.congress.gov

[2] Novillo Johnny, Hernández Dixys, Mazón Bertha, Molina Jimmy, Cárdenas Oscar. (2018). Arduino y el Internet de las cosas. 3ciencias.

[3] Unión Internacional de Telecomunicaciones (2023). Internet de los Objetos. https://www.itu.int/itunews/manager/display.asp?lang=es&year=2005&issue=09&ipage=things&ext=html.

[4] Villacis, Francisco. (2020). Internet de las Cosas: un mundo inteligente. https://www.ieee.org.

[5] Evans, Dave. (2011). Internet de las cosas cómo la próxima evolución de Internet lo cambiará todo. Cisco IBSG.

[6] Rose Karen, Eldridge Scott. & Chapin Lyman. (2015). La Internet de las cosas- una breve reseña: Para entender mejor los problemas y desafíos de un mundo más conectado. Internet Society. Retrieved February 16, 2022, de https://www.internetsociety.org/wp-content/uploads/2017/09/report-InternetOfThings-20160817-es-1.pdf.

[7] Norris, Donald. (2015). The Internet of Things: Do-It-Yoursel Projects with Arduino, Raspberry Pi, and BeagleBone Black. New York. McGranw-Hill Education.

[8] Instituto Federal de Telecomunicaciones. EL IFT PRESENTA EL CATÁLOGO DE DISPOSITIVOS DE INTERNET DE LAS COSAS (IoT). 21 de diciembre de 2022. https://www.ift.org.mx/sites/default/files/comunicacion-y-medios/comunicados-ift/comunicado116ift1.pdf.

[9] Unión Internacional de Telecomunicaciones (UIT). Iniciativa Mundial de Normalización sobre Internet de las cosas. T0B0400003C2C01PDFS.pdf (itu.int).

[10] ISO. Norma ISO/IEC 30141 sobre Internet de las Cosas (IoT). https://www.iso.org.

[11] Salazar, Jordi., Silvestre, Santiago. (2017). Internet de las cosas. TECHpedia.

[12] Ma, H. (2011). Internet of Things: Objectives and Scientific Challenges. Journal of Computer Science and Technology, 26(6), 919-924. https://doi.org/10.1007/s11390-011-1189-5

[13] Hosmer, Chet. (2018). Defending IoT infrastructures with the Raspberry Pi: Monitoring and Detecting Nefarious Behavior in Real Time. Longs, South Carolina, USA: Apress.

[14] Tsuchofening Hannes, Arkko Jari, Thaler Dave y McPherson Danny. (2015). RFC 7452: “Architectural Considerations in Smart Object Networking”.

[15] Aguilar, Luis. (2021). Internet de las cosas: Un futuro hiperconectado: 5G, inteligencia artificial, Big Data, Cloud, Blockchain y ciberseguridad. México: Marcombo.

[16] Congressional Research Service. (2020). The Internet of Things (IoT): An Overview. Congressional Research Service: https://crsreports.congress.gov

[17] Congressional Research Service. (2019). Technological Convergence: Regulatory, DigitalPrivacy, and Data Security Issues. Congressional Research Service: https://crsreports.congress.gov

[18] Teigens Vasil, Skalfist Peter y Mikelsten, Daniel. (2020). Inteligencia artificial: la cuarta revolución industrial. Cambridge Stanford Books.

[19] Seuba, Manuel. (2019). Internet de las Cosas: La transformación digital de la sociedad. Madrid: Ra-Ma Editorial.

Capítulo 3

Prácticas

Práctica 1. Puertos de Entrada/Salida

Introducción

Los puertos de entrada y salida son parte fundamental para la comunicación entre el usuario y el microcontrolador, entre el microcontrolador y el mundo exterior, ya sea para recabar datos o para transmitirlos.

Hay diferentes tipos de puertos de entrada y salida, pero todos tienen características en común.

En esta práctica analizaremos los puertos de Entrada/Salida de PIC16F887, Arduino UNO y Raspberry Pi 4 modelo B, comparándolos para identificar cual es el conveniente para IoT.

Objetivo

Identificar el número de bits en los puertos de entrada y salida de cada tecnología.

Identificar beneficios que ofrece el puerto de entrada y salida de cada tecnología.

3.1 Práctica 1. Puertos de Entrada/Salida en PIC 16F887

Equipo y material empleado

- PIC16F887
- Tarjeta de programación EasyPIC V6
- Software MikroC PRO for PIC.
- Pantalla LCD.
- Pantalla LCD incorporada.

3.1.1 Puertos de Entrada y Salida de PIC 16F887

El microcontrolador PIC16F887 cuenta con treinta y cinco pines de E/S de propósito general disponibles, estos pines se agrupan en cinco puertos llamados puerto A, puerto B, puerto C, puerto D y puerto E. Todos ellos cuentan con las siguientes características en común:

- La mayoría de los pines de E/S son multifuncionales, esto quiere decir que pueden ser utilizados como pin de E/S de propósito general.
- Cada puerto tiene su propio registro de control de flujo llamado registro TRIS, el cual determina el comportamiento de bits del puerto. Cuando TRIS=0 se configura al pin como una salida, al contrario, al poner TRIS=1 se configura al pin como una entrada.

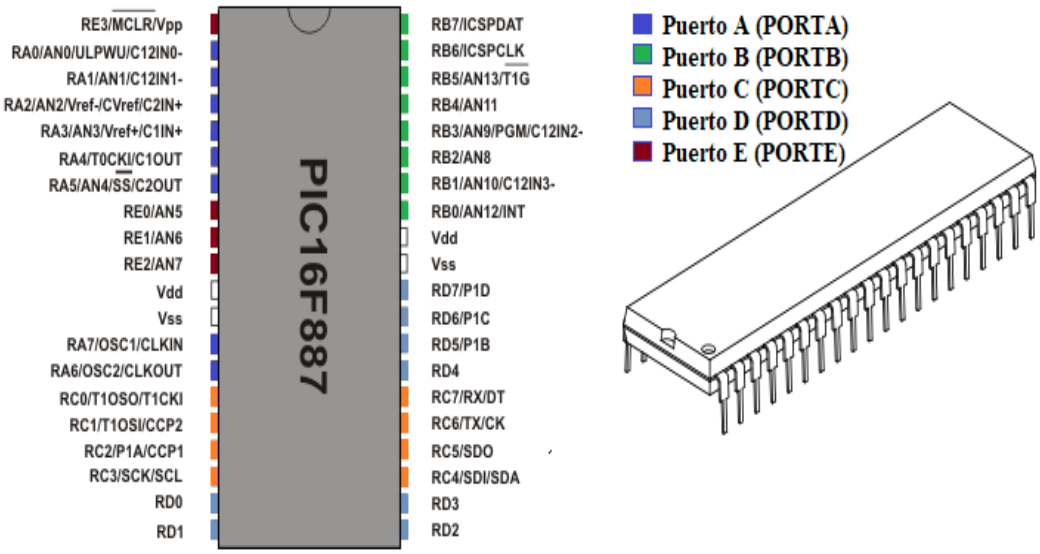


FIGURA 3. 1. PINOUT DE PIC 16F887.

3.1.1.1 Puerto A (PORTA) y registro (TRISA)

El PORTA es un puerto bidireccional de 8 bits de longitud. Cuenta con dos registros de dirección de datos llamados TRISA y ANSEL, estos dos registros controlan los pines del PORTA, mientras que en TRISA se establece si el pin es de entrada=1 o salida=0, en ANSEL se configura si el pin es entrada analógica=1 o entrada/salida digital=0.}

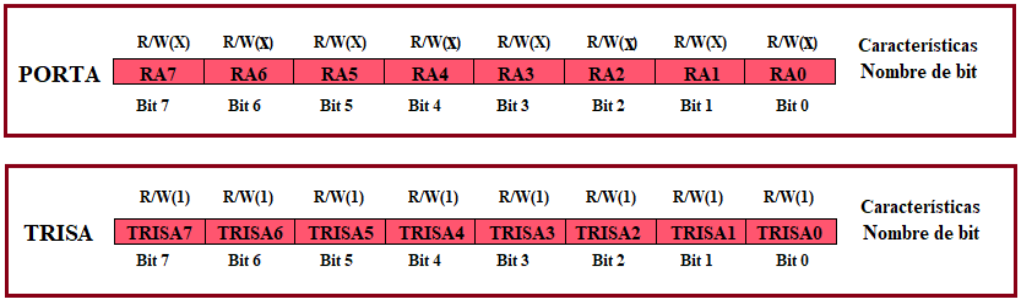


FIGURA 3. 2. PUERTO A.

En PORTA el bit 0 (RA0) cuenta con una función llamada *Ultra Low-Power Wake-up* (ULPWU), la cual permite que un voltaje de caída lenta genere una interrupción ocasionando un cambio en RA0 respecto al consumo de corriente. Al generar la interrupción el dispositivo se despierta y ejecuta la siguiente instrucción.

PIN	RA0	RA1	RA2	RA3	RA4	RA5	RA 6	RA7
E/S de propósito general	X	X	X	X	X	X	X	X
Entrada analógica para el ADC.	X	X	X	X		X		
Entrada analógica negativa al comparador C1 y C2	X	X						
Entrada de selección de para esclavo.						X		
Entrada de referencia de voltaje negativo para ADC.			X					
Entrada de referencia de voltaje positivo para ADC.				X				
Salida de referencia del voltaje del comparador.			X					
Entrada analógica positiva al comparador C2			X					
Entrada analógica positiva al comparador C1.				X				
Entrada de reloj para Timer0.					X			
Salida digital del comparador C1.					X			
Salida digital del comparador C2.						X		
Conexión de cristal/resonante							X	X
Salida de reloj.							X	
Entrada de reloj								X

TABLA 3. 1. CARACTERÍSTICAS DEL PUERTO A.

3.1.1 2 Puerto B (PORTB) y registro (TRISB)

El puerto B (PORTB) es un puerto bidireccional de 8 bits de longitud, al igual que el PORTA cuenta con un registro TRISB el cual determina la función de los pines.

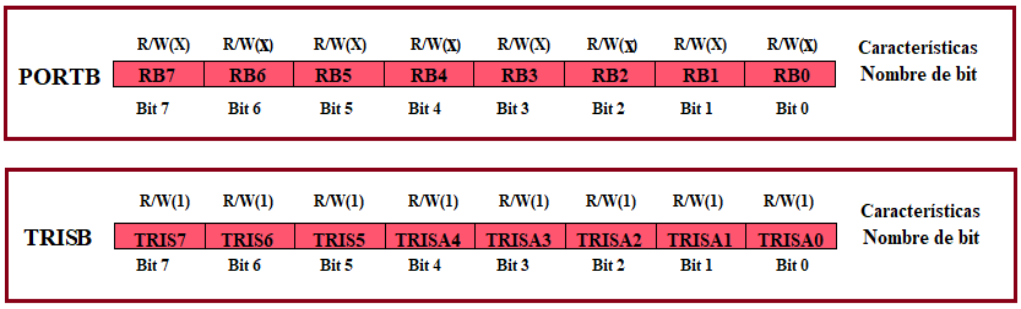


FIGURA 3. 3. PUERTO B.

Este puerto es utilizado con mayor frecuencia, ya que, todos sus pines tienen resistencias *pull-up* integradas, que los hacen perfectos para conectar teclados, interruptores y optoacopladores. Al tener un alto nivel de resistencia (resistencia virtual) no se afecta a los pines configurados como salidas, el cual funciona como un complemento útil a las entradas. Las resistencias *pull-up* se encuentran conectadas en las entradas de los circuitos CMOS.

Los pines del puerto B son utilizados para la transmisión de señal de reloj, y de datos al cargar un programa, además de que es necesario suministrar un voltaje de alimentación Vdd de 5[V] y de un voltaje Vpp de 12 [V]-14 [V] para la programación de la memoria Flash.

El PORTB al igual que el PORTA cuenta con funciones adicionales las cuales se muestran en la Tabla 3.2.

PIN	RB0	RB1	RB2	RB3	RB4	RB5	RB6	RB7
E/S de propósito general	X	X	X	X	X	X	X	X
Entrada analógica para el ADC	X	X	X	X	X	X		
Interrupción por flanco externo	X							
Salida PWM		X	X		X			
Habilitación de Bajo Voltaje para Programación Serial en Circuito				X				
Entrada analógica al comparador C1 o C2		X		X				
Entrada para Timer1						X		
Reloj de programación en serie en el circuito							X	
Datos de programación serie en el circuito								X

TABLA 3. 2. CARACTERÍSTICAS DEL PUERTO B.

3.1.1.3 Puerto C (PORTC) y registro (TRISC)

El PORTC es un puerto bidireccional de 8 bits de longitud, su registro de dirección de datos es TRISC, este registro controla los controladores de salida del pin PORTC.

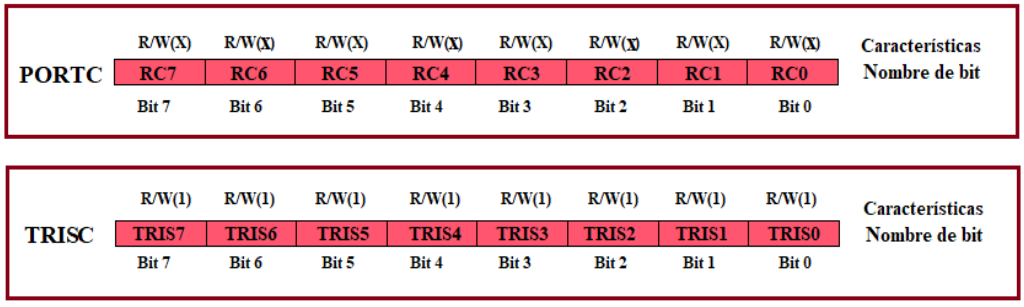


FIGURA 3. 4. PUERTO C.

PIN	RC0	RC1	RC2	RC3	RC4	RC5	RC6	RC7
E/S de propósito general	X	X	X	X	X	X	X	X
Salida del oscilador Timer1	X							
Entrada reloj Timer1	X							
Entrada del oscilador Timer1		X						
Entrada PWM para el comparador C2		X						
Salida PWM			X					
Entrada de captura y salida de comparación para comparador C1			X					
Reloj SPI				X				
Reloj I2C				X				
E/S de datos SPI					X			
E/S de datos I2C					X			
Salida serial de datos						X		
Salida serie asíncrona							X	
E/S de reloj síncrono							X	
Entrada serie asíncrona								X
E/S de datos serie síncronos								X

TABLA 3. 3. CARACTERÍSTICAS DEL PUERTO C.

3.1.1.4 Puerto D (PORTD) y registro (TRISD)

El puerto D (PORTD) es un puerto bidireccional de 8 bits de longitud, los bits del registro TRISD determinan la función de sus pines.

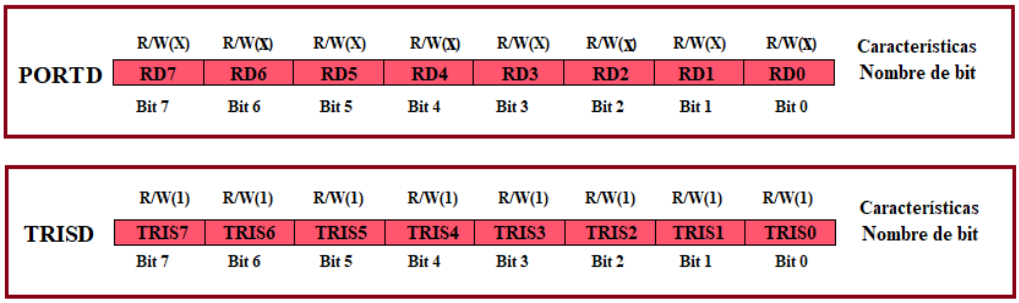


FIGURA 3. 5. PUERTO D.

PIN	RD0	RD1	RD2	RD3	RD4	RD5	RD6	RD7
E/S de propósito general	X	X	X	X	X	X	X	X
Salida PWM						X	X	X

TABLA 3. 4. CARACTERÍSTICAS DEL PUERTO D.

3.1.1.5 Puerto E (PORTE) y registro (TRISE)

El puerto E (PORTE) es un puerto bidireccional de 4 bits de longitud. El registro de dirección de datos es TRISE el cual determina la función de sus pines. El pin RE3 siempre será configurado como una entrada, ya que tiene una función predeterminada como entrada analógica y su registro de configuración se establece en este modo al encender el microcontrolador.

Por lo cual el pin RE3 se configura como una entrada analógica, pero esta configuración se puede cambiar mediante la programación del microcontrolador.

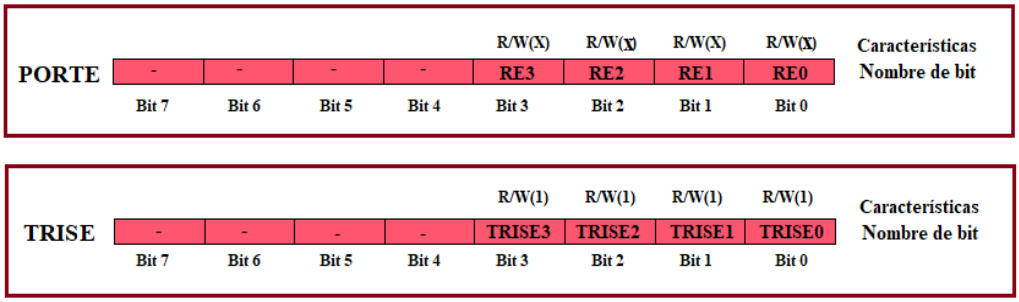


FIGURA 3. 6. PUERTO E.

PIN	RE0	RE1	RE2	RE3
E/S Propósito General	X	X	X	X
Entrada analógica para ADC	X	X	X	
Reset maestro con pull-up				X

TABLA 3. 5. CARACTERÍSTICAS DEL PUERTO E.

La máxima capacidad de corriente de cada uno de los pines de los puertos en modo sumidero (*sink*) o en modo fuente (*source*) es de 25 [mA], de igual modo su consumo depende del puerto en el cual trabaje.

Modo	Puerto A	Puerto B	Puerto C	Puerto D
Sumidero	150 [mA]	200 [mA]	200 [mA]	200 [mA]
Fuente	150 [mA]	200 [mA]	200 [mA]	200 [mA]

TABLA 3. 6. CAPACIDAD DE CORRIENTE EN CADA PUERTO EN MODO SUMIDERO O FUENTE.

El consumo de corriente del microcontrolador depende del voltaje de operación, la frecuencia y de los elementos que sean conectados en los pines.

PIC 16F887

La corriente máxima de los puertos del PIC16F887 soporta 25 [mA]

Para poder programar el PIC se hará uso del Programa *mikroC PRO for PIC* el cual se inicia mediante el icono de la figura 3.7.

FIGURA 3. 7. ICONO DEL PROGRAMA MIKROC PRO FOR PIC.

Estando dentro del programa se nos mostrara una ventana donde podremos crear el proyecto nuevo, en este elegiremos el PIC 16F887, de igual forma la frecuencia a la cual trabajaremos.

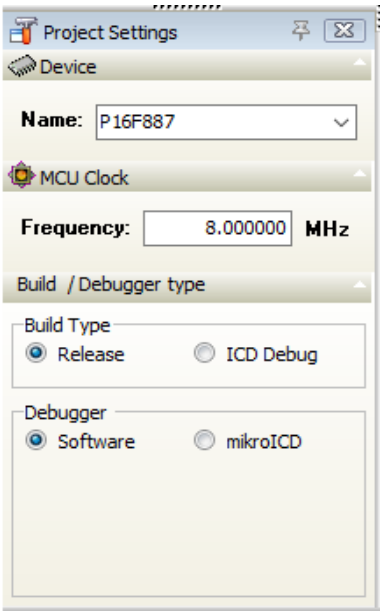


FIGURA 3. 8. SELECCIÓN DEL PIC DENTRO DEL PROGRAMA MIKROC.

En la tarjeta de desarrollo EasyPIC V6 figura 3.9. es donde insertaremos el PIC 16F887 y nos permitirá realizar prácticas de una forma más sencilla.

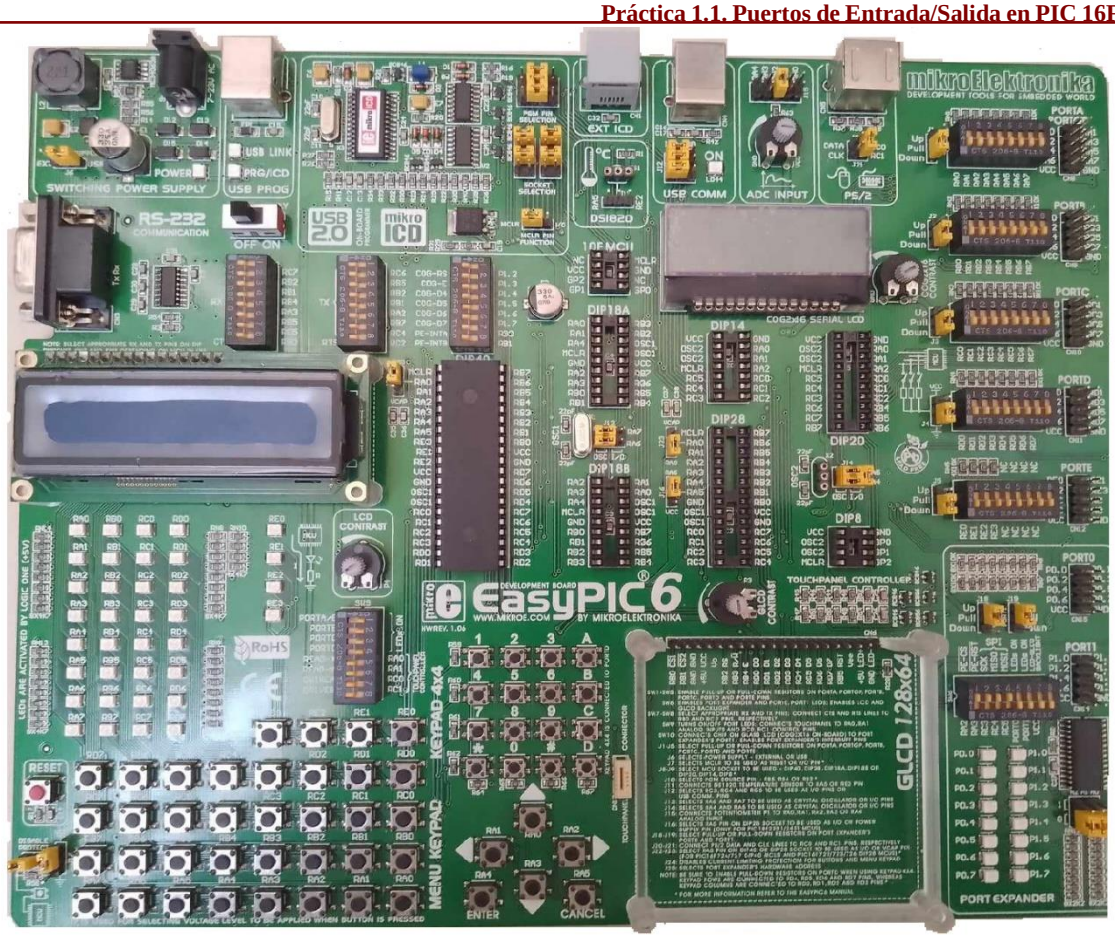


FIGURA 3. 9. TARJETA DE DESARROLLO EASYPIC V6.

Para encender el LED del puerto A escribimos el programa en mikroC y lo cargamos en el PIC que se encontrará montado en el zócalo de 40 pines de la tarjeta de desarrollo EasyPIC. Como se muestra en la figura 3.11 podemos ver el LED RA0 encendido.

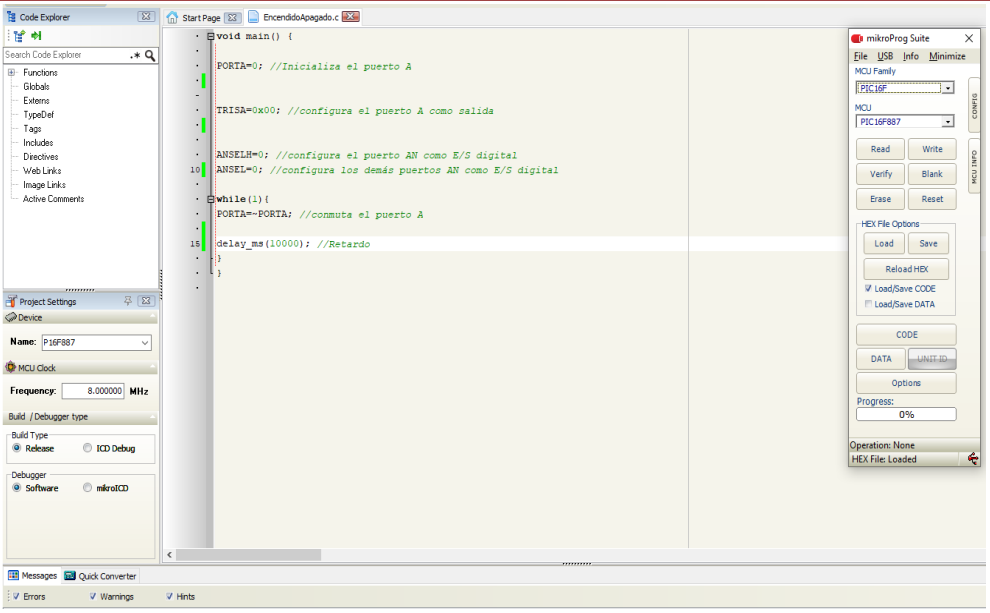


FIGURA 3. 10. PROGRAMA PARA ENCENDER EL MÓDULO A DE LA EASYPIC.

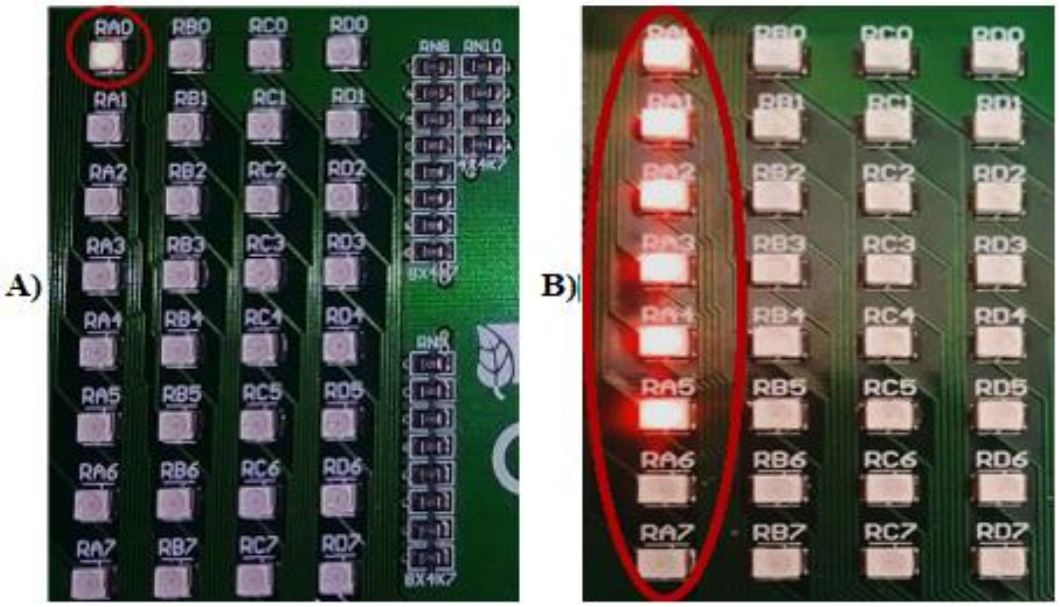


FIGURA 3. 11. A) MÓDULO DE PUERTOS DE LEDS DE LA PLACA EASYPIC V6

B) ENCENDIDO DE LA COLUMNA A DE TARJETA DE DESARROLLO.

De igual forma podemos ocupar todas las columnas y filas del bloque de LED de la EasyPIC, realizando los cambios en el programa, para lograr encender todos los LED´s en la tarjeta de desarrollo.

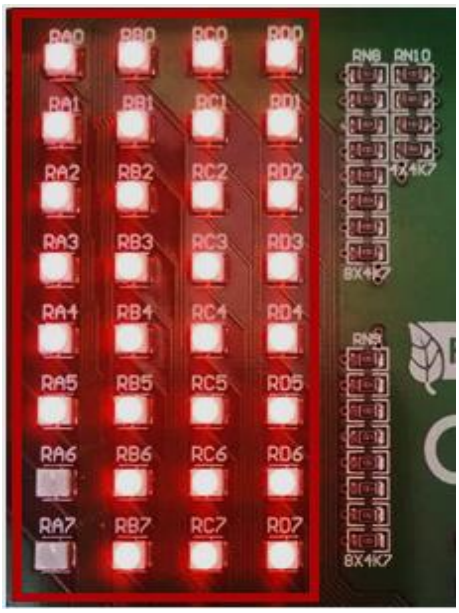
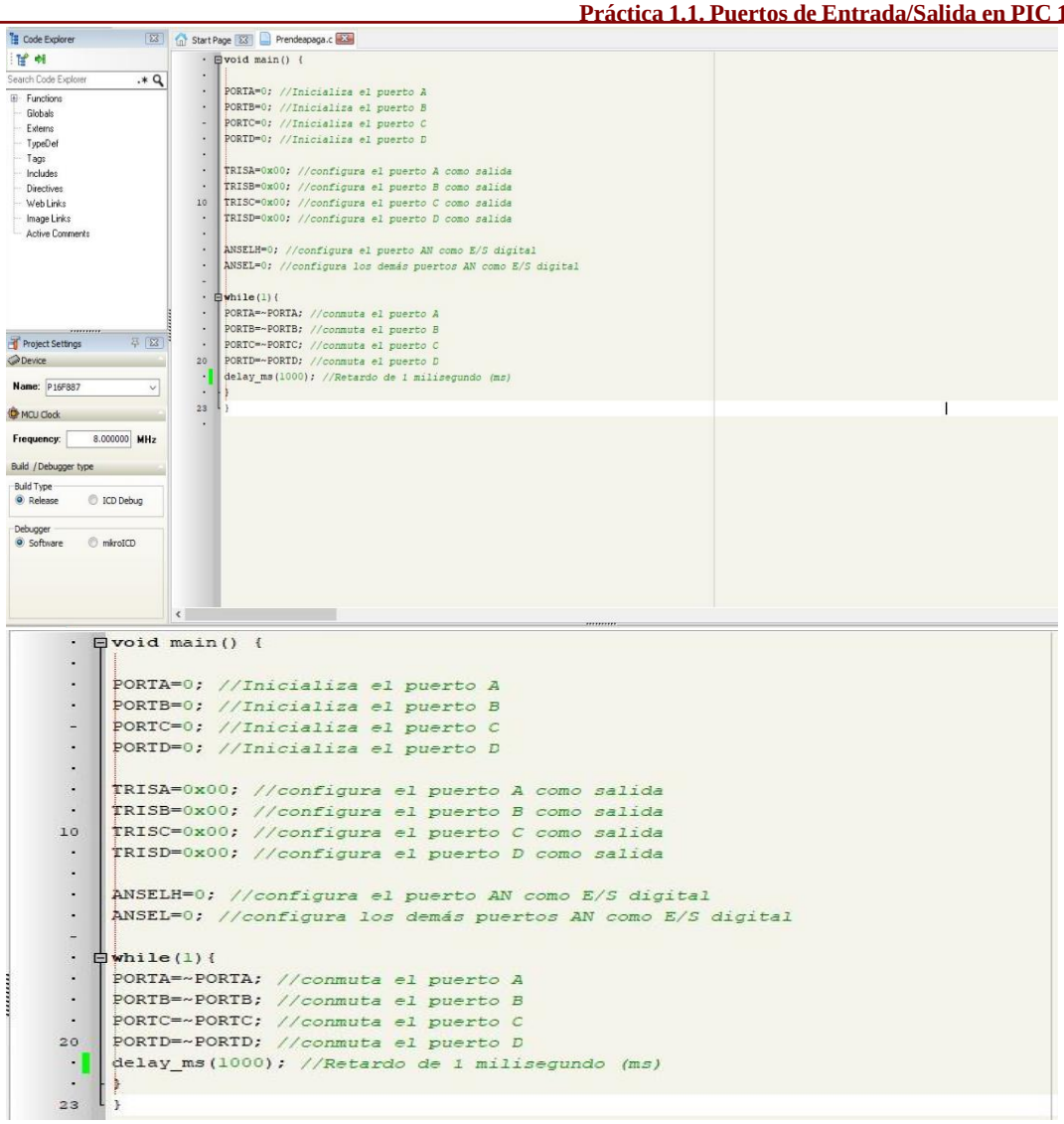


FIGURA 3. 12. MÓDULO DE PUERTOS LED ENCENDIDO




```
void main() {

PORTA=0; //Inicializa el puerto A
PORTB=0; //Inicializa el puerto B
PORTC=0; //Inicializa el puerto C
PORTD=0; //Inicializa el puerto D

TRISA=0x00; //configura el puerto A como salida
TRISB=0x00; //configura el puerto B como salida
TRISC=0x00; //configura el puerto C como salida
TRISD=0x00; //configura el puerto D como salida

ANSELH=0; //configura el puerto AN como E/S digital
ANSEL=0; //configura los demás puertos AN como E/S digital

while(1){
PORTA=~PORTA; //conmuta el puerto A
PORTB=~PORTB; //conmuta el puerto B
PORTC=~PORTC; //conmuta el puerto C
PORTD=~PORTD; //conmuta el puerto D
delay_ms(1000); //Retardo de 1 milisegundo (ms)
}
}
```

FIGURA 3. 13. PROGRAMA PARA ACTIVAR TODO EL MÓDULO DE LEDS DE LA EASYPIC.

La EasyPIC V6 cuenta con tres pantallas LCD, para poder utilizarlas es necesario configurar la tarjeta de desarrollo y activar las librerías indicadas para cada pantalla.

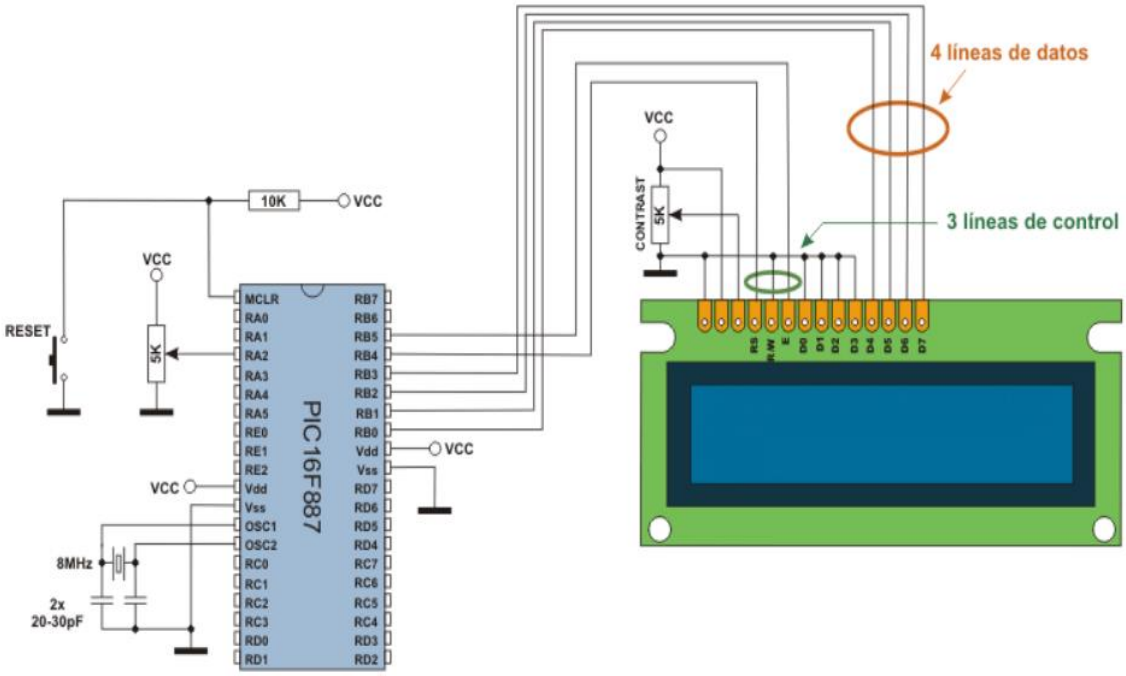


FIGURA 3. 14. DIAGRAMA DE CONEXIÓN PARA LA PANTALLA LCD 2X16.

Antes de insertar la pantalla LCD de 2x16 en la tarjeta de desarrollo configuramos la misma, primero encendemos el interruptor de luz de la pantalla que se encuentra en la parte inferior derecha marcado con el nombre SW6; en este switch encendemos el botón 8, colocamos la pantalla en la tarjeta de desarrollo, la cual nos permite variar el contraste de la pantalla mediante el potenciómetro de 10 [kΩ] ubicado debajo de la pantalla.

FIGURA 3. 15. POTENCIÓMETRO PARA VARIAR EL CONTRASTE DE LA PANTALLA LCD, MODULO PARA INSERTAR LA PANTALLA LCD.

Las librerías que deben estar activadas para que el programa y la tarjeta de desarrollo funcionen son:

- ADC
- LCD

Para que la pantalla integrada de 2x16 funcione adecuadamente es necesario configurar la tarjeta de desarrollo, para ello es necesario poner en posición ON el switch SW10 el cual podemos observar encerrado en el recuadro rojo, la LCD de 2x16 tiene un potenciómetro para ajustar el contraste, esta LCD integrada no cuenta con luz de fondo, pero recibe los datos por medio del expansor de puertos que utilizan la comunicación SPI.

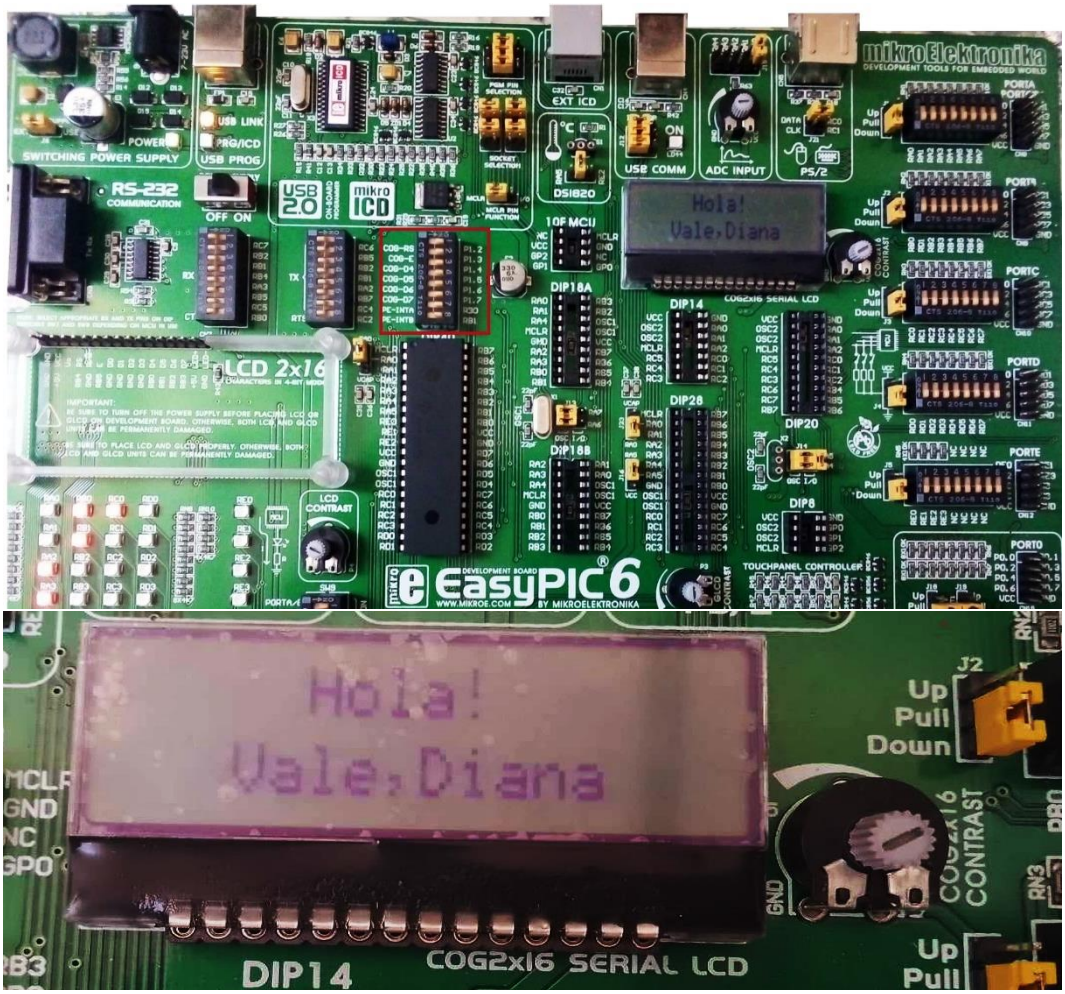
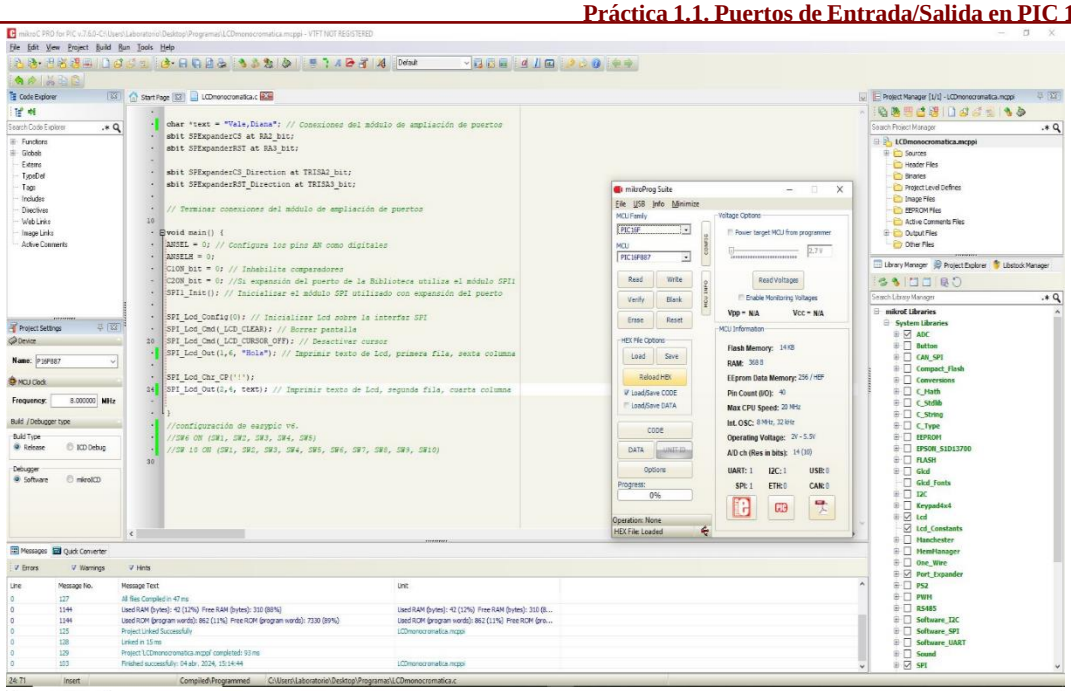


FIGURA 3. 16. PANTALLA INTEGRADA DE 2X16.



```
char *text = "Vale,Diana"; // Conexiones del módulo de ampliación de puertos
sbit SPExpanderCS at RA2_bit;
sbit SPExpanderRST at RA3_bit;

sbit SPExpanderCS_Direction at TRISA2_bit;
sbit SPExpanderRST_Direction at TRISA3_bit;

// Terminar conexiones del módulo de ampliación de puertos

void main() {
ANSEL = 0; // Configura los pins AN como digitales
ANSELH = 0;
C1ON_bit = 0; // Inhabilita comparadores
C2ON_bit = 0; //Si expansión del puerto de la Biblioteca utiliza el módulo SPI1
SPI1_Init(); // Inicializar el módulo SPI utilizado con expansión del puerto

SPI_Lcd_Config(0); // Inicializar Lcd sobre la interfaz SPI
SPI_Lcd_Cmd(_LCD_CLEAR); // Borrar pantalla
SPI_Lcd_Cmd(_LCD_CURSOR_OFF); // Desactivar cursor
SPI_Lcd_Out(1,6, "Hola"); // Imprimir texto de Lcd, primera fila, sexta columna

SPI_Lcd_Chr_CP('!');
SPI_Lcd_Out(2,4, text); // Imprimir texto de Lcd, segunda fila, cuarta columna

}
//configuración de easypic v6.
//SW6 ON (SW1, SW2, SW3, SW4, SW5)
//SW 10 ON (SW1, SW2, SW3, SW4, SW5, SW6, SW7, SW8, SW9, SW10)
```

FIGURA 3. 17. PROGRAMA DE LA PANTALLA INTEGRADA DE 2X16.

3.2 Practica 1.2. Puertos de Entrada/Salida en Arduino UNO

Equipo y material empleado

- Arduino UNO
- Protoboard
- LED
- Resistencias
- Jumper macho-macho

3.2.1 Puertos de Entrada y Salida de Arduino UNO

La placa Arduino Uno está equipado con el microcontrolador ATmega328P. La letra P al final del nombre se refiere a que se incorpora la tecnología *picopower*, esto quiere decir que su consumo eléctrico es menor en comparación a los modelos que cuentan con el microcontrolador ATmega328.

ATmega328P utiliza una arquitectura de tipo AVR, la cual es desarrollada por Atmel, la competencia de la arquitectura de PIC del fabricante Microchip.

La tecnología Arduino cuenta con registros llamados puerto B, puerto C y puerto D. El total de pines de Entrada / Salida es de 14 pines, estos pines suelen llamarse de propósito general GPIO ("*General Purpose Input / Output*") numerados del 0 hasta el 13, de los cuales 6 se pueden utilizar como salidas PWM, además de tener 6 pines de entradas analógicas. Como se puede observar en la figura 3.19 los 14 pines pueden utilizarse como pines analógicos o digitales, y algunos de estos pines sirven para la comunicación serial.

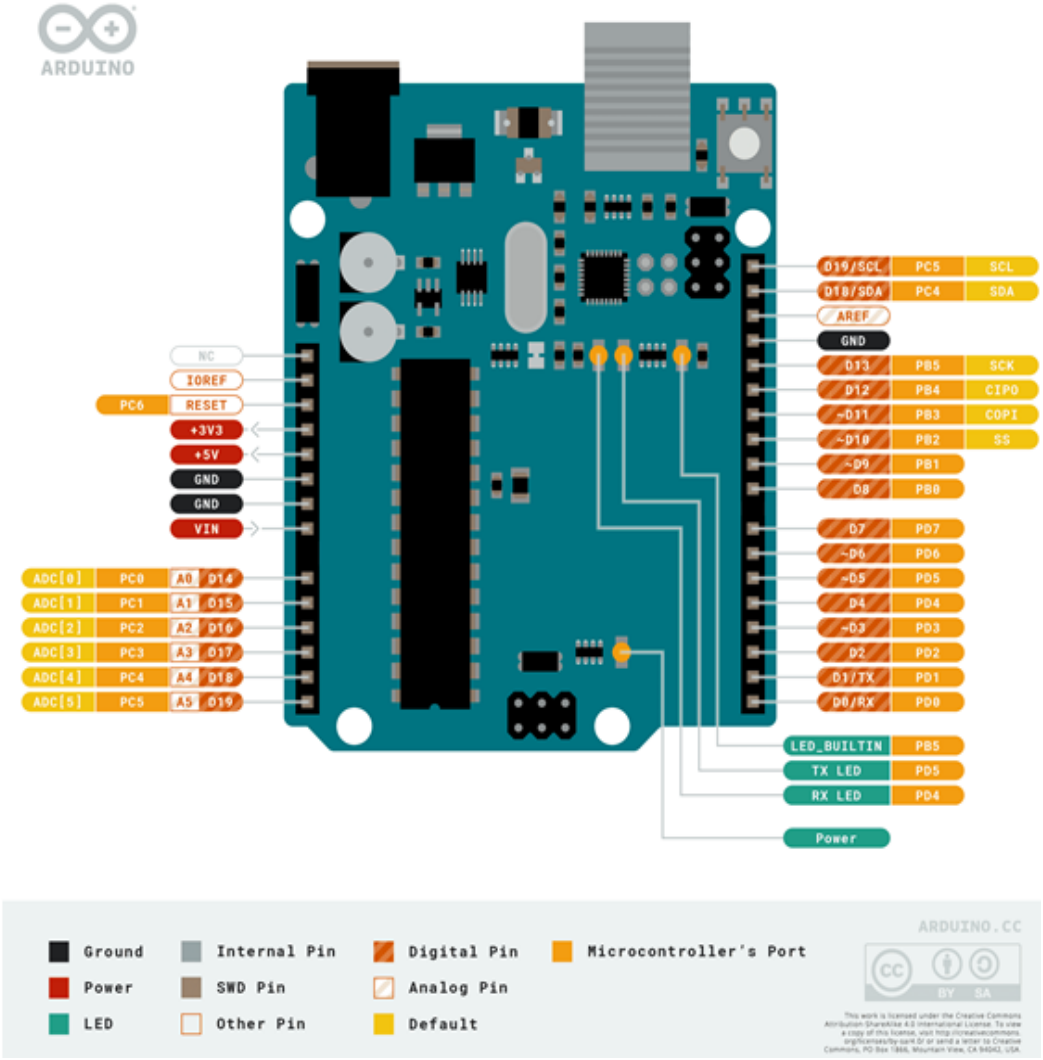


FIGURA 3. 18. PINOUT DE ARDUINO UNO.

UNO R3 | ARDUINO DOCUMENTATION

Estos pines digitales funcionan con 5 [V] y proporcionan o reciben un máximo de 40 [mA]. Arduino UNO tiene un límite de corriente total para todos los pines digitales combinados de alrededor de 200 [mA], pero no es seguro que cada pin suministre esa cantidad de corriente individualmente. Entonces, si se conectan varios elementos a los pines digitales, es necesario calcular la corriente total que consumirán todos estos elementos y asegurarse de que esté por debajo de los 200 [mA].

Arduino UNO tiene resistencias pull-up internas que pueden activarse en los pines digitales. Estas resistencias son de aproximadamente 20-50 [kΩ]. Cuando se activan las resistencias pull-up internas, proporcionan una conexión a VCC para mantener el estado alto del pin cuando no hay una conexión externa.

Los pines digitales tienen las siguientes características:

- Pin 0 (RX) y 1 (TX) son utilizados para recibir y para transmitir datos para la comunicación serial.
- Pin 2 y 3 se utilizan para administrar las interrupciones externas. Estos pines son los encargados de interrumpir el programa secuencial establecido por el programador.
- Pines para salida PWM de 8 bits a 490 [Hz]
  - ✓ Pin 3
  - ✓ Pin 5
  - ✓ Pin 6
  - ✓ Pin 9
  - ✓ Pin 10
  - ✓ Pin 11
- Pines para comunicación serie con el protocolo SPI.
  - ✓ Pin 10 (SS)
  - ✓ Pin 11 (MOSI)
  - ✓ Pin 12 (MISO)
  - ✓ Pin 13 (SCK)
- Pin 13 este permite detectar de forma rápida señales externas, cuando se recibe una señal el LED en la placa se encenderá indicando que está en HIGH de otra forma el LED permanecerá apagado LOW.
- Pines para comunicación serie con protocolo I2C.
  - ✓ Pin A4 SDA.
  - ✓ Pin A5 SCL.

El Arduino Uno tiene 6 entradas analógicas numeradas de A0- A5, cada uno de estos pines tiene 10 bits. Estos pines cuentan con funciones especiales las cuales son:

- Pin IOREF Este pin indica a los módulos que se encuentren conectados a la placa el voltaje en el que trabajan los pines de Arduino.
- Pin AREF (*Analog Reference*, Referencia Analógica) este pin ayuda a mejorar la precisión de las entradas analógicas mediante un voltaje de referencia externo.
- Pin RESET permite reiniciar el *bootloader*.

Además de esto la placa Arduino es alimentada por una conexión USB o con una alimentación externa a través de un conector jack. La placa soporta un mínimo de 6 [V] y un máximo de 20[V], pero es recomendable un voltaje de 7[V] a 12 [V] para evitar daños en la placa. La corriente máxima de los puertos del Arduino UNO soporta 40 [mA]

led Arduino 1.8.4

Archivo Editar Programa Herramientas Ayuda

led §

```
const int ledPIN = 9;

void setup() {
  //Serial.begin(9600);     //iniciar puerto serie
  pinMode(ledPIN , OUTPUT);  //definir pin como salida
}

void loop(){
  digitalWrite(ledPIN , HIGH);    // poner el Pin en HIGH
  delay(500);
  digitalWrite(ledPIN, LOW);
  delay(500);
}
```

FIGURA 3. 19. PROGRAMA PARA ENCENDER UN LED

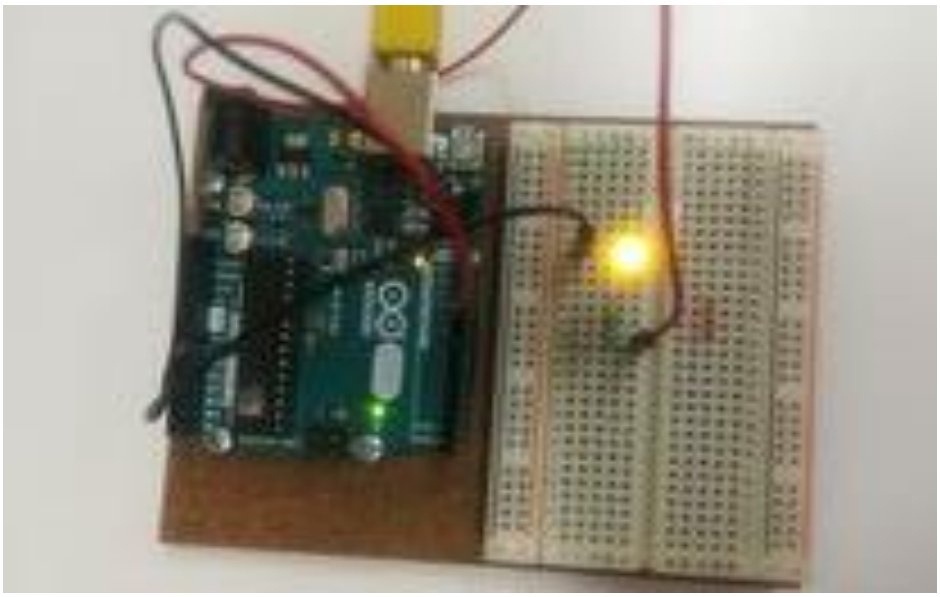


FIGURA 3. 20. LED EN ARDUINO.

3.3 Práctica 1.3. Puertos de Entrada/Salida en Raspberry Pi 4

Equipo y material empleado

- Raspberry Pi 4 modelo B
- Protoboard
- Resistencia
- Diodo LED
- Jumpers

3.3.1 Puertos de Entrada y Salida de Raspberry Pi 4

Cuando se agrupan un conjunto de pines GPIO se le llama puerto. Los pines GPIO son líneas de entrada y salida (E/S) digital de propósito general.

Los pines GPIO de la Raspberry Pi no son tolerantes a voltajes de 5V. Están pensados para su utilización con circuitos de 3.3V y no tienen ningún tipo de protección. No se debe demandar más de 16 [mA] por pin.

Características

- Pueden estar activados (ON) o desactivados (OFF).
- Pueden ser de entrada (lectura) o de salida (escritura).
- Son binarios: nivel alto (3.3 [V], HIGH) representa 1 lógico y nivel bajo (0 [V], representa un 0 lógico).

El puerto GPIO de la Raspberry Pi 4 está compuesto por 40 pines.

- 27 pines GPIO
- 1 puerto serie UART
- 2 buses SPI (SPI0 y SPI1)
- 1 bus I^2C
- 2 pines de alimentación de 5 [V]
- 2 pines de alimentación de 3.3 [V]
- 8 pines GND

Función	GPIO	PIN		GPIO	Función
3V3		1	2	-	5 [V]
I2C SDA	GPIO 2	3	4	-	5 [V]
I2C SCL	GPIO 3	5	6	-	GND
GPCLK0	GPIO 4	7	8	GPIO 14	UART (TX)
GND	-	9	10	GPIO 15	UART (RX)
-	GPIO 17	11	12	GPIO 18	PCM_CLK
-	GPIO 27	13	14	-	GND
-	GPIO 22	15	16	GPIO 23	-
3V3	-	17	18	GPIO 24	-
MOSI	GPIO 10	19	20	-	GND
MISO	GPIO 9	21	22	GPIO 25	-
SCLK	GPIO 11	23	24	GPIO 8	CE0
GND	-	25	26	GPIO 7	CE1
ID_SD	GPIO 0	27	28	GPIO 1	ID_SC
-	GPIO 5	29	30	-	GND
-	GPIO 6	31	32	GPIO 12	PWM0
PWM1	GPIO 13	33	34	-	GND
PCM_FS	GPIO 19	35	36	GPIO 16	-
-	GPIO 26	37	38	GPIO 20	PCM_DIN
GND	-	39	40	GPIO 21	PCM_DOUT

TABLA 3. 7. PINOUT DE RASPBERRY PI 4, EL PIN 1 ES EL MÁS CERCANO A LA TARJETA MICROSD.

- Los pines 3 y 5 se pueden configurar como interfaz I^2C.
- Los pines 8 y 10 pueden configurarse como interfaz UART.
- El pin 12 puede configurarse como salida PWM.
- Los pines 19, 21, 23, 24 y 26 se pueden configurar como la primera interfaz SPI.
- Los pines 27 y 28 no están disponibles. Están reservados para la incorporación adicional de una memoria serie en las placas de expansión conforme a la especificación HAT. Son los únicos pines que en el arranque se configuran como salidas.
- Los pines 29, 31, 32, 33, 35, 36, 37, 38 y 40 proporcionan acceso a nuevos pines de GPIO. Además de que los pines 32, 33 y 35 pueden utilizarse para salidas PWM.

Imagen	Puerto	Función
	Puerto USB-C	Alimentación 5 [V] y 3 [A]
	2 x Micro HDMI	Transmite audio y video de alta calidad
	2 x USB 2.0	Conector compatible con USB segunda versión.
	2 x USB 3.0	Conector compatible con USB tercera versión.
	CSI	Puerto utilizado para conectar la cámara oficial de Raspberry Pi
	DSI	Puerto empleado para conectar pantalla táctil en Raspberry Pi.
	Jack	Puerto de 3.5 mm para conectar auriculares / vídeo compuesto
	Micro SD	Puerto para tarjeta Micro-SD en la cual se cargar el sistema operativo y se realiza el almacenamiento de datos.
	Ethernet	Puerto para conectar la placa Raspberry Pi mediante Ethernet.

TABLA 3. 8. PUERTOS EN RASPBERRY PI 4 MODELO B.

La corriente máxima de los puertos de Raspberry Pi 4 soportan 50 [mA]

Voltaje mínimo de 6 [V]

Voltaje máximo de 20 [V]

Voltaje recomendado de 7[V] - 12[V]

Para utilizar un LED en uno de estos pines, se necesita una resistencia de 1 [kΩ] en serie con el LED, con uno de los pines GPIO que se elija.

Para poder programar un pin GPIO, es necesario instalar la librería llamada **Rpi.GPIO**. Lo que nos permitirá controlar el puerto GPIO.

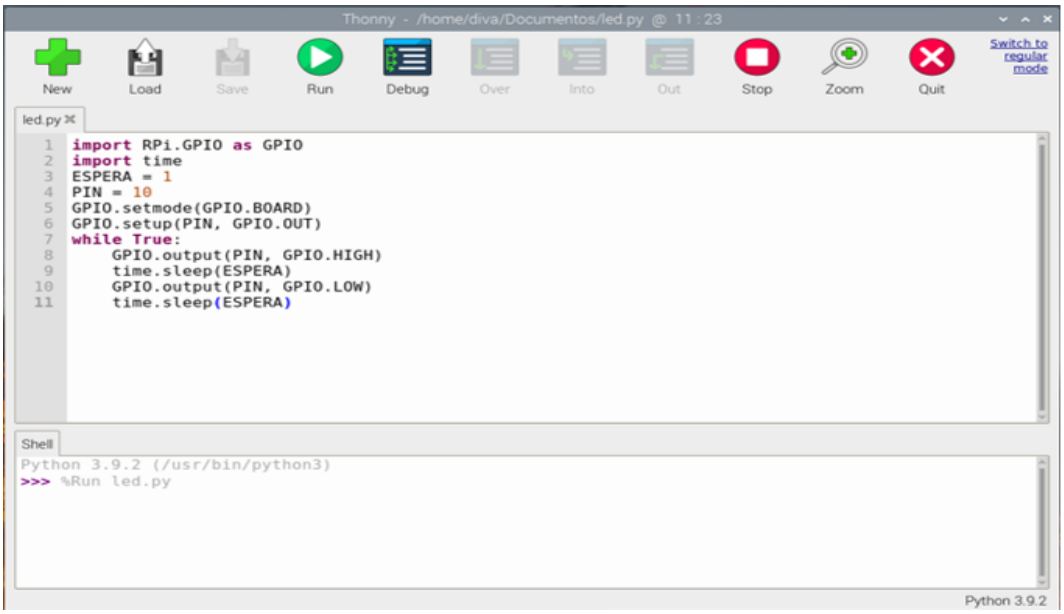


FIGURA 3. 21. PROGRAMA EN THONY.

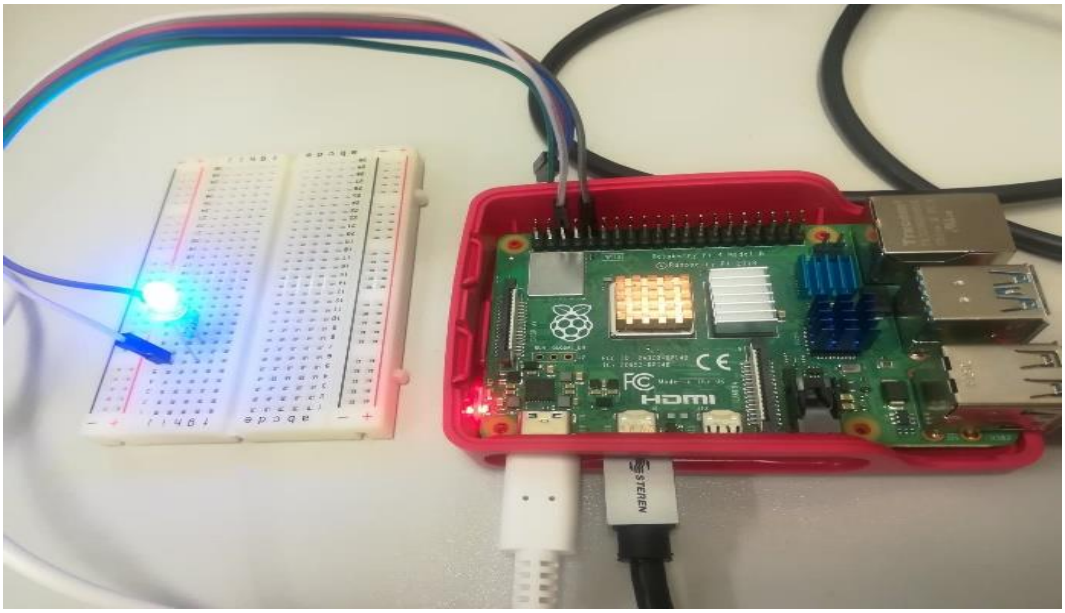


FIGURA 3. 22. LED EN RASPBERRY PI 4.

Referencias de la Práctica 1. Puertos de Entrada/Salida

[1] Verle, M. (2009). PIC Microcontrollers: Programming in C. Mikroelektronika.

[2] Benchimol, Daniel. Microcontroladores: Funcionamiento, Programación y Aplicaciones Prácticas, USERS - pág. 124

Arduino

[3] Aldea, E. L. (2016). Arduino. Guía práctica de fundamentos y simulación: Guía práctica de fundamentos y simulación. Ra-Ma Editorial.

[4] Torrente, Ó. (2013). Arduino: Curso práctico de formación. Alpha Editorial.

[5] Calaza, G. T. (2014). Taller de Arduino: un enfoque práctico para principiantes.

Raspberry Pi

[6] López, Eugenio. (2017). Raspberry Pi. Fundamentos y aplicaciones. Madrid: RA-MA Editorial.

Práctica 2. Convertidor Digital a Analógico (DAC)

Introducción

Las tecnologías para el IoT son digitales, es decir, está codificado en valores de 0 y 1.

En el mundo real puede ser digital, podemos recibir información sobre algo que solo acepta valores absolutos, sí o no. Por ejemplo, la puerta está abierta o cerrada. Pero realmente se presentan muchos más valores analógicos que pueden variar de modo continuo.

En esta práctica analizaremos y compararemos el puerto DAC en cada una de las tecnologías, específicamente en PIC16F887, Arduino UNO y Raspberry Pi 4 modelo B.

La tecnología Raspberry Pi no cuenta con un conversor DAC integrado, por lo que solo trabaja con entradas y salidas digitales, debido a esto se trabajará con un módulo externo para poder obtener una señal analógica.

Objetivo:

Identificar y comparar velocidad de conversión, resolución**,** rangos de entrada y salida.

3.4 Práctica 2.1. DAC PIC16F887

Equipo y material empleado

- PIC 16F887
- Protoboard
- Jumper macho-macho
- Diodo LED
- Resistencias

3.4.1 DAC en PIC 16F887

La mayoría de los PIC no cuentan con ningún módulo convertidor de señal digital a analógica (DAC, *Digital to Analog Converter*), por lo cual es necesario utilizar módulos externos para realizar dicha conversión, otra de las formas es utilizando PWM (*Pulse Width Modulation*, Modulación de ancho de Pulso), éste lo podemos encontrar en el PIC en el Módulo CCP (Comparador, Captura y PWM).

3.4.2 Módulo CCP

El módulo CCP es el que se encarga de realizar tres funciones básicas basadas en el manejo de los *Timer* las cuales son:

- Comparador: compara el valor del temporizador con el valor de un registro.
- Captura: obtiene el valor del *Timer* en un tiempo establecido.
- PWM: genera una señal modulada en amplitud de pulso.

El PIC 16F887 utilizado, es de gama media el cual tiene hasta 2 módulos CCP, tales módulos funcionan de la misma forma. Tras realizarse un *reset* el módulo se encuentra deshabilitado.

Cada módulo CCP posee un registro de 16 bits que puede ser utilizado de tres diferentes formas:

- Registro de 16 bits para capturar el valor del *Timer* (captura).
- Registro de 16 bits para comparar su valor con el valor del *Timer* (comparador).
- Registro de 10 bits para el ciclo de trabajo de una señal PWM.

El modo que es de nuestro interés es el modo PWM, ya que, el PIC 16F887 no cuenta con un módulo de conversión digital a analógico.

3.4.3 Modo PWM

El modo PWM permite obtener en los pines CCPx una señal periódica, en dicha señal se puede modificar su ciclo de trabajo (*Duty Cicle*), esto se puede lograr variando el tiempo en el cual la señal se encuentre en nivel alto o cuando la señal se encuentre en nivel bajo. De esta forma se puede tener control en el voltaje cuando se encuentre encendido. PWM se encuentra presente

en aplicaciones tales como la velocidad de motores, la luminosidad variable de lámparas, etc. [1]

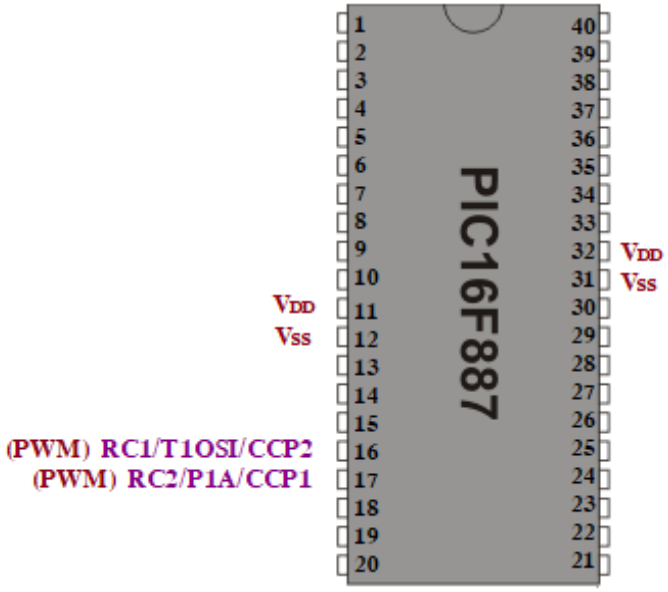


FIGURA 3. 23. PINES DEL MODO PWM EN EL PIC16F887.

3.4.4 Ciclo de trabajo de PWM

El ciclo de trabajo de PWM se especifica al utilizar en total 10 bits. Los 8 bits más significativos provienen del registro CCPR1L y los dos bits menos significativos provienen del registro CCP1CON (DC1B1 y DC1B0).

La señal PWM es una secuencia de pulsos la cual varía su ciclo de trabajo, mediante una frecuencia específica, se tiene un número limitado de combinaciones de ciclos de trabajo, pues este número está limitado por la resolución medida en bits.

Dicha resolución de salida en el PIC 16F887 en PWM es de:

$$\mathbf{10}\ \boldsymbol{bits} = \mathbf{2^{10} = 1024 - 1 = 1023}\ \boldsymbol{ciclos\ de\ trabajo}$$

Para que este módulo funcione adecuadamente es necesario que el pin sea configurado como pin de salida. En el microcontrolador PIC 16F887 la resolución es determinada por el registro PR2 y el máximo valor se obtiene al usar el número FFh. [1], [2]

Las frecuencias y resoluciones de PWM cuando (Fosc =8MHz) son:

Frecuencia del PWM	1.22 [kHz]	4.90 [kHz]	19.61 [kHz]	76.92 [kHz]	153.85 [kHz]	200 [kHz]
Preescalar del temporizador	16	4	1	1	1	1
Valor del PR2	65h	65h	65h	19h	0Ch	09h
Resolución máxima	8	8	8	6	5	5

TABLA 3. 9. FRECUENCIA Y RESOLUCIÓN DE PWM.

El periodo de la señal PWM se obtiene configurando el TIMER2 y el registro PR2. El período de la señal de PWM se puede calcular mediante la siguiente fórmula:

$$PWMT = (PR2 + 1) * 4T_{osc} * (Valor\ de\ preescalar\ del\ TMR2)$$

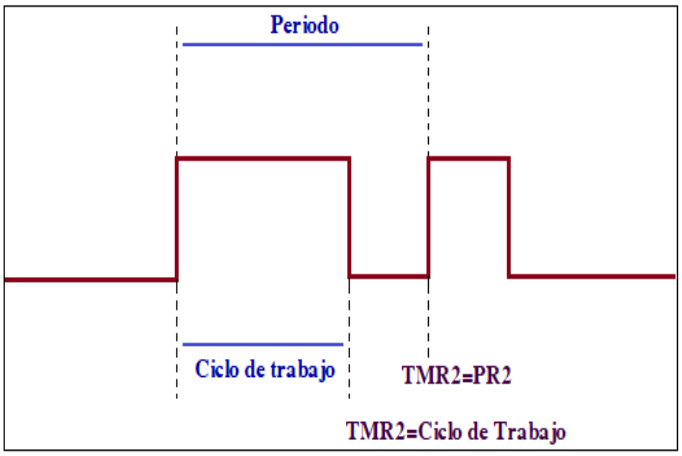


FIGURA 3. 24. SEÑAL PWM EN PIC16F887.

3.4.5 Timer2

El Timer2 es un contador de 8 bits que dispone de pre-escalador (*Prescaler*) y post-escalador (*Postscaler*) programables. El estado del temporizador 2 con el registro de periodo PR2, al igualarse se genera un pulso de salida el cual reinicia el Timer2 en el siguiente pulso de reloj. En el post-escalador del Timer2 se incrementa y su salida es utilizada para generar una interrupción siempre y cuando se encuentre habilitado. Los registros TMR y PR2 son de escritura y lectura. El reinicio del TMR2 puede ser utilizado para determinar la velocidad de transmisión.

3.4.6 Registro CCP1CON

P1M1, P1M0 (PWM *Output Configuration bits*, bits de configuración del modo PWM). Los pines P1B, P1C y P1D actúan como los pines de E/S del puerto D, en cuanto el pin P1A es la entrada del módulo de Captura/Comparación en todos los modos excepto en el modo PWM. En el modo PWM estos bits afectan el funcionamiento del módulo CCP1 como se muestra en la tabla 3.10. [2], [3]

P1M1	P1M0	Modo
0	0	PWM con una sola salida.
		P1A sale una señal modulada, en P1B, P1C y P1D son E/S del puerto D.
0	1	Configuración Full Bridge Forward.
		P1D sale una señal modulada y en P1B y P1C se encuentran inactivos.
1	0	Configuración Half Bridge.
		Pin P1A y P1B sale una señal modulada, en P1C y P1D son E/S del puerto D.
1	1	Configuración Full Bridge Reverse.
		P1B sale una señal modulada, en P1C se encuentra activo, mientras P1A y P1D están inactivos.

TABLA 3. 10. BITS DEL MODO PWM QUE AFECTAN AL MÓDULO CCP1.

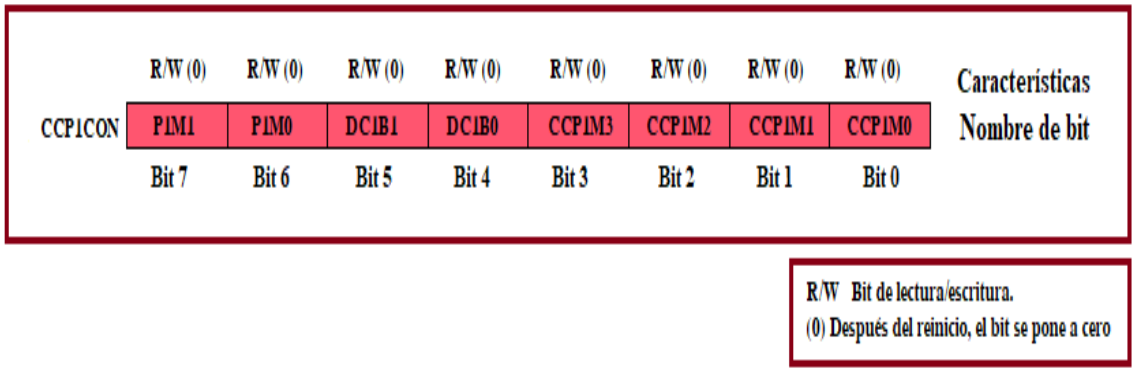


FIGURA 3. 25. REGISTRO CCP1CON.

DC1B1, DC1B0-PWM *Duty Cycle Least Significant bits* (bits menos significativos del ciclo de trabajo PWM) es utilizado solo en modo PWM, este bit es el que representa los dos bits menos significativos de los 10 bits que tiene la resolución de salida de PWM. Además de que este bit determina el ciclo de trabajo de la señal PWM. El resto de los bits se almacenan en el registro CCPR1L.

CCP1M3	CCP1M2	CCP1M1	CCP1M0	MODO
0	0	0	0	**Modulo esta deshabilitado.**
0	0	0	1	No utilizado.
0	0	1	0	**Modo de comparación**
				El bit CCP1IF bit se pone a 1 al ocurrir una coincidencia.
0	0	1	1	No utilizado.
0	1	0	0	**Modo de captura**
				Cada flanco negativo en el pin CCP1
0	1	0	1	**Modo de captura**
				Cada flanco positivo en el pin CCP1
0	1	1	0	**Modo de captura**
				Cada cuarto flanco positivo en el pin CCP1
0	1	1	1	**Modo de captura**
				Cada decimosexto flanco positivo en el pin CCP1
1	0	0	0	**Modo de comparación**
				La salida y CCP1IF se pone a 1 al ocurrir una coincidencia.
1	0	0	1	**Modo de comparación**
				La salida se pone a 0 y el CCP1IF se pone a 1al ocurrir una coincidencia.
1	0	1	0	**Modo de comparación**
				Llega solicitud de interrupción y CCP1IF se pone a 1 al ocurrir una coincidencia
1	0	1	1	**Modo de comparación**
				CCP1IF se pone a 1, en cuanto a los registros de temporizador 1 o 2 se borran cuando se tiene una coincidencia.
1	1	0	0	**Modo PWM**
				P1A y P1C están activos a nivel alto. P1B y P1D están activos a nivel alto.
1	1	0	1	**Modo PWM**
				P1A y P1C están activos a nivel alto. P1B y P1D están activos a nivel bajo.
1	1	1	0	**Modo PWM**
				P1A y P1C están activos a nivel bajo. P1B y P1D están activos a nivel alto.
1	1	1	1	**Modo PWM**
				P1A y el P1C están activos a nivel bajo. P1B y P1D están activos en nivel bajo.

TABLA 3. 11. MODO ACTIVO DE ACUERDO A LOS BITS DEL REGISTRO CCP1CON.

3.4.7 Módulo CCP1 en modo mejorado

El módulo CCP1 es el único módulo que se encuentra en modo mejorado, de hecho, funciona de la misma manera que el modo normal, la única mejora es en la transmisión de la señal PWM a los pines de salida. [2], [3]

3.4.8 Modo PWM con una salida

Este modo solo está habilitado para el caso en el cual P1M1 y P1M0 se encuentren en cero en el registro CCP1CON. En este modo la señal PWM se encuentra disponible simultáneamente en máximo cuatro diferentes pines de salida y dicha señal puede aparecer normal o invertida. En caso de obtener una señal invertida, los pines en estado bajo y los pulsos que tienen la misma forma de la onda se generan en parejas quedando como P1A y P1C, así como en P1B y P1D. [2], [3]

3.4.8.1 Modo de Medio-Puente

En este modo la señal PWM se da en la salida P1A, en cuanto a la señal complementaria PWM es una salida en P1B. Este modo se utiliza para activar los controladores MOSFET que habilitan/deshabilitan el flujo de corriente por el dispositivo.

Una de las desventajas de este modo es que al encender los controladores MOSFET simultáneamente se puede ocasionar un corto circuito, para evitar esta situación se da un tiempo muerto (*time delay*) entre el encendido y apagado de los controladores. [2], [3]

3.4.8.2 Modo Puente-Completo

En el modo de Puente completo se utilizan cuatro pines como salidas, este modo se utiliza para activar motores y tener un control en la velocidad y en la dirección de rotación. De este tipo de modo se desglosan dos modos:

- Modo *Full Bridge-Forward* (puente completo con salida directa)
  En modo directo el pin P1A se encuentra en alto (1, HIGH), la secuencia pulsos aparece en P1D y los pines P1B y P1C se encuentran en bajo (0, LOW), como se muestra en la figura 3.26.

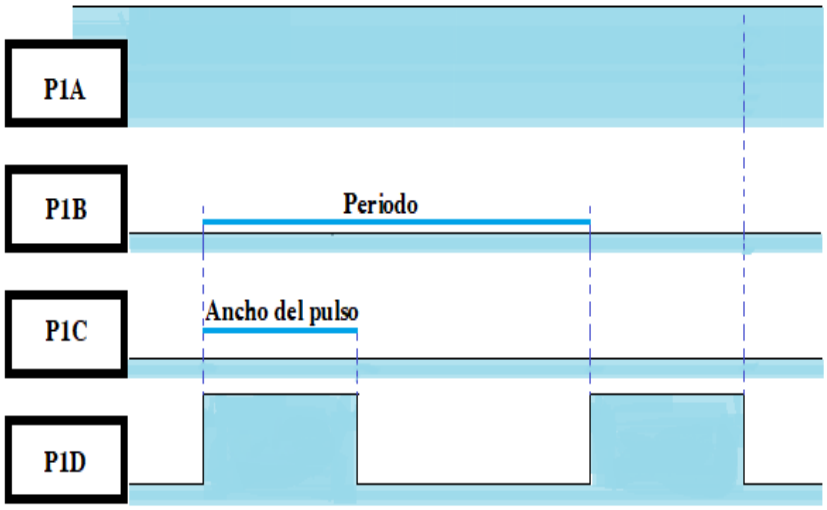


FIGURA 3. 26. MODO FULL BRIDGE-FORWARD.

- Modo *Full Bridge-Reverse* (puente completo con salida inversa)
  En el modo inverso el pin P1C se encuentra en alto (1, HIGH), mientras que los pulsos aparecen en el pin P1B y el P1A y P1D se encuentran en bajo (0, LOW), como se muestra en la figura 3.27.

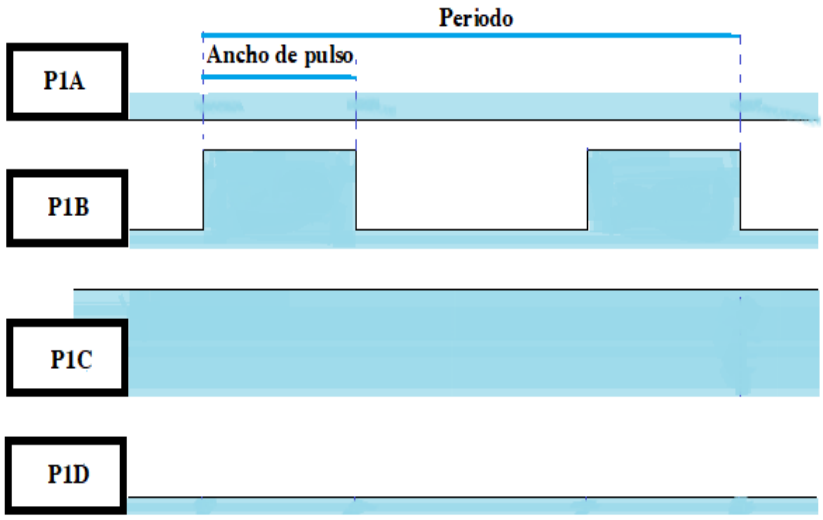


FIGURA 3. 27. MODO FULL BRIDGE-REVERSE.

El registro con el que cuenta el PIC16F887 es de 10 bits para el ciclo de trabajo de una señal PWM por lo cual tendremos:

$$2^{10} = 1024 = 1024 - 1 = 1023 \text{ posibles valores}$$

En la tarjeta de desarrollo EasyPIC V6, el correcto funcionamiento del módulo, se debe a las siguientes funciones que se encuentran dentro de la librería especializada de PWM,

- *PWM1_init* que tiene el comando: *void Pwm1_init(long freq*); donde Freq ajusta la frecuencia de la señal PWM dada en Hz
- *PWM1_Start* que tiene el comando: *void Pwm1_Start(void).*
- *PWM1_Set_Duty* que tiene el comando: *void Pwm1_Set_Duty (unsigned short duty_ratio);* donde *duty_ratio* ajusta la duración de pulsos en una secuencia de pulsos.

Esta librería también contiene la función PWM_Stop la cual deshabilita el modo y se escribe como *void Pwm1_Stop(void);*

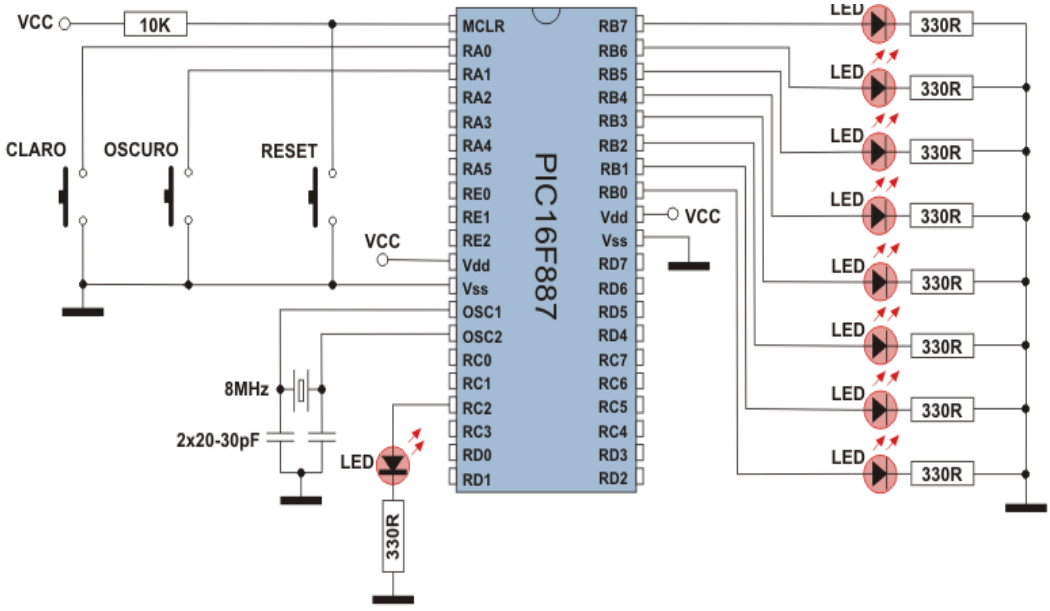
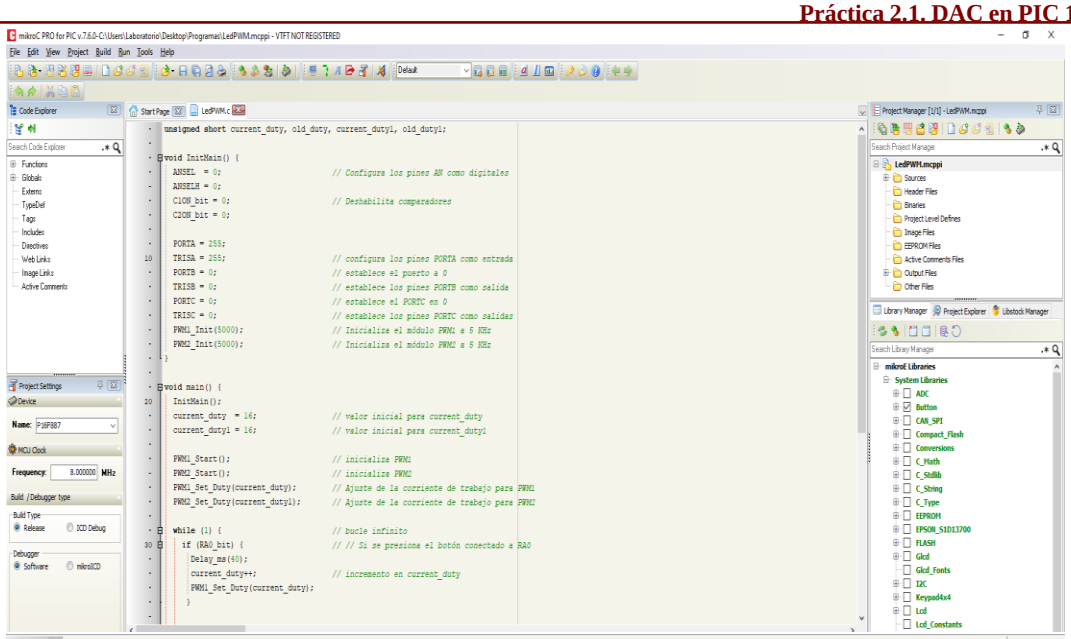


FIGURA 3. 28. DIAGRAMA PARA PWM.

Para que este módulo funcione es necesario activar las librerías de PWM y Button.



```
unsigned short current_duty, old_duty, current_duty1, old_duty1;

void InitMain() {
  ANSEL  = 0;                 // Configura los pines AN como digitales
  ANSELH = 0;
  C1ON_bit = 0;               // Deshabilita comparadores
  C2ON_bit = 0;

  PORTA = 255;
  TRISA = 255;                // configura los pines PORTA como entrada
  PORTB = 0;                  // establece el puerto a 0
  TRISB = 0;                  // establece los pines PORTB como salida
  PORTC = 0;                  // establece el PORTC en 0
  TRISC = 0;                  // establece los pines PORTC como salidas
  PWM1_Init(5000);            // Inicializa el módulo PWM1 a 5 KHz
  PWM2_Init(5000);            // Inicializa el módulo PWM2 a 5 KHz
}

void main() {
  InitMain();
  current_duty  = 16;         // valor inicial para current_duty
  current_duty1 = 16;         // valor inicial para current_duty1

  PWM1_Start();               // inicializa PWM1
  PWM2_Start();               // inicializa PWM2
  PWM1_Set_Duty(current_duty);// Ajuste de la corriente de trabajo para PWM1
  PWM2_Set_Duty(current_duty1);// Ajuste de la corriente de trabajo para PWM2

  while (1) {                 // bucle infinito
    if (RA0_bit) {            // // Si se presiona el botón conectado a RA0
      Delay_ms(40);
      current_duty++;         // incremento en current_duty
      PWM1_Set_Duty(current_duty);
    }
```

FIGURA 3. 29. PROGRAMA DE PWM EN PIC 16F887 EN MIKROC PRO.

3.5 Práctica 2.2. DAC Arduino UNO.

Equipo y material empleado

- Arduino UNO
- Protoboard
- Jumper macho-macho

Práctica DAC en Arduino UNO

3.5.1 Conversión Digital-Analógica (DAC)

Cuando una señal es adquirida de un sistema digital por medio de un proceso de conversión analógico/digital, con frecuencia dicha señal se convierte a otra señal analógica. Por ejemplo, el procesamiento digital de audio, en donde la voz se convierte en una señal digital, la cual es modificada, procesada y convertida de nuevo a una señal analógica. Dicho proceso de convertir señales digitales en señales analógicas se realiza con un convertidor digital/analógico.

Una de las técnicas más utilizadas para convertir señales digitales en señales analógicas es por medio del método de suma en el cual una señal digital (representada por 1 o 0) entra en el convertidor digital/analógico desde el bit más significativo hasta el bit menos significativo, donde un comparador comprueba su estado lógico (*High, Low*), el cual es representado por un nivel de voltaje de 5 [V] o 0 [V] para representar 1 o 0 lógico. Una vez procesados todos los bits de la señal digital, los niveles de voltaje obtenidos se suman para producir el valor final del voltaje analógico.

3.5.2 DAC convertidores externos

En algunas versiones de microcontroladores no cuentan con un módulo DAC integrado, es por ello que existe una amplia variedad de módulos externos que proporcionan dicha función, por mencionar algunos:

- DAC monocanal de 8 bits conectado a través de un puerto paralelo.
- DAC de 8 bits y cuatro canales, conectado a través de un puerto paralelo.
- Canal cuádruple, DAC de 8 bits conectado a través de SPI.
- Canal octal, DAC de 8 bits conectado a través de SPI.

3.5.3 DAC con el entorno de desarrollo de Arduino

Arduino UNO no cuenta con un DAC (*Digital to Analog Converter*, Convertidor Digital Analógico) integrado, por ello, se aprovecha que los microcontroladores emiten un voltaje de entre 0 y 5 [V], lo cual puede aproximar el efecto de un DAC mediante la creación de una señal PWM (*Pulse Width Modulation*, Modulación de Ancho de Pulso).

Una señal PWM está formada por un ciclo de trabajo y por una frecuencia. Los pines en Arduino UNO enfocados a PWM se pueden identificar por un ~, en dicha placa se tienen 6 pines los cuales son 3, 5, 6, 9, 10 y 11. Arduino UNO tiene una resolución de 8 bits, por lo que el ciclo de trabajo se puede ajustar a $2^8 = 256$ valores diferentes.

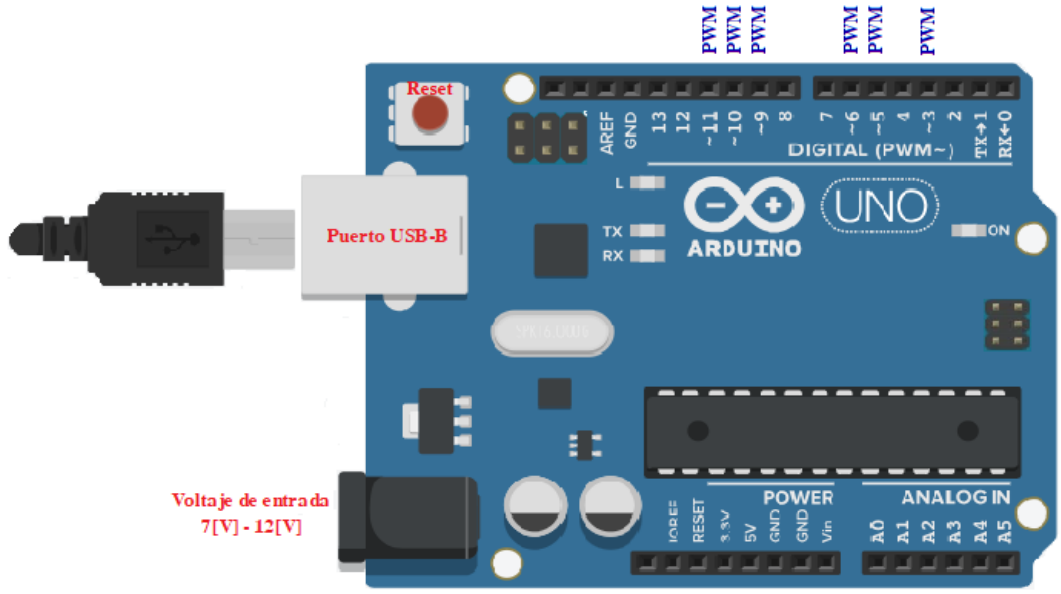


FIGURA 3. 30. PINES EN ARDUINO UNO PARA PWM.

Un periodo de la señal está formado por un ciclo de encendido y un ciclo de apagado, en dicho periodo la fracción durante la cual la señal se encuentra encendida es lo que comúnmente se conoce como ciclo de trabajo. Por medio de PWM se puede controlar la potencia entregada mediante el uso de señales de ON/OFF. En cuanto a la frecuencia de la señal es la que determina la rapidez con la que la señal PWM completa un ciclo, es decir cuánto tarda de cambiar de HIGH a LOW. Con esta acción la salida se puede interpretar como una señal analógica de voltaje el cual se proporciona a los dispositivos.

PWM es utilizado para imitar el control analógico, pero no siempre se puede utilizar este método. Por ejemplo, PWM funciona muy bien en controlar motores de corriente continua a velocidades variables, pero es deficiente en controlar altavoces ya que necesita de un módulo externo.

El comando llamado *analogWrite* utilizado en el entorno de desarrollo de Arduino emite una señal de 0 a 5 VDC, en donde es enviada una señal constante de 0 a 255 utilizando modulación por ancho de pulsos (PWM). Dicha señal, sirve como señal de CC.

La forma del comando *analogWrite* es la siguiente:

analogWrite (pin, value) donde;

- Pin es el número de pin.

Value es el valor utilizado para controlar el ciclo de trabajo de la señal PWM (0 a 255)

PWM funciona modulando el ciclo de trabajo de una onda cuadrada, lo cual se refiere al porcentaje de tiempo en donde la onda permanece encendida o apagada. Por ejemplo, en una onda cuadrada si se tiene un ciclo de trabajo del 50% estará en alto la mitad del tiempo y en bajo la otra mitad.

Con el comando de *analogWrite* se puede asignar el ciclo de trabajo de la onda:

- En 0 se tiene un 0
- En 64 se tiene un 25
- En 127 se tiene un 50
- En 191 se tiene un 75
- En 255 se tiene un 100

Por ejemplo, una mayor cantidad de bits requiere mayor precisión al igual que mayor complejidad.

La placa Arduino dispone de Convertidor ADC (A0 a A5)

- Convierten valores de voltaje hasta un máximo de 5 [V]
- Sus pines analógicos utilizan 10 bits
- Resolución de $2^{10} = 1024$ posibles valores.
- Cada escalón de medida es de 5 [V] /1024 = 0.0048828125 [V] = 4.88 [mV]

En Arduino la frecuencia de PWM es de 500 [Hz] (este valor puede modificarse) la frecuencia de los pines es de 490 [Hz] aproximadamente, excepto en los pines 5 y 6 que es de 980 [Hz] para el Arduino UNO

$$\frac{1}{500}\ [Hz] = 0.002\ seg.$$

En salidas PWM se modificará el ciclo de trabajo de la señal para que actúe de forma similar a una salida analógica.

Una limitación de las salidas PWM en Arduino es que estas salidas son de 10 bits con lo que la señal es de aproximadamente de 4[mV].

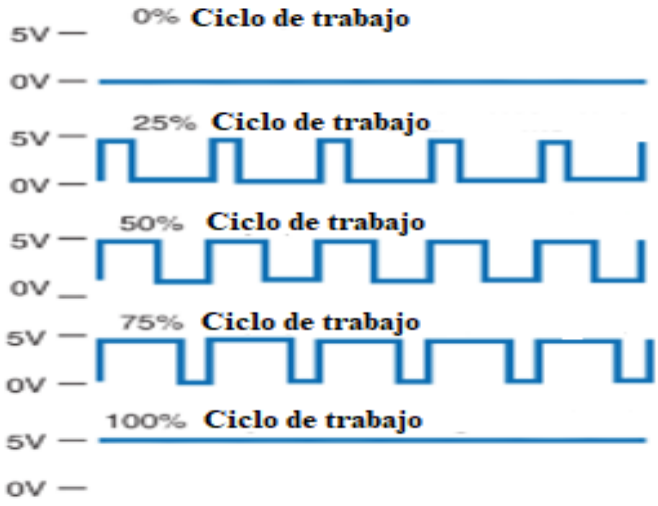


FIGURA 3. 31. CICLO DE TRABAJO

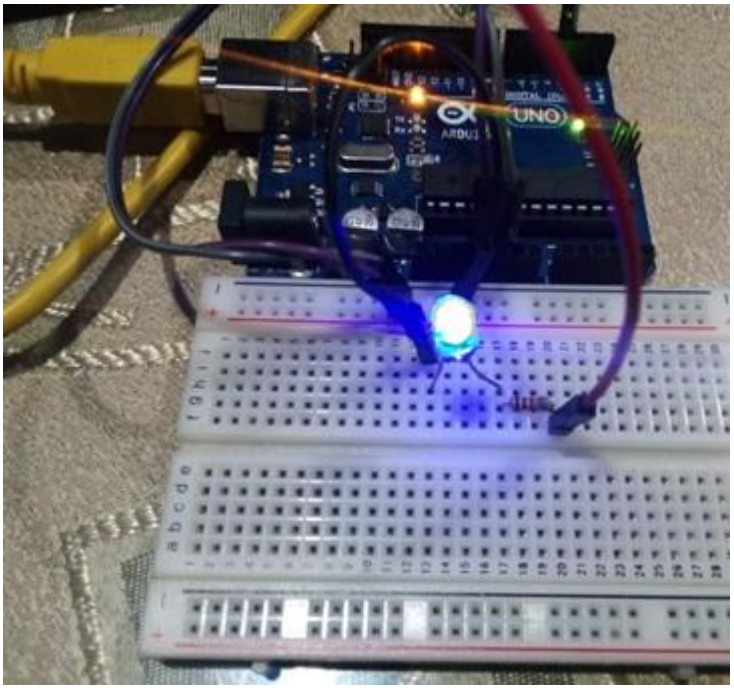


FIGURA 3. 32. CICLO DE TRABAJO A 20%

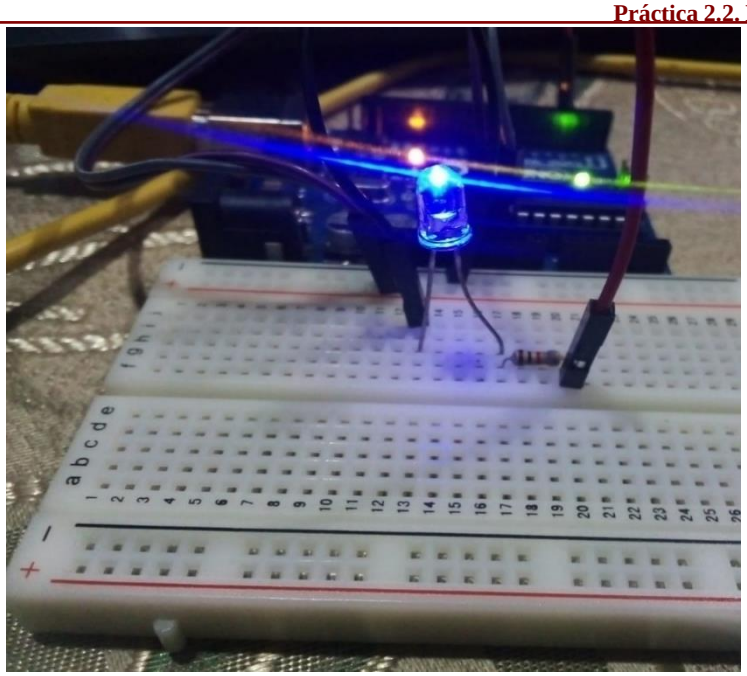


FIGURA 3. 33. CICLO DE TRABAJO A 50%

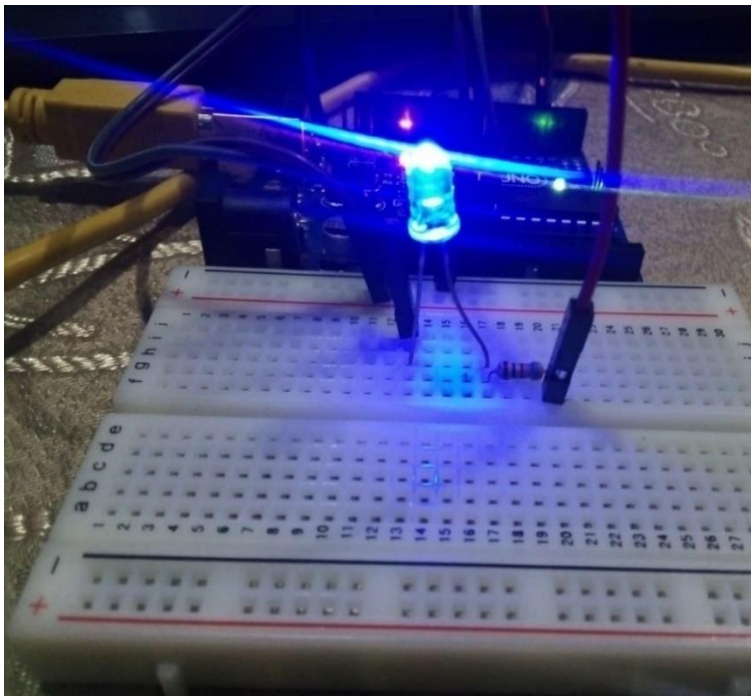


FIGURA 3. 34. CICLO DE TRABAJO A 80%

3.6 Práctica 2.3. DAC Raspberry Pi 4.

Equipo y material empleado

- Raspberry Pi 4 modelo B
- Protoboard
- Jumper macho-macho
- Módulo MCP4725

3.6.1 Práctica DAC en Raspberry Pi 4

Los microcontroladores funcionan solo con valores digitales, pero el mundo real es de señales analógicas. Es por eso que ADC está ahí para convertir valores analógicos en digital para que los microcontroladores puedan procesar las señales.

A diferencia de PIC y Arduino, Raspberry Pi no cuenta con entradas analógicas ni un convertidor integrado. Si lo que se requiere es usar sensores, cuyos valores no pueden ser emitidos digitalmente, se utiliza un convertidor digital analógico (DAC).

3.6.1.1 Módulo DAC MCP4725

MCP4725 es un módulo convertidor digital analógico de 12 bits que se utiliza para generar voltajes analógicos de salida de (0-5V) y se controla mediante comunicación I^2C. Este módulo cuenta con memoria EEPROM no volátil integrada.

Tiene una salida analógica estable

Características

- Resolución de 12 bits, $2^{12} = 4096\ ciclos\ de\ trabajo$.
- Interfaz I^2C
- Funciona con un voltaje de 2.7 [V] a 5.5 [V], el voltaje máximo que puede proporcionar es Vcc.
- La corriente máxima que puede proporcionar es de 25 [mA].
- El tiempo de cambio de salida típico es de 6 [µs].
- EEPROM interna para almacenar configuraciones.
- Diagrama

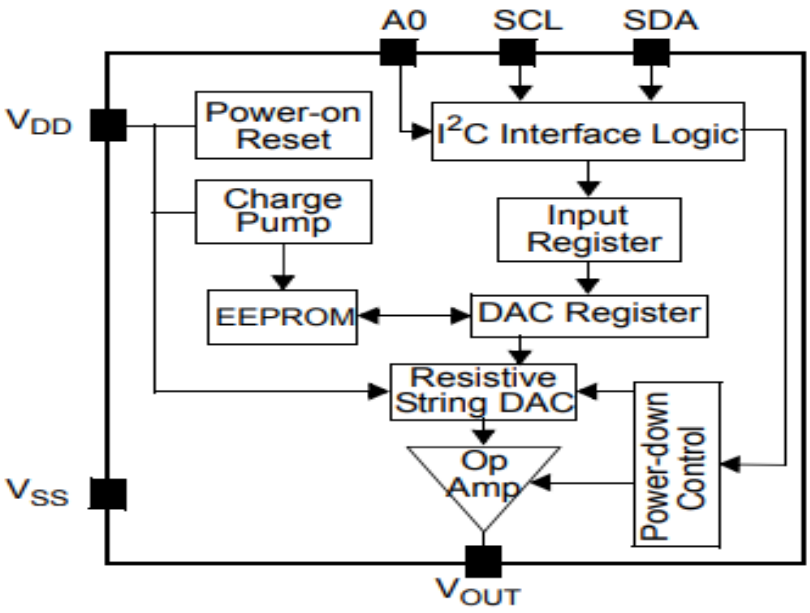


FIGURA 3. 35. MÓDULO MCP4725.

- Para que Raspberry Pi 4 pueda devolver una señal analógica a partir de datos digitales necesitamos un convertidor digital-analógico (DAC).
- Convertiremos una señal por medio de PWM. Que contiene la información modulada dentro de una señal cuadrada con frecuencia fija, con ancho de pulso variable.
- Para configurar la salida analógica debemos indicarle al código que pin vamos a usar con PWM y la frecuencia de la señal.
- Usamos como variables:

  - **PWM( )**
  - **pwm.freq( )**

Referencias de Práctica 2. DAC

PIC

[1] García, Eduardo. (2009). Compilador C CCS y simulador Proteus para microcontroladores PIC. México: Alfaomega Grupo Editor.

[2] Verle, M. (2009). PIC Microcontrollers: Programming in C. Mikroelektronika.

[3] Benchimol, Daniel. Microcontroladores. Funcionamiento, Programación y Aplicaciones Prácticas, USERS.

[4] Bates, Martin. (2008). Programming 8-bit PIC microcontrollers in C: with Interactive Hardware Simulation. United States of America: ELSEVIER.

[5] Katzen, Sid. (2013). The Quintessential PIC® Microcontroller. United States of America: Springer Science & Business Media.

Arduino

[1] Kaul, Lukas. (2023). Practical Arduino Robotics: A hands-on guide to bringing your robotics ideas to life using Arduino. Packt Publishing Ltd.

[2] Kosky, Philip; Balmer, Robert; Keat, William y Wise, George. (2020). Exploring engineering: An Introduction to Engineering and Design. Academic Press.

[3] Blum, Jeremy. (2013). Exploring Arduino: Tools and Techniques for Engineering Wizardry. John Wiley & Sons.

Raspberry Pi

[1] MICROCHIP. (2009). 12-Bit Digital-to-Analog Converter with EEPROM Memory in SOT-23-6 https://ww1.microchip.com/downloads/en/devicedoc/22039d.pdf.

Práctica 3. Conectividad.

Introducción

La conectividad amplía la capacidad de almacenamiento de información y mejora las comunicaciones. Una buena conectividad ofrece una comunicación de calidad y confiable.

Cada tecnología dispone de diversas opciones para comunicarse mediante dispositivos electrónicos. Cada uno de los dispositivos tiene una capacidad de conectividad de acuerdo a sus características, puede ser una conectividad fija, móvil o inalámbrica.

Por lo que en esta práctica veremos la conectividad de cada una las tecnologías de PIC, Arduino UNO y Raspberry Pi 4.

Objetivo

Analizaremos y compararemos protocolos de comunicación, en cada una de las 3 tecnologías.

3.7 Práctica 3.1. Conectividad en PIC16F887

Equipo y material empleado

- PIC16F887
- Tarjeta de desarrollo EasyPIC V6
- Cable RS-232 macho – USB macho
- PC

Los microcontroladores PIC utilizan dos modos de transmisión en serie:

- Puerto serie síncrono (SSP)
- Interfaz de comunicación serie (SCL) o receptor transmisor serie síncrono/asíncrono (USART)

SSP se utiliza en la comunicación con otros microcontroladores o con periféricos. Sus dos interfaces son:

- SPI *(Serial Peripheral Interface*, Interfaz Serial Periférica)
- Modo I^2C (*Inter-Integrated Circuit*, Circuito inter-integrado)

3.7.1 Protocolo de comunicación USART

El modo USART transmite y recibe datos en serie; este modo funciona de dos formas:

- Síncrona: la cual utiliza una señal de reloj y una línea de datos, se permite la transmisión continua de datos y no existe límite de tamaño. Su transmisión es semiduplex.
- Asíncrona: no se envía la señal de reloj, por lo cual tanto el transmisor como el receptor tienen relojes con la misma frecuencia. Cada trama de datos tiene un tamaño fijo y posee un bit inicial o de arranque y un bit final o de parada. Su transmisión es full-duplex.

La USART síncrona es utilizada cuando se tienen distancias cortas, en tanto la USART asíncrona es utilizada con distancias largas. El modo USART transfiere tramas de datos de 8 bits o 9 bits para transmisión y para detección de errores de transmisión, también genera interrupciones cuando se realiza una recepción de datos o cuando es completada la transmisión de datos. La transmisión consiste en enviar los datos de bit a bit a través de una línea común en periodos de tiempo fijo.

La versión nueva de este modo es llamada EUSART (*Enhanced Universal Synchronous Asynchronous Receiver Transmitter*) se encuentra en el PIC 16F887 tiene características tales como:

- Transmisión y recepción asíncrona en modo full-duplex;
- Longitud de 8-9 bits programables.
- Detección de dirección
- Detección de errores por saturación del búfer de entrada

- Comunicación *half-duplex* en modo síncrono.
- Maestro síncrono semiduplex
- Esclavo síncrono semiduplex
- Funcionamiento en reposo

El modo EUSART también es conocida como Interfaz de Comunicaciones Serie (SCI) se puede configurar como un sistema asíncrono *full-duplex* o síncrono *half-duplex*, el modo *full-duplex* es utilizado para comunicaciones terminales CRT y PC. En su modo síncrono se aplica a comunicaciones con circuitos integrados A/D o D/A, EEPROM serie u otros microcontroladores, no cuenta con reloj interno de transmisión y requiere de una señal externa proporcionada por un dispositivo maestro síncrono. EUSART contiene los generadores de señales de reloj, registros de desplazamiento para realizar transmisión de datos serie de E/S.

El módulo EUSART funciona por medio de tres registros los cuales son:

- Estado y control de transmisión (TXSTA).
- Estado y control de recepción (RCSTA).
- Control de velocidad de transmisión (BAUDCTL).

3.7.1.1 Módulo Puerto Serie Síncrono Maestro

El MSSP (*Master Synchronous Serial Port*, Puerto Serie Síncrono Maestro) es el módulo que permite realizar la transmisión de datos a alta velocidad ya sea, entre un microcontrolador y un periférico o con otro microcontrolador, mediante sus E/S (utilizando dos o tres líneas). Este tipo de comunicación suele ser síncrona y es utilizada entre un maestro y entre uno o varios esclavos. Este módulo suele funcionar en dos modos los cuales son:

- Modo SPI
- Modo I^2C

El funcionamiento de ambos modos depende de si el módulo trabaja como maestro o esclavo. Cuando el módulo trabaja como maestro este genera una señal de reloj para lograr transmitir o recibir, es decir que el maestro es el encargado de establecer la conexión. Por otra parte, cuando el módulo trabaja como esclavo este siempre tiene que esperar a que un dispositivo maestro envíe la solicitud de transmisión de datos.

3.7. 2 Protocolo de comunicación SPI

Dicho modo permite la transmisión y recepción simultánea de datos de 8 bits, este modo utiliza 3 líneas de E/S las cuales son:

- SDO (*Serial Data Out*, Salida de datos serie) que es la línea de transmisión y se encuentra en el pin 24 (RC5/SDO).
- SDI (*Serial Data In*, Entrada de datos serie) que es la línea de recepción y se encuentra en el pin 23 (RC4/SDI/SDA).
- SCK (*Serial Clock*, Reloj) que es la línea de sincronización y se encuentra en el pin 18 (RC3/SCK/SCL).

Existe una cuarta línea llamada SS (*Slave Select*, Selección de esclavo) la cual se encuentra en el pin 7 (RA5/$\overline{SS}$/AN4), esta cuarta línea solo se encuentra activa si el microcontrolador funciona como esclavo o cuando el dispositivo externo requiere intercambiar datos.

Al funcionar en modo SPI, el módulo MSSP hace uso de 4 registros los cuales son:

- SSPSTAT registro de estado.
- SSPCON registro de control.
- SSPBUF bufer serie de transmisión/recepción.
- SSPSR registro de desplazamiento.

De dichos registros los tres primeros son de lectura/escritura, mientras que el último registro no se puede acceder y es utilizado para convertir los datos en formato serie.

El modo SPI funciona de diferentes forman tales como:

- Modo maestro (SCK es la salida del reloj).
- Modo esclavo (SCK es la entrada del reloj).
- Frecuencia de reloj (modo maestro).
- Modo de selección de esclavo (modo esclavo).
- Fase de muestreo de entrada de datos.

Para que funcione el modo SPI los pines SDI, SDO, SCK y SS deben de cumplir lo siguiente:

- SDI es controlado automáticamente por el módulo SPI.
- SDO debe tener el TRISC borrado.
- SCK (modo maestro) debe tener el bit TRISC despejado.
- SCK (modo esclavo) debe tener el bit TRISC establecido.
- SS debe tener el bit TRISA establecido.

3.7.3 Protocolo de comunicación I^2C

Este modo es utilizado cuando el microcontrolador debe intercambiar los datos con un circuito integrado dentro de un mismo dispositivo, por ejemplo, en memorias, sensores, circuitos especializados, etc. En el modo I^2C la transmisión de datos es síncrona y bidireccional, en ella, solo son utilizados dos pines para realizar la transmisión de datos los cuales son:

- SDA (Datos seriales) se encuentra en el pin 23 (RC4/SDI/SDA)
- SCL (Reloj serial) se encuentra en el pin 18 (RC3/SCK/SCL).

Los dos pines mencionados anteriormente, se configuran como E/S por los bits TRISC.

El registro SSPCON permite seleccionar diferentes modos de I^2C los cuales son:

- Modo I^2C maestro.
- Modo esclavo I^2C (direccionamiento de 7 bits).
- Modo esclavo I^2C (direccionamiento de 10 bits).
- Modo esclavo I^2C (direccionamiento de 7 bits) con interrupciones de bits de inicio y paradas activadas.
- Modo esclavo I^2C (direccionamiento de 10 bits) con interrupciones de bits de inicio y paradas activadas.
- Funcionamiento maestro controlado por firmware I^2C, en ésta el esclavo está inactivo.

En I^2C se pueden conectar hasta 112 diferentes componentes al utilizar solo dos pines de E/S. El número de dispositivos que se pueden conectar está limitado por la capacidad de direccionamiento que tiene el cual puede ser de 7 bits a 10 bits.

El reloj, que es el encargado de sincronizar ambos dispositivos, es generado por el dispositivo maestro (microcontrolador) y su frecuencia afecta a la velocidad de transmisión de datos. El protocolo tiene una frecuencia máxima de 3.4 [MHz] (I^2C de alta velocidad), de igual forma cuenta con una frecuencia estándar de 100 [kHz] a 400 [kHz] o 1 [MHz].

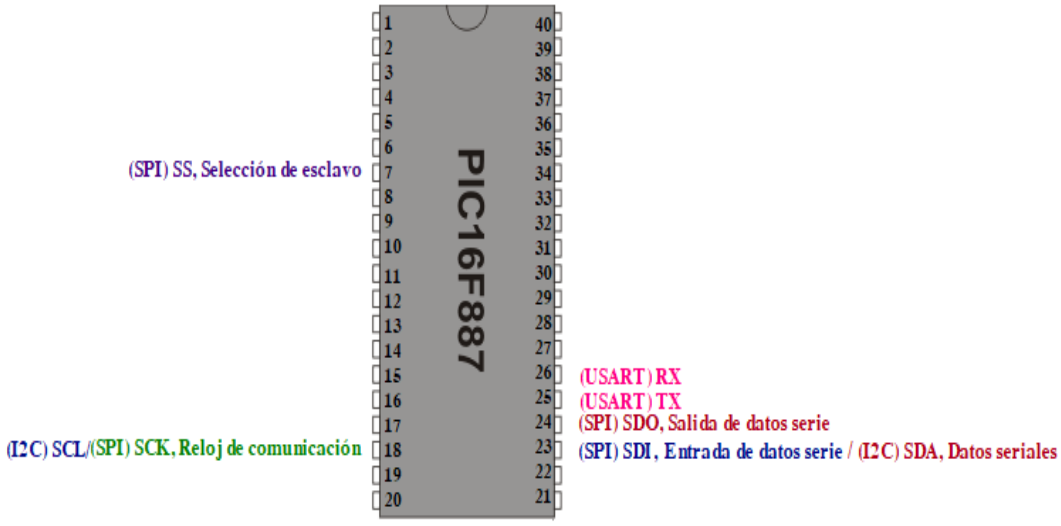


FIGURA 3. 36. COMUNICACIONES EN SERIE DEL PIC16F887.

La comunicación serial en la tarjeta de desarrollo, la realizamos conectando el cable convertidor USB a serial RS232 en la tarjeta de desarrollo y en la PC, utilizando el módulo USART del microcontrolador.

FIGURA 3. 37. MÓDULO DE COMUNICACIÓN SERIE RS-232

Los pines del microcontrolador utilizados en la comunicación serie son:

- RX (*receive data*) / recibir datos.
- TX (*transmit data*) / transmitir datos.
- CTS (*clear to send*) / permitido para transmitir.
- RTS (*request to send*) / listo para enviar.

La velocidad que utiliza es de 115 [Kbps], para realizar la conexión es necesario tener los *drivers* (controladores) necesarios para la transferencia de datos, al trabajar en la tarjeta de desarrollo EasyPIC no se contaban con dichos controladores y por lo tanto utilizamos los controladores Prolific.

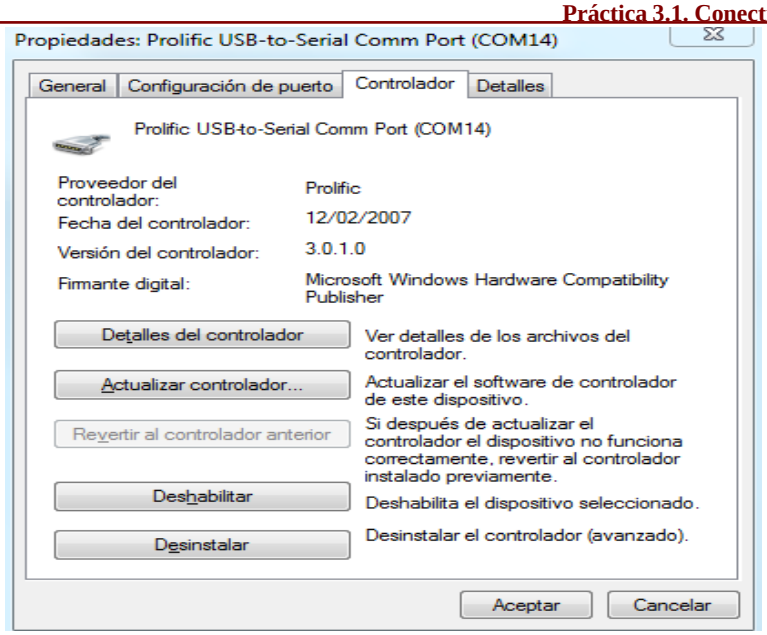
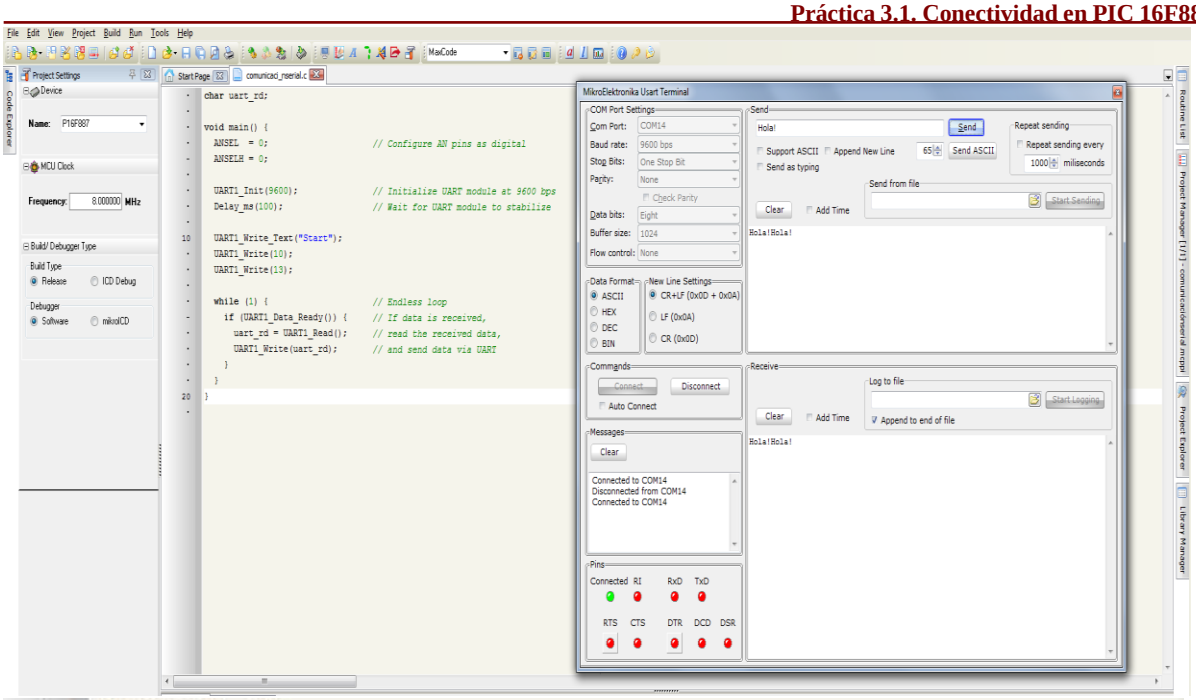


FIGURA 3. 38. CONTROLADOR PROLIFIC PARA LA COMUNICACIÓN SERIAL ENTRE LA TARJETA DE DESARROLLO Y LA PC.

Una vez que hemos cargado el programa en la tarjeta de desarrollo EasyPIC v6 a través del programa MikroC PRO, podemos monitorear la transmisión observando el ícono de una PC que representa la terminal serial. Al abrir esta terminal, si notamos que la transmisión de datos no está ocurriendo, es probable que los pines mencionados anteriormente estén inactivos (representados en rojo), tal como se muestra en la Figura 3.40. Dentro de la terminal, debemos seleccionar el puerto COM correcto y activar la conexión. Una vez que hayamos seguido estos pasos, los LEDs indicadores, cambiarán su color a verde, tal como se muestra en la Figura 3.41.




```
char uart_rd;

void main() {
  ANSEL  = 0;                 // Configure AN pins as digital
  ANSELH = 0;

  UART1_Init(9600);           // Initialize UART module at 9600 bps
  Delay_ms(100);              // Wait for UART module to stabilize

  UART1_Write_Text("Start");
  UART1_Write(10);
  UART1_Write(13);

  while (1) {                 // Endless loop
    if (UART1_Data_Ready()) { // If data is received,
      uart_rd = UART1_Read(); // read the received data,
      UART1_Write(uart_rd);   // and send data via UART
    }
  }
}
```

FIGURA 3. 39. PROGRAMA EN MIKROC PARA PIC 16F887.

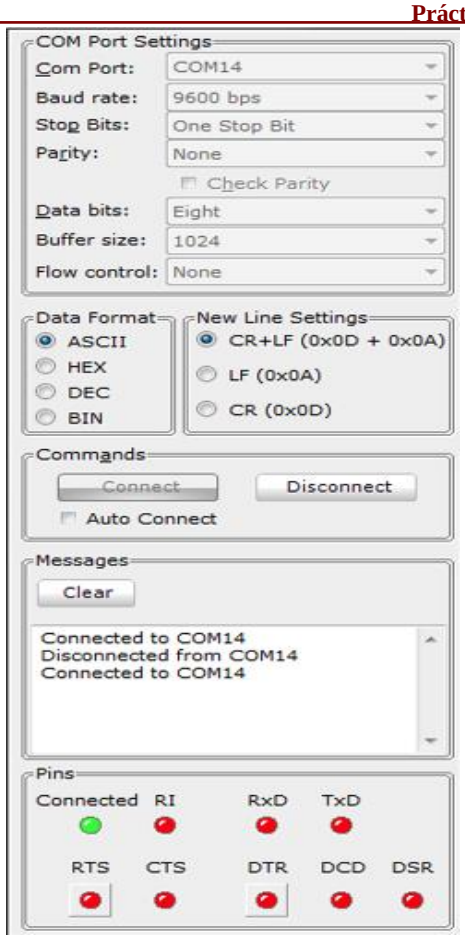
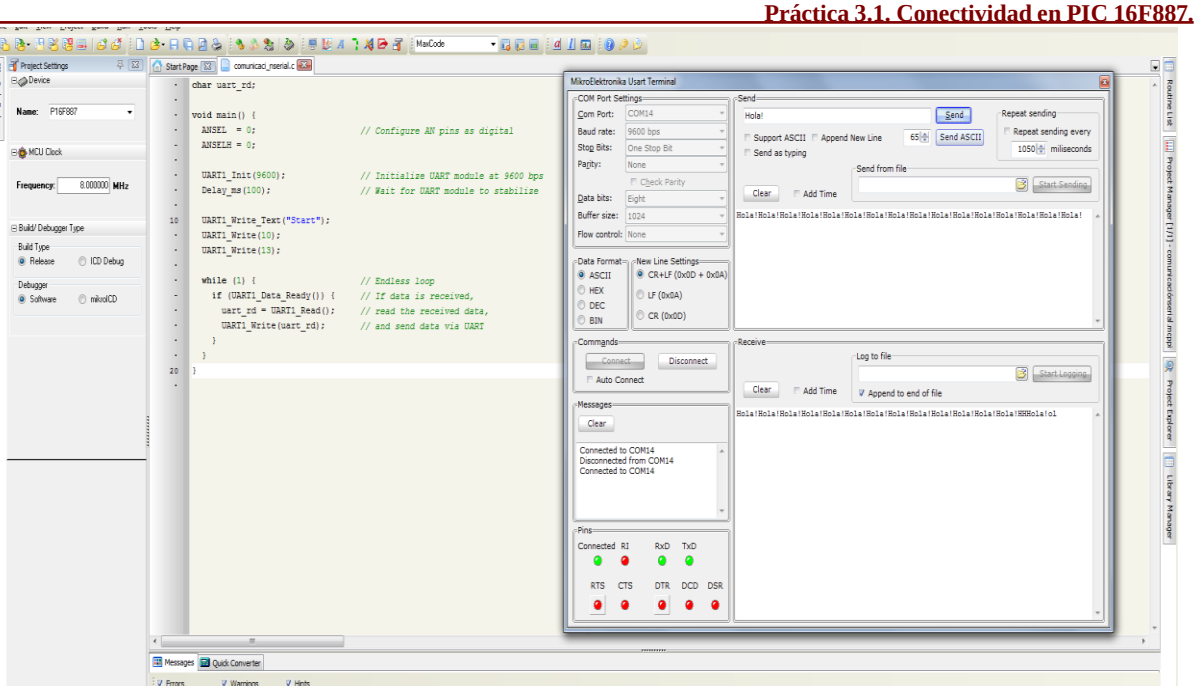


FIGURA 3. 40. ERROR EN LA TRANSMISIÓN DE DATOS EN LA COMUNICACIÓN RS- 232



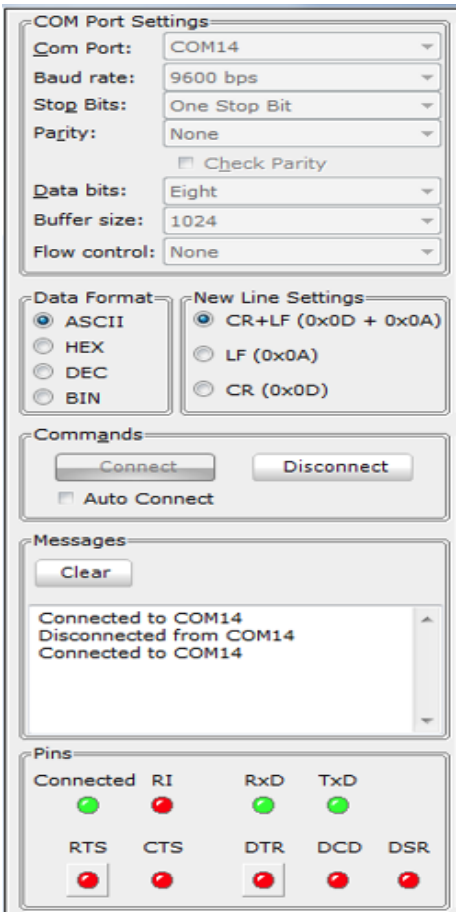


FIGURA 3. 41. TRANSMISIÓN CORRECTA DE LA COMUNICACIÓN RS-232.

3.8 Práctica 3. Conectividad en Arduino UNO

Equipo y material empleado

- 3 placas Arduino UNO
- Protoboard
- Jumper macho-macho
- Cable USB de Arduino.
- PC

La placa Arduino UNO está integrado por el microcontrolador ATmega328, éste se encuentra equipado con un subsistema de comunicación serie, integrado por el receptor y transmisor serie síncrono y asíncrono universal (USART), SPI y TWI, dichos sistemas tienen en común la transmisión de datos en serie, en este tipo de transmisión los datos son enviados de bit en bit desde el transmisor hasta el receptor.

Comunicaciones en Arduino UNO

Los microcontroladores para obtener información realizan un intercambio de datos, ya sea, con otros microcontroladores o con dispositivos periféricos.

Dicha transmisión envía bytes de datos, dependiendo del tipo de transmisión se pueden tener menor número de líneas. En la transmisión en serie un byte de datos se envía de bit en bit, una vez recibidos los ocho bits es reconstruido el byte de datos.

El microcontrolador Arduino UNO cuenta con diferentes protocolos de comunicación en serie algunos de los cuales son:

- USART (*Universal Serial Asynchronous Receiver Transmitter*, Receptor/Transmisor Asíncrono Universal)
- SPI (Interfaz Serial Periférica)
- I^2C (*Inter-Integrated Circuit*) también llamada TWI (*Two Wire Interface,* Interfaz serie de dos cables)

Dichos protocolos de comunicación ocupan en Arduino UNO diferentes pines como se puede observar en la figura. 3.42.

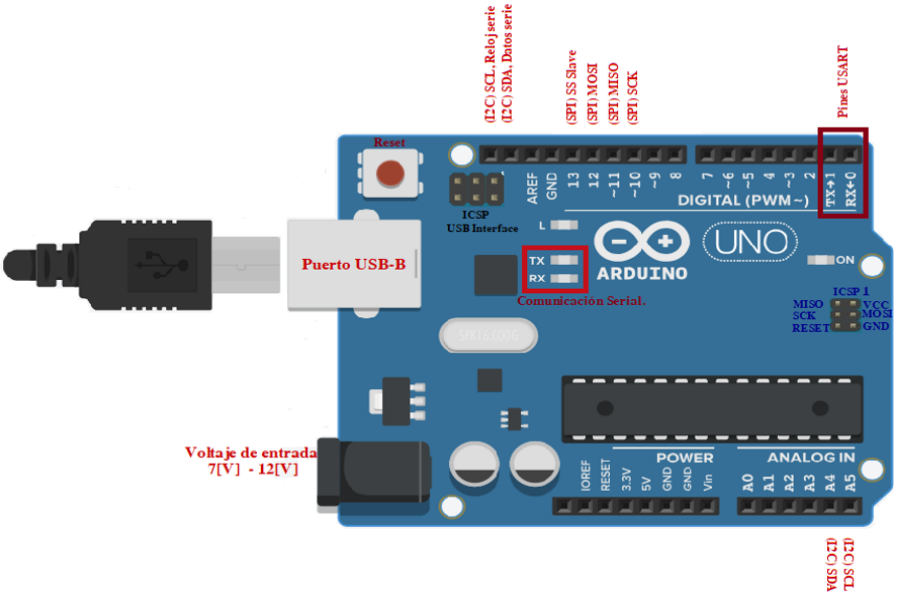


FIGURA 3. 42. PINES PARA COMUNICACIONES I2C Y SPI EN ARDUINO UNO.

3.8.1 Protocolo de comunicación USART

El ATmega328 cuenta con el protocolo de comunicación USART también llamado UART el cual utiliza una comunicación *full-duplex* entre el receptor y el transmisor. Este protocolo normalmente utiliza la comunicación asíncrona, pero de igual forma utiliza el modo síncrono, en la cual no se tiene un reloj común entre la transmisión y la recepción, para mantenerlos sincronizados se utiliza un bit de inicio y parada, al inicio y al final de cada byte de datos.

Este protocolo es bastante flexible ya que, tiene la capacidad de trabajar en diferentes velocidades de transmisión comúnmente conocidas como velocidad de baudios (bits por segundo). De igual forma la USART se puede configurar para trabajar con longitudes de bits de datos de 5 a 9 bits con uno o dos bits de parada. Además, cuenta con un bit de paridad generado por hardware (par o impar) el cual permite la detección de un error dentro de un byte.

La comunicación serie entre dos dispositivos en USART utiliza los pines digitales 0 (RX) y 1 (TX), y mediante el puerto USB se comunica a la PC. La placa Arduino UNO cuenta con dos LEDs marcados como TX y RX los cuales nos indican cuando la placa está enviando o recibiendo datos. Cuando Arduino UNO se comunica mediante USB a la PC, el LED de RX se enciende para indicar que existe la recepción de datos, de igual forma cuando se transmiten datos el LED de TX se enciende. En Arduino UNO no se puede utilizar el USB y los pines (RX-TX) a la vez.

La comunicación USART puede conectarse a un dispositivo de entrada externa compatible con la USART a un dispositivo de salida o a un microcontrolador, de igual forma Arduino UNO puede comunicarse a la PC a través de un cable USB.

En la IDE de Arduino para establecer comunicación con otro dispositivo, no es necesario instalar una librería en específico, ya que, podemos hacer uso de funciones con las cuales haremos la transmisión y recepción de datos, las cuales son:

- Serial.begin ()
  Esta función establece la velocidad de datos en bits por segundo para la transmisión de datos en serie. Para poder comunicarse con otro equipo se pueden ocupar diferentes velocidades tales como: 300, 600, 1 200, 2 400, 4 800, 9 600, 19 200, 28 800, 38 400, 57 600 o 115 200 bps.
- Serial.println ()
  Esta función imprime los datos en el puerto serie como texto ASCII seguido de un caracter de retorno de carro y devuelve el número de bytes escritos.
- Serial.print ()
  Esta función imprime datos en el puerto serie como texto ASCII. Los números se imprimen con un caracter ASCII para cada dígito. Los flotantes se imprimen de manera similar como dígitos ASCII con dos decimales predeterminados. Los bytes se envían como un solo caracter.
- Serial.available ()
  Esta función devuelve un entero con el número de bytes disponibles para leer desde el buffer serial.
- Serial.read ()
  Lee los datos de serie entrantes y devuelve el primer byte de datos serie entrantes disponibles.
- Serial.write ()
  Escribe datos binarios en el puerto serie, dichos datos se envían como un byte o una serie de bytes y devuelve el número de bytes escritos. [4], [5]

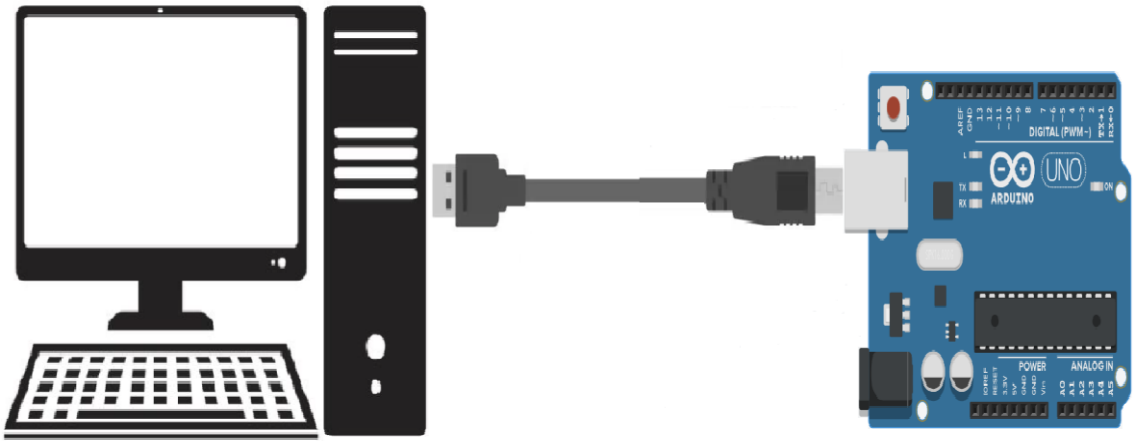


FIGURA 3. 43. PROTOCOLO USART.

3.8.2 Protocolo de comunicación SPI

El protocolo SPI (*Serial Peripheral Interface*, Interfaz Serial Periférica) es la comunicación que se utiliza para conectar uno o más circuitos integrados esclavos a un único dispositivo maestro. Utiliza una comunicación bidireccional entre el transmisor y el receptor, y comparten una fuente de reloj común, la cual requiere de una línea de reloj adicional entre el transmisor y el receptor. En comparación con USART la SPI maneja una mayor velocidad de transmisión de datos.

El protocolo SPI tiene un registro de 16 bits la cual se divide en 8 bits para el transmisor y 8 bits para el receptor. El transmisor es asignado como maestro y es el que proporciona la fuente de reloj para realizar la sincronización con el receptor asignado como esclavo.

Esta comunicación en comparación con I^2C es más rápida, y el tamaño de los mensajes puede ser más grande.

El software de Arduino IDE dispone de una librería para este protocolo llamada SPI. Los pines que utiliza SPI en la placa Arduino UNO son:

- MOSI (*Master Out Slave In*) se utilizada para enviar datos del maestro al esclavo.
- MISO (*Master In Slave Out*) se utiliza para enviar datos del esclavo al maestro.
- SCK (*Serial Clock*) señal del reloj.
- SS o CS (*Slave Select*) línea para activar al esclavo.
- GND todos los dispositivos conectados deben compartir la misma GND.

Arduino UNO dichos pines son 10 (SS), 11 (MOSI), 12 (MISO) y 13 (SCK), de igual forma se pueden tomar del conector *header* (ICSP) de la placa para tener una programación más directa en el chip ATmega328. Este header se muestra en la figura 3.44.

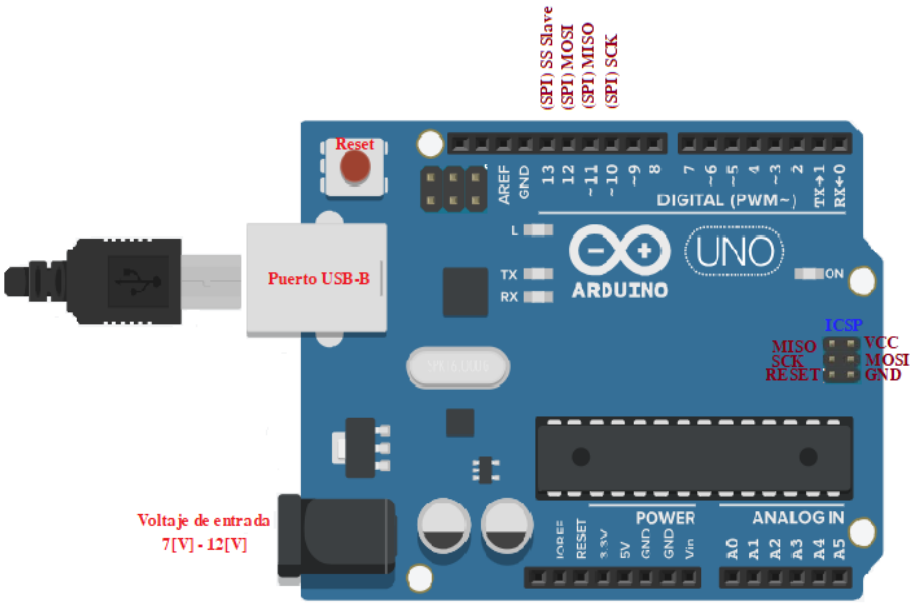


FIGURA 3. 44. PINES QUE SE UTILIZAN EN EL PROTOCOLO SPI.

3.8.3 Protocolo de comunicación I^2C /TWI

I^2C es el Protocolo de Conexión de Interfaz por Bus en Serie, este protocolo utiliza solo dos cables llamados SDA (datos serie) y SCL (reloj serie). El protocolo I^2C funciona a base de una notificación de recibido, esto quiere decir que el transmisor comprueba si el receptor ha recibido los datos correctamente. I^2C trabaja en dos modos:

- Modo maestro.
- Modo esclavo.

Para realizar la comunicación uno de los dispositivos tiene que trabajar en modo maestro, dicho dispositivo se encarga de iniciar la comunicación con el dispositivo que trabajará en modo esclavo, para ello es necesario conocer la dirección del dispositivo esclavo, ya que el dispositivo esclavo responde al dispositivo maestro cuando es direccionado. En I^2C se tiene una dirección única de 7 o de 10 bits, para acceder a los dispositivos esclavos, el dispositivo maestro debe conocer esta dirección única.

En Arduino la librería para I^2C se llama *Wire*, los pines que utiliza Arduino UNO para el puerto I^2C/ TWI son los pines A4 (SDA) y A5 (SCL).

El I^2C habilita conectar simultáneamente hasta 128 diferentes dispositivos, teniendo un máximo de frecuencia de reloj de 3.4 [MHz] (de alta velocidad), pero la frecuencia mínima que se utiliza es de 100 [KHz].

Características	SPI	I^2C
Nombre	Serial Peripheral Interface	Inter-Integrated Circuit
Tipo de comunicación	Síncrona	Síncrona
Número de líneas	4 (SCLK, MOSI, MISO, SS)	2 (SCL, SDA)
Topología de conexión	Punto a punto o multidireccional	Multidireccional
Velocidad de transmisión	Alta	Baja a moderada
Comunicación	Comunicación f*ull-duplex.*	Comunicación h*alf- duplex.*
Dirección de dispositivos esclavos	No está estandarizada	7 bits (extendido a 10 bits)
Implementación	Más compleja	Menos compleja

TABLA 3. 12. COMPARATIVA ENTRE COMUNICACIONES SPI E I2C.

Protocolo	UART	I2C	SPI
Complejidad	Fácil	Conexión de múltiples dispositivos	Se vuelve complejo a medida que aumentan los dispositivos.
Velocidad	Lento	Más rápido que UART	El más rápido de los 3.
Número de dispositivos	2	127	Varios, lo cual lo hace complejo.
Número de líneas	1	2	4
Tipo de comunicación	*full-duplex*	*half-duplex*	*full-duplex.*
Admisión de maestro/esclavo.	Solo uno a la vez	Varios maestros y esclavos	Un maestro y varios esclavos.
Direccionamiento	Sí	Sí	No

TABLA 3. 13. COMPARACIÓN DE PROTOCOLOS UART, I2C Y SPI.

3.8.3.1 Comunicación I^2C en Arduino Uno

Para lograr una comunicación I^2C en Arduino Uno basta con utilizar las dos entradas analógicas marcadas con los nombres de A4 y A5.

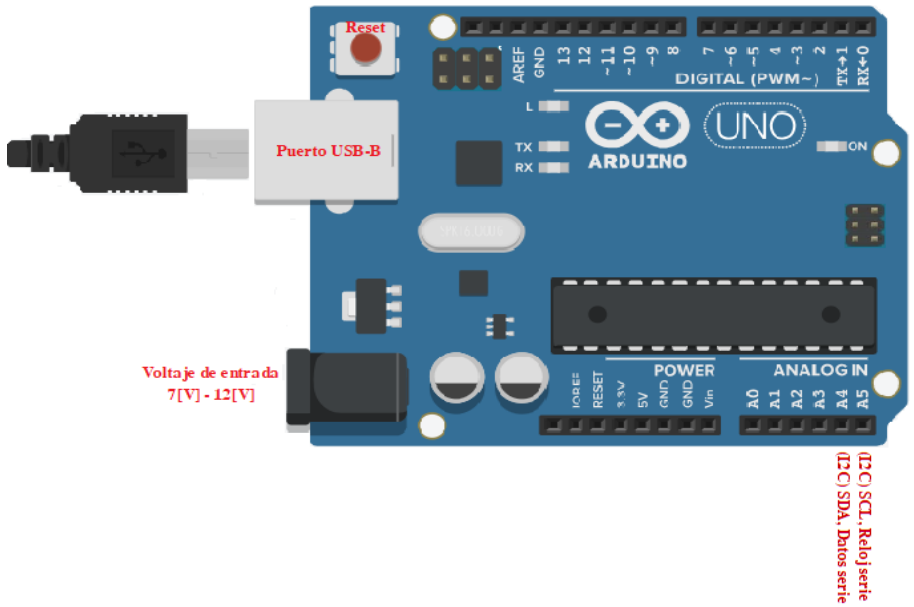


FIGURA 3. 45. ENTRADAS PARA LA COMUNICACIÓN I^2C EN LA TECNOLOGÍA ARDUINO UNO A4 Y A5.

Pero también se pueden utilizar los pines que están marcados específicamente para la comunicación I^2C en la tecnología Arduino UNO, lo cual nos permite tener disponibles los pines de A4 y A5 como entradas/salidas, en caso de ser necesario.

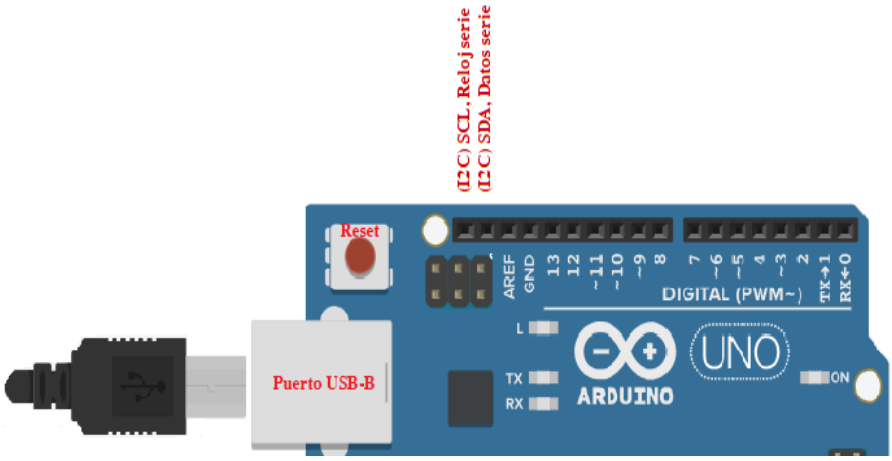


FIGURA 3. 46. ENTRADAS I^2C ESPECIFICAS EN LA TECNOLOGÍA ARDUINO UNO.

Haciendo uso de dos placas Arduino UNO realizamos una comunicación I^2C mediante la cual las dos placas de Arduino UNO serán conectadas mediante los pines A4 y A5; en esta prueba, la primer placa, Arduino UNO actuará como el dispositivo Maestro, mientras que la segunda placa actuará como el dispositivo Esclavo.

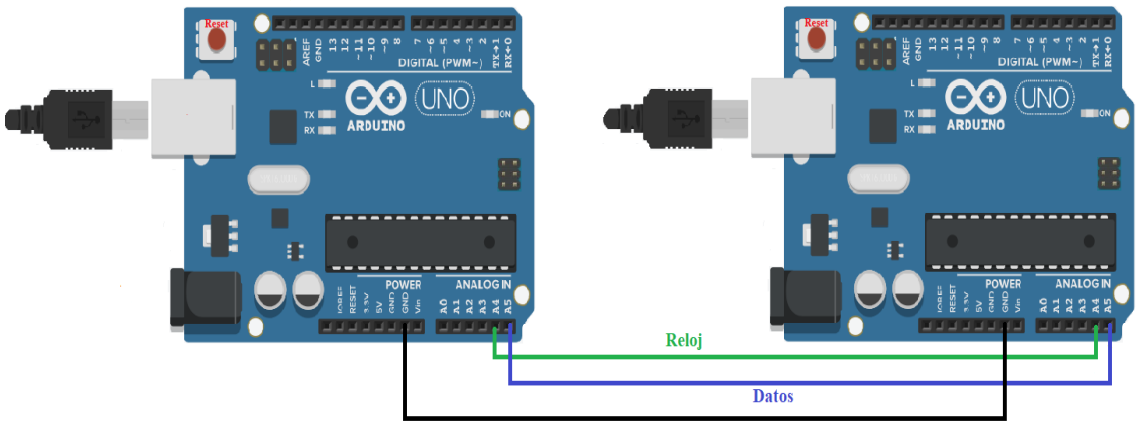


FIGURA 3. 47. DIAGRAMA DE CONEXIÓN ENTRE LOS ARDUINO UNO PARA UNA COMUNICACIÓN I2C.

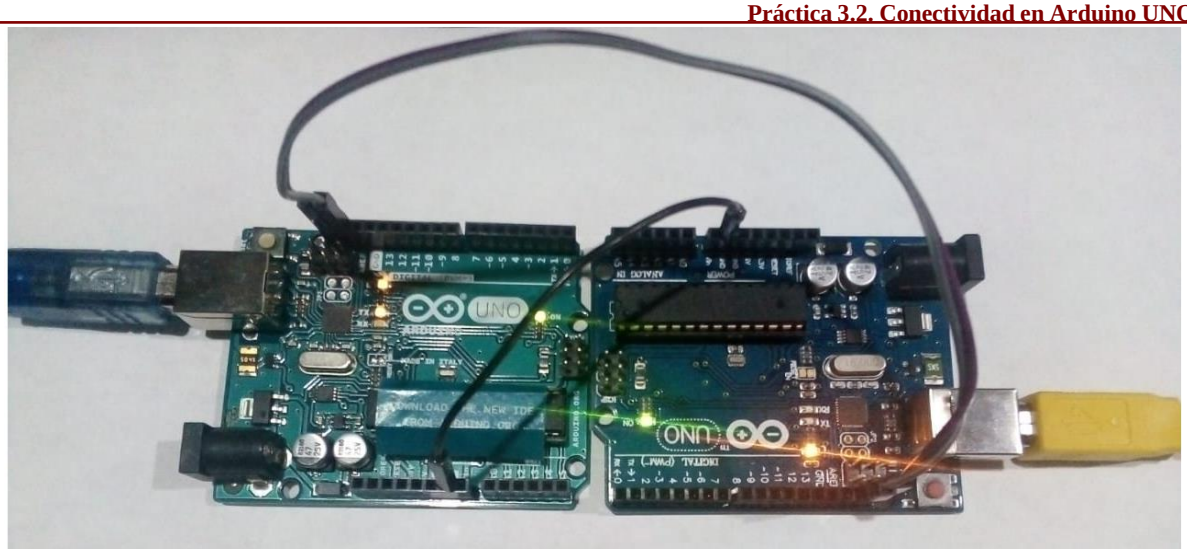


FIGURA 3. 48. CONEXIÓN EN COMUNICACIÓN I2C.

Para cargar los programas en los dos dispositivos Arduino UNO se puede realizar al mismo tiempo, solo que se debe de tener cuidado al cargar el programa en las placas, puesto que se tienen diferentes puertos COM.

En nuestro caso utilizaremos el Puerto COM3 para el dispositivo Maestro y el Puerto COM4 para el dispositivo Esclavo. Para visualizar lo que se obtiene del dispositivo Esclavo en el dispositivo Maestro, abrimos el monitor serial del COM3 como podemos apreciar en la figura. 3.50, donde trabajamos con una velocidad de 9600 s, al aumentar o disminuir la velocidad en el monitor serial COM3 podemos notar que el mensaje se deforma como se muestra en la figura. 3.51. La deformación del mensaje ocurre debido a la necesidad de mantener una coherencia en la velocidad de transmisión para garantizar una comunicación adecuada. Si la velocidad establecida en el código de Arduino difiere de la velocidad seleccionada en el monitor serial COM, los datos se interpretarán de manera errónea, lo que inevitablemente ocasionará la distorsión del mensaje.

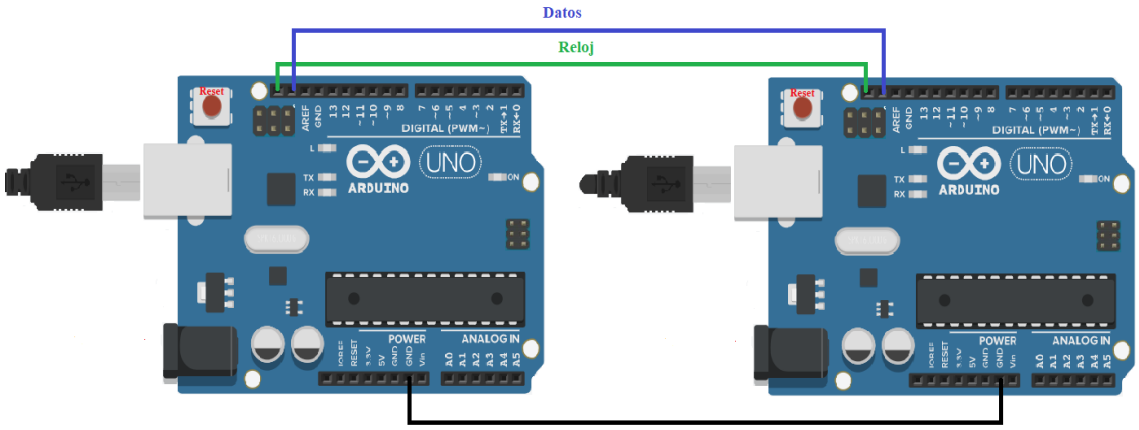


FIGURA 3. 49. DIAGRAMA DE CONEXIÓN UTILIZANDO LOS PINES SCL Y SDA.

Esta conexión se puede realizar también utilizando los pines SCL y SDA, como se muestra en la figura. 3.49, funciona de la misma forma que utilizando los pines A4 y A5.

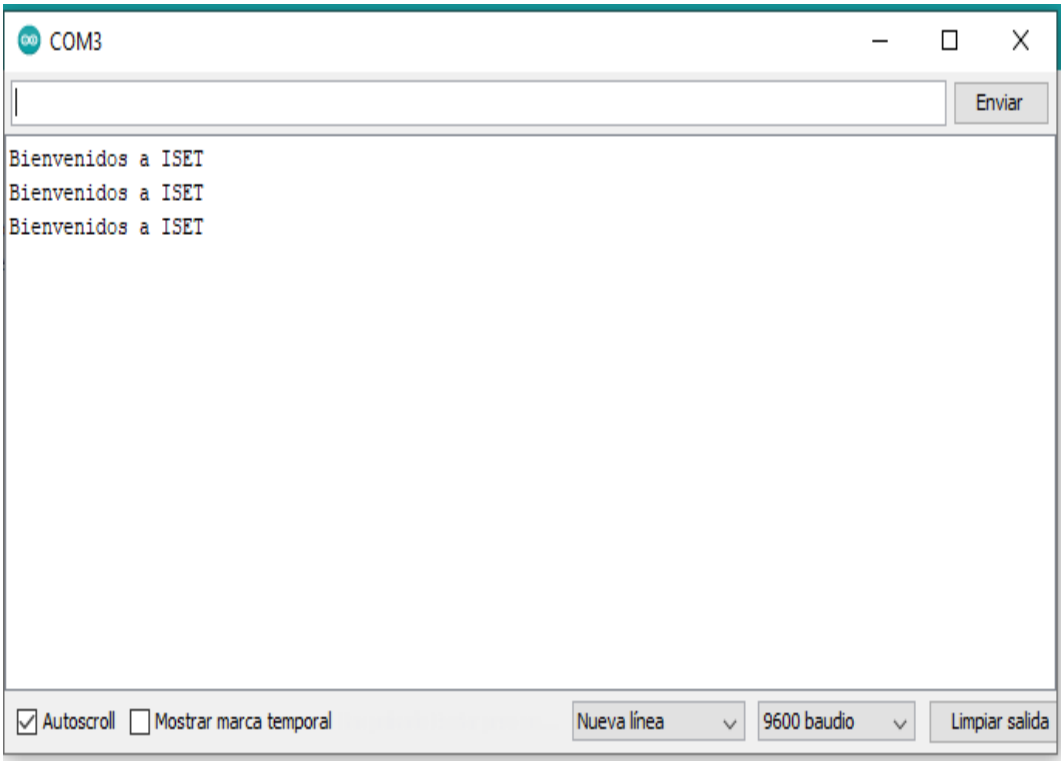


FIGURA 3. 50. MONITOR SERIAL DEL ARDUINO ESCLAVO.

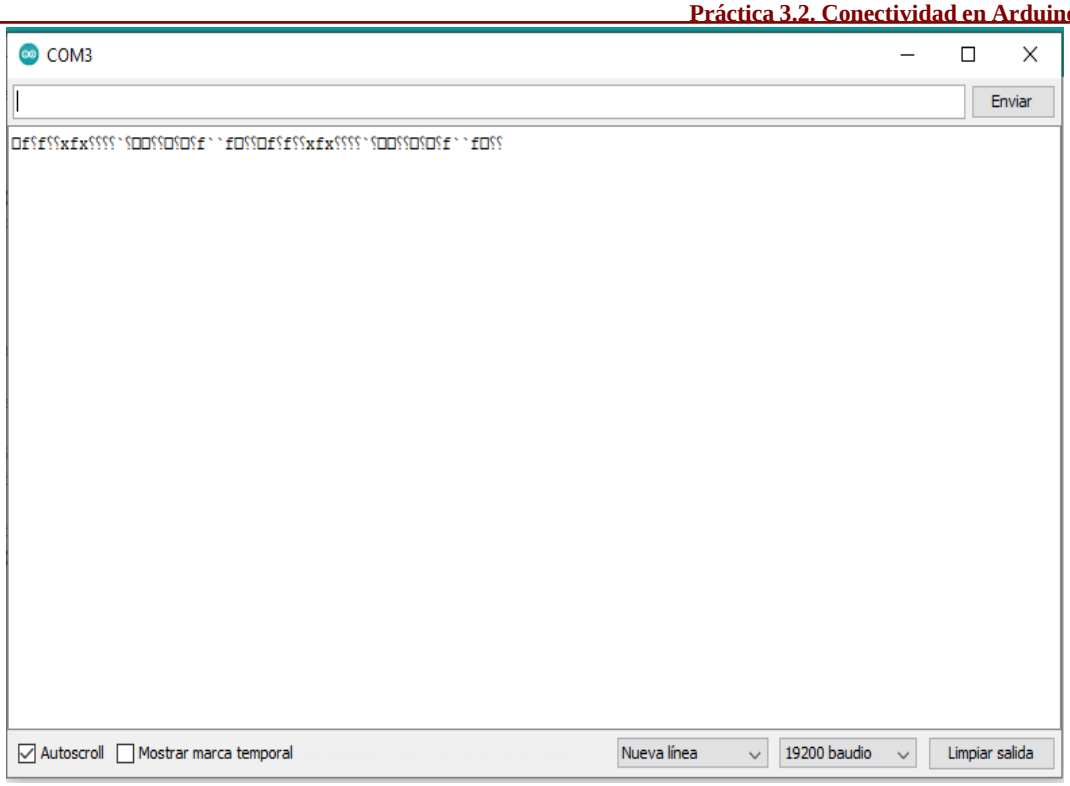


FIGURA 3. 51. MONITOR SERIAL DEL ARDUINO ESCLAVO CON MENSAJE DEFORMADO.

Una de las ventajas de este tipo de comunicación es que podemos conectar varios dispositivos Arduino UNO para que trabajen como esclavo a un Arduino UNO maestro, como podemos observar en la figura. 3.52.

De igual forma que en la conexión anterior se debe de tener cuidado al momento de cargar los programas en cada placa pues cada una tiene un COM diferente.

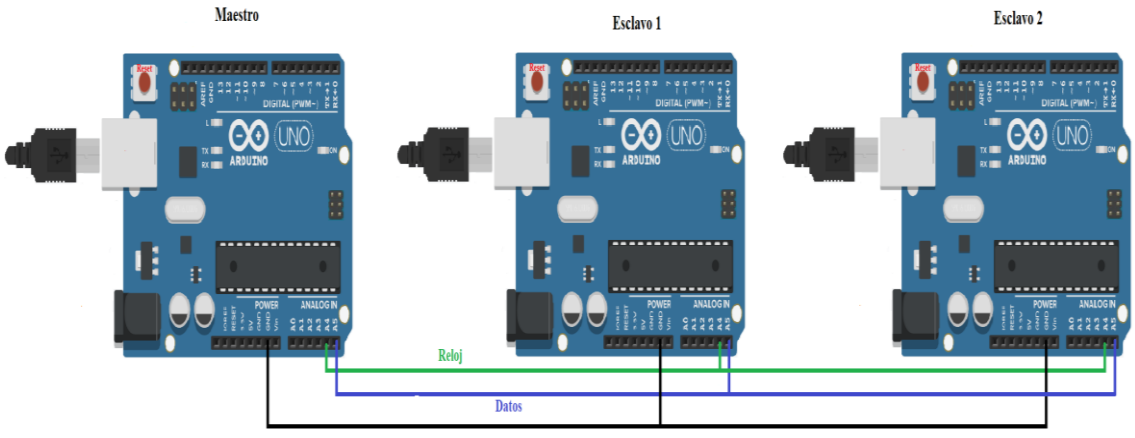


FIGURA 3. 52. DIAGRAMA DE UN ARDUINO MAESTRO Y DOS ARDUINO UNO ESCLAVOS.

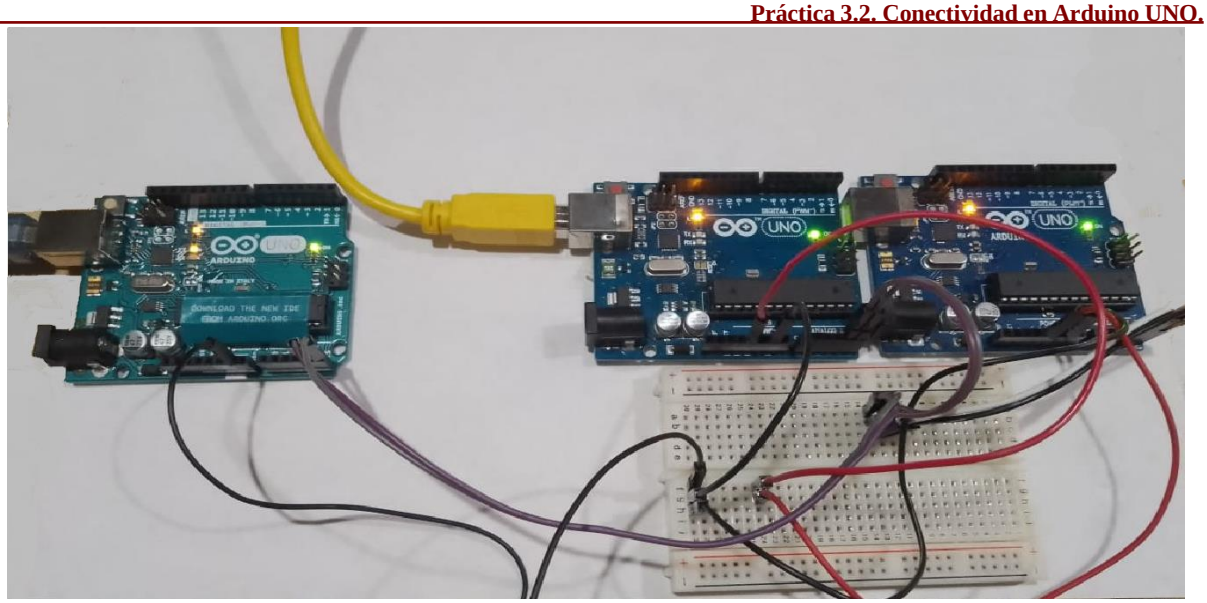


FIGURA 3. 53. CONEXIÓN CON 3 ARDUINO UNO.

En este caso se utilizan el puerto COM3, COM4 y COM5, para cargar los diferentes programas a la placa como se muestra en la figura. 3.54.

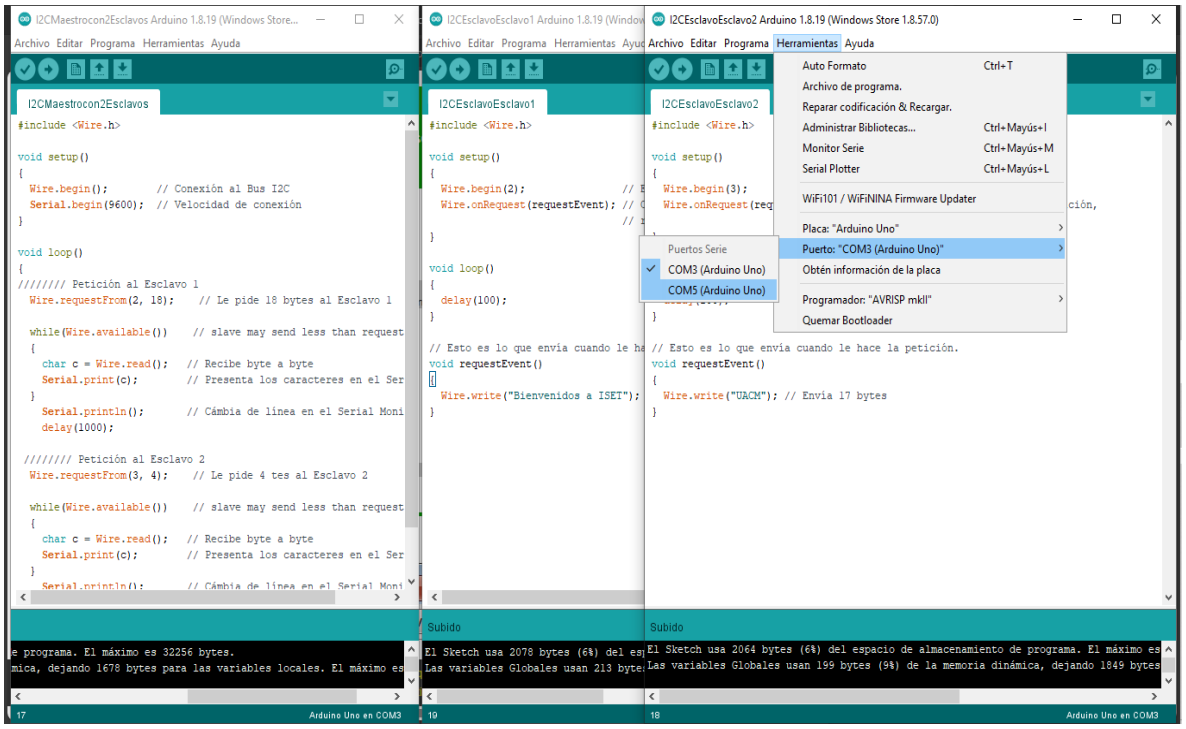


FIGURA 3. 54. CONSOLA DE PROGRAMACIÓN DE ARDUINO.

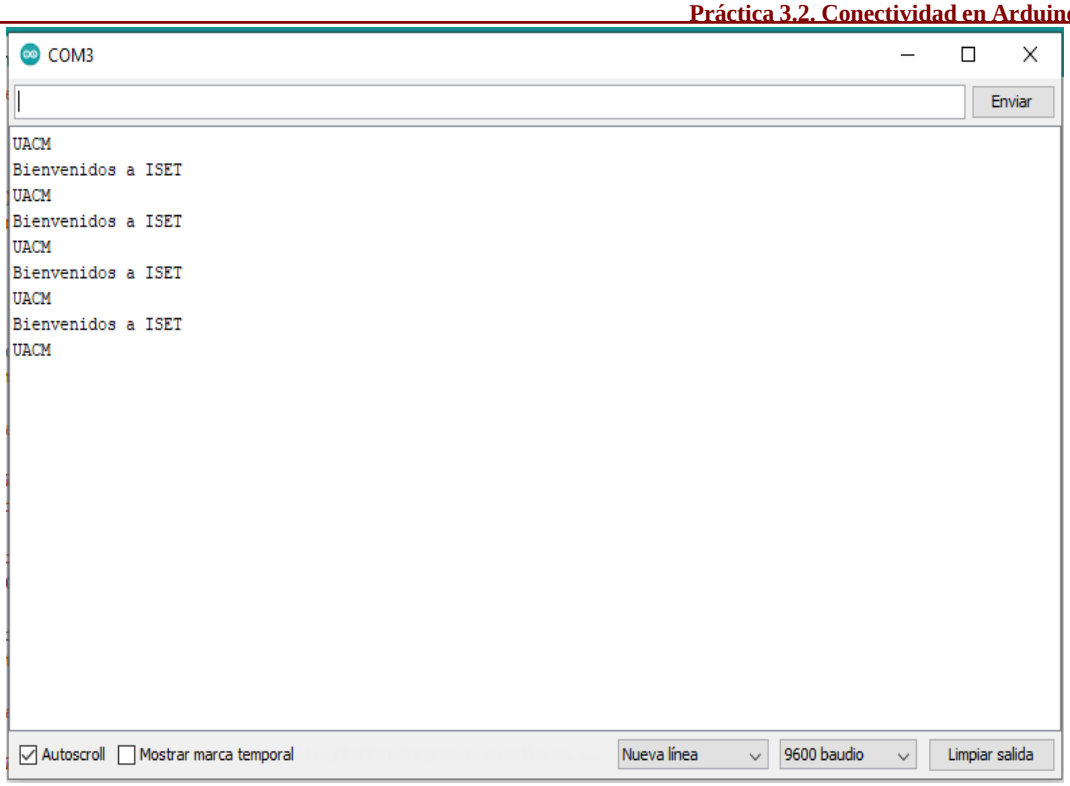


FIGURA 3. 55. MONITOR SERIAL DEL DIAGRAMA DE UN ARDUINO MAESTRO Y DOS DISPOSITIVOS ESCLAVOS.

Este tipo de comunicación es empleado en domótica, ya que, el dispositivo que funciona como maestro está en constante comunicación con sus esclavos, los cuales solo reaccionan cuando el dispositivo maestro solicita realizar alguna acción o recopilar datos de determinado sensor, el encendido o apagado de algún elemento, etc.

3.9 Práctica 3. Conectividad en Raspberry Pi 4

Equipo y material empleado

- Raspberry Pi 4.
- Protoboard.
- Jumper macho-macho.
- Conversor lógico bidireccional de 8 canales.

3.9.1 Protocolo de comunicación UART

UART (*Universal Asynchronous Receiver-Transmitter*, Universal Asincrónico Receptor-Transmisor) también llamado serial, usa 2 pines de conexión una para datos y uno de recepción de datos, se requiere que la tierra sea igual en el receptor como en el emisor.

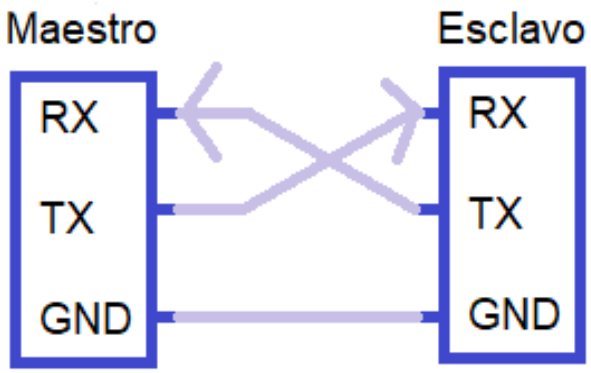


FIGURA 3. 56. ESQUEMA DE COMUNICACIÓN UART.

Características:

- Envía 8 bits.
- Velocidad de transmisión, puede transmitir hasta 1 [Mbps].
- El tipo de comunicación puede ser half-duplex o full-duplex debido a que tiene un pin de recepción de datos Rx y otro de transmisor de datos Tx.
- Es un protocolo de comunicación asíncrono, no tiene señal de reloj, esto implica que el emisor y el receptor deben de tener la misma velocidad de lectura entre las cuales la más común es de 9600 bps.

El puerto serie ocupa los pines GPIO14 (TxD) y GPIO15 (RxD). Para poder utilizar los pines GPIO14 y GPIO15 como pines de entrada/salida digital la opción serial debe estar desactivada.

3.9.2 Protocolo de comunicación SPI

SPI (Interfaz Periférica Serial) es un protocolo serie, la dirección de los dispositivos se trasmite utilizando pines específicos para seleccionarlos. Cuenta con un maestro que toma el rol activo en la comunicación y uno o varios esclavos que asumen el rol de pasivo.

Usa cuatro pines, uno para envío de datos, otro para recepción de datos, otro para una señal de reloj y un adicional es un habilitador.

Pines que utiliza para la comunicación SPI

- **SCL** (*Serial CLock*). Señal de reloj.
- **MISO** (*Master In, Slave Out*). Entrada de datos para el maestro, salida de datos para el esclavo.
- **MOSI** (*Master Out, Slave In*). Salida de datos para el maestro, entrada de datos para el esclavo.
- **CEn** (*Chip Enable*). Una o varias señales de selección de destino activas a nivel bajo. Para enviar o recibir del dispositivo "cero" se activa la señal CE0, para el dispositivo "uno" se activa CE1, etc.

Características

- Velocidad de transmisión, puede transmitir hasta 60 [Mbps].
- El tipo de comunicación es full-duplex, es decir puede enviar y recibir datos al mismo tiempo, esto porque tiene un pin dedicado a cada una de estas funciones.
- El pin para enviar datos es conocido como MOSI.
- El pin para recibir dados es conocido como MISO.
- El tipo de comunicación es síncrono, ya que tiene una señal para el reloj.

La Raspberry Pi cuenta con tres interfaces maestro SPI (SPI0, SPI1 y SPI2) aunque solo una de ellas (SPI0) es visible para los modelos originales y dos de ellas (SPI0 y SPI1) para los modelos con conector J8 de 40 pines. Cada una de estas interfaces cuenta con dos líneas de selección de destino (CE0 y CE1). De ellas SPI0 es más evolucionada, puesto que permite DMA (acceso directo a memoria). SPI0 está diseñada para transferencias de alta velocidad (reloj de hasta 125 [MHz]) sin producir una carga significativa para el procesador. Sin embargo, SPI1 no tiene posibilidad de usar DMA y solo dispone de una pequeña FIFO (First In, First Out) de cuatro palabras de 32 [bits].

Configuración de parámetros para la comunicación con un dispositivo SPI:

- La polaridad de la señal *chip select* (**CSPOL**). Normalmente es activa a nivel bajo y no necesita modificarse.
- La polaridad del reloj (CPOL). Un valor 0 significa que el nivel de descanso del reloj es bajo. Un valor 1 significa que el nivel de descanso del reloj es alto.

- La fase del reloj (CPHA). Si tiene un valor 0 significa que las transiciones de SCLK ocurren en la mitad de cada bit de datos transmitidos. Un valor 1 significa que las transiciones de SCLK ocurren al principio de cada bit.
- Frecuencia del reloj. Se configura seleccionando una fuente de reloj y un divisor (CDIV). La fuente de reloj normalmente es de 125 [MHz].

Para realizar la activación del protocolo de comunicación SPI se debe realizar lo siguiente:

Abrir la terminal y escribir el comando: ***sudo raspi-config***.

Una vez que se abra la siguiente ventana, elegimos el protocolo SPI, para activarlo damos clic en aceptar y reiniciamos Raspberry Pi.

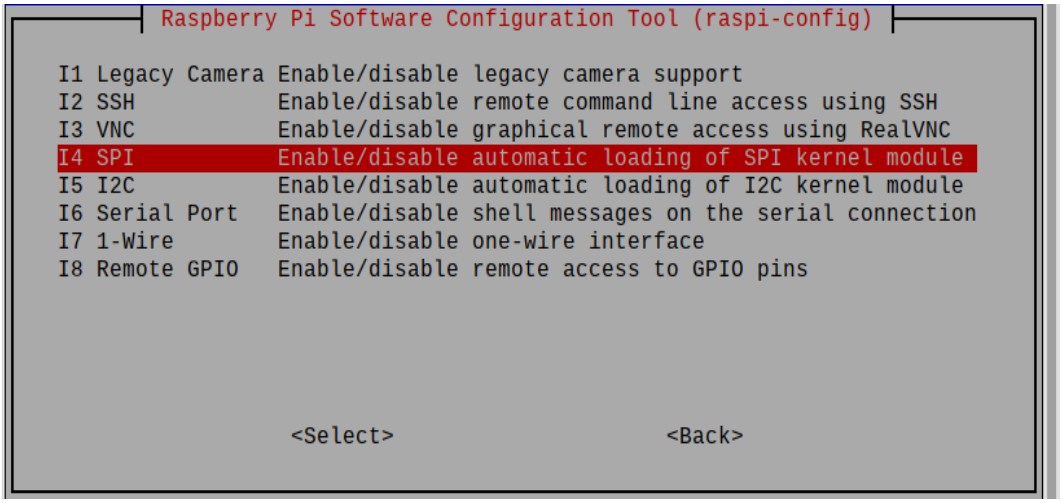


FIGURA 3. 57. VENTANA DE CONFIGURACIÓN DE RASPBERRY PI.

3.9.3 Protocolo de comunicación I^2C

La comunicación I^2C (*Inter Integrated Circuit*, (es un protocolo de una línea de datos) (SDA) y una línea de reloj (SCL). Se pueden realizar transmisiones serie bidireccionales de hasta 100 [kbit/s] en modo estándar y 400 [kbit/s] en modo rápido. Las versiones más modernas de I^2C incorporan modos con mayores tasas de transferencia (hasta 5 [Mbit/s]) pero el procesador de la Raspberry Pi no los implementa.

Características de la comunicación I^2C

- Se implementa con dos señales digitales, la señal de datos SDA y la señal de reloj SCL.
- Dado que la configuración del bus es de colector abierto, usan resistencias *pull up*.
- Velocidad de transmisión, puede transmitir hasta 3 [Mbps], sin embargo, lo más común son velocidades bajas en modo estándar 100 [kHz] y en modo de alta velocidad 400 [kHz].
- Comunicación *half-duplex*, solo se puede transmitir un dato a la vez ya sea de lectura o de escritura y esto porque solo tiene una línea para datos SDA.

- Está diseñado para conectar varios dispositivos al mismo bus de datos a un número máximo de 127 cuando se maneja direccionamiento de 7 bits.
- El tipo de comunicación es síncrono, ya que tiene una señal para el reloj.

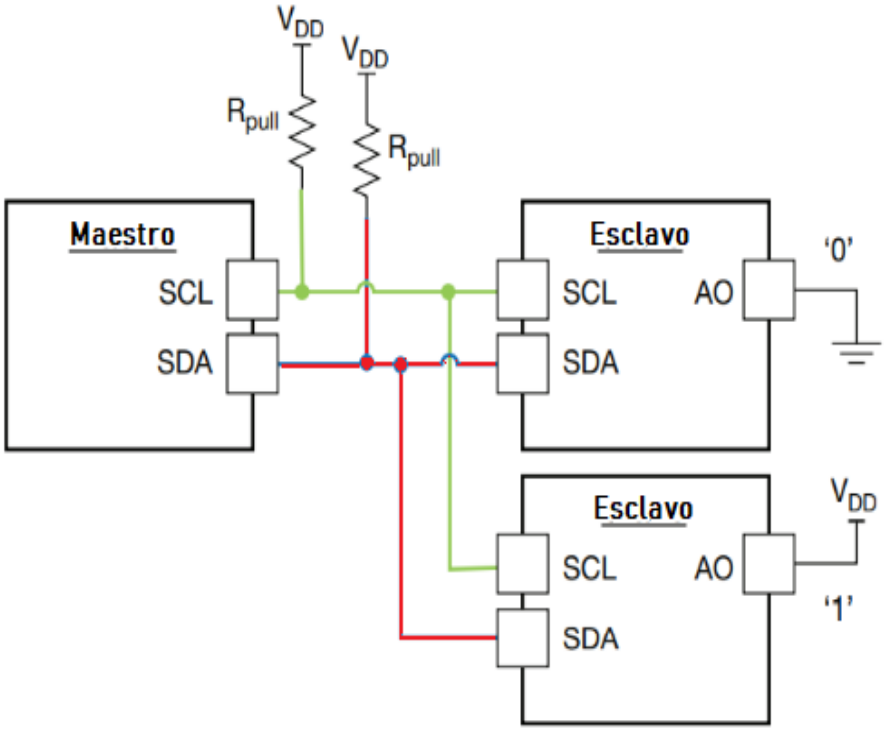


FIGURA 3. 58. CONFIGURACIÓN I^2C.

La Raspberry Pi dispone de dos periféricos para implementar I^2C, el BSC (Broadcom Serial Controller) que implementa el modo maestro, y el BSI (Broadcom Serial Interface) que implementa el modo esclavo.

El BSC implementa tres maestros independientes que tienen que estar en buses I^2C separados (no permite multi-maestro).

- BSC0 se reserva para la identificación de las placas de expansión (especificación HAT, pines 27 y 28).
- BSC2 es de uso exclusivo para la interfaz HDMI.
- Se utiliza BSC1.

Para establecer la comunicación con múltiples dispositivos en un mismo puerto, cuenta con una dirección, usada para llamar al dispositivo con el que se quiere establecer comunicación. Esta dirección puede ser de dos tipos, de lectura o de escritura.

Escritura: el maestro va escribir un dato sobre este dispositivo esclavo.

Lectura: es porque va a leer un dato proveniente del dispositivo esclavo.

La dirección es de 8 bits. En la dirección, los bits del 7 al 1 son usados como número asignado al dispositivo y el bit 0 se usa como identificador de lectura o escritura, siendo un 0 para escritura y un 1 para lectura.

En Raspberry Pi la señal SDA se encuentra en el GPIO 2 y la señal SCL en GPIO 3.

Para realizar la activación del protocolo de comunicación I^2C se debe realizar lo siguiente:

Para usar la interfaz I^2C de Raspberry Pi, debe estar habilitada. Para ello, se introduce el siguiente comando en la terminal: ***sudo raspi-config***

En el menú, seleccione I5- Opciones de interfaz y luego I5 I^2C y valide.

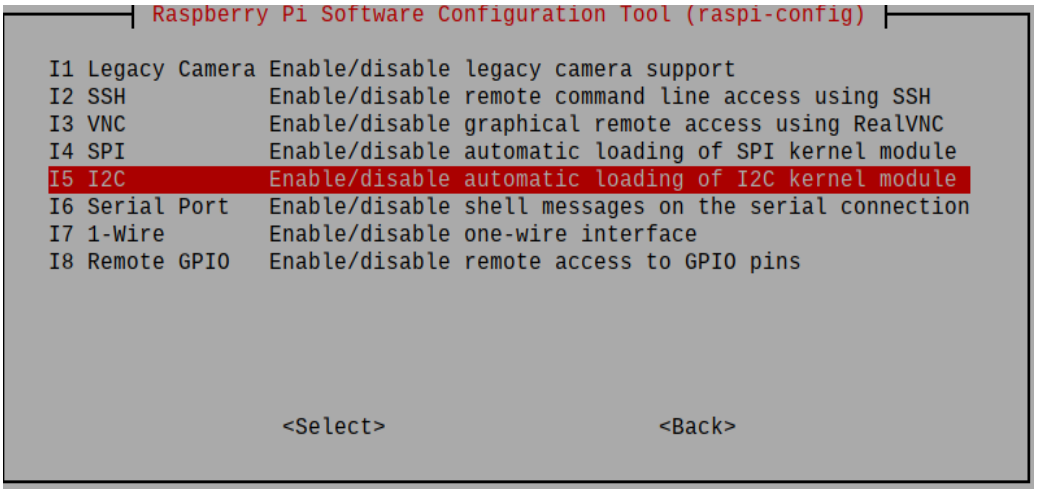


FIGURA 3. 59. VENTANA DE CONFIGURACIÓN DE RASPBERRY PI.

Una vez realizada la conexión, puede comprobar los dispositivos conectados al bus tecleando el siguiente comando: ***sudo i2cdetect -y 1***

La Raspberry Pi devuelve la lista direcciones detectadas en el bus.

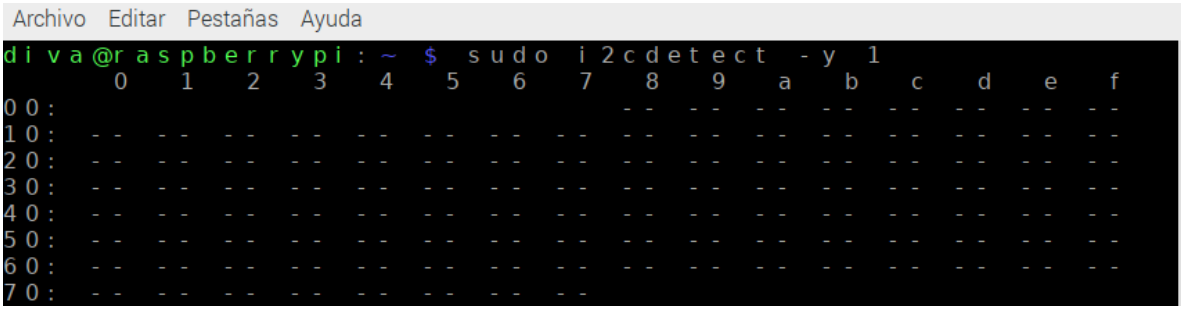


FIGURA 3. 60. VENTANA PARA COMPROBAR LOS DISPOSITIVOS CONECTADOS AL BUS I^2C.

Para establecer la comunicación I^2C entre Raspberry Pi y Arduino, necesitamos conectar físicamente el bus que utiliza 3 pines. Una comunicación I^2C se define por un bus de dos cables y una dirección. Los pines utilizados para la comunicación I^2C suelen estar fijos para cada dispositivo. Una en la que se envían los datos (Línea de datos en serie SDA) y otra en la que se envía el reloj de sincronización (Línea de reloj en serie SLC). Los fundamentos de las dos tecnologías deben estar conectados para establecer una referencia potencial común.

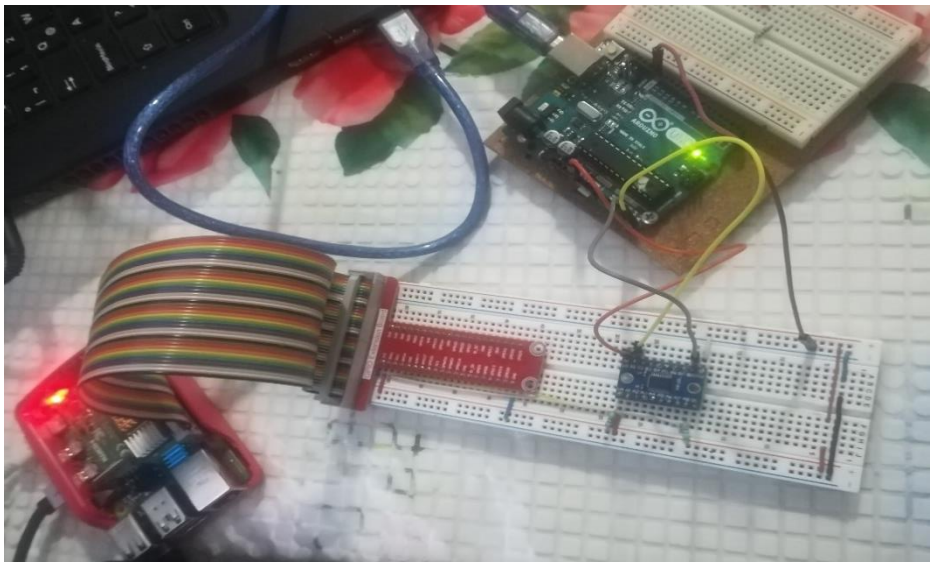


FIGURA 3. 61. COMUNICACIÓN **I^2C** ENTRE RASPBERRY PI 4 Y ARDUINO UNO.

Referencias de Práctica 3. Conectividad

PIC

[1] García, Eduardo. (2009). Compilador C CCS y simulador Proteus para microcontroladores PIC. México: Alfaomega Grupo Editor.

[2] Verle, M. (2009). PIC Microcontrollers: Programming in C. Mikroelektronika.

[3] Daniel Benchimol, Microcontroladores. Funcionamiento, Programación y Aplicaciones Prácticas, USERS -.

Arduino

[1] Aldea, E. L. (2016). Arduino. Guía práctica de fundamentos y simulación: Guía práctica de fundamentos y simulación. Ra-Ma Editorial.

[2] Benchimol, Daniel. (2011). Microcontroladores. Funcionamiento, programación y usos prácticos. Buenos Aires: Manual Users.

[3] Aliverti, Paolo. (2019). Arduino. Trucos y secretos: 120 ideas para resolver cualquier problema. Marcombo

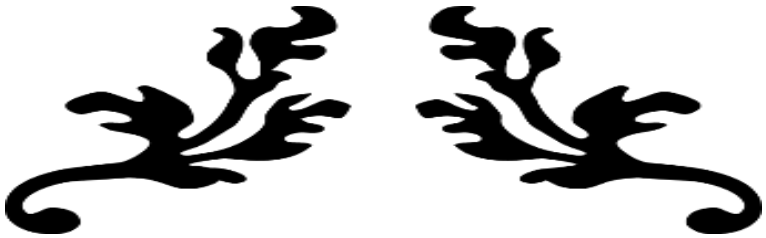
[4] Arduino - home. (s. f.). https://www.arduino.cc/

[5] Ganazhapa, Byron. (2016). Arduino. Guía Práctica. México: Alfaomega Grupo Editor S.A de C.V.

[6] Barrett, Steven. (2013). Arduino Microcontroller Processing for Everyone! Third Edition. Morgan & Claypool Publishers.

Raspberry Pi

[1] Michael R. Sweet, Guía de programación en serie para sistemas operativos POSIX, 5.ª ed., 1994-99, en http://www.cmrr.umn.edu/~strupp/serial.ht



Capítulo 4

Análisis de resultados

	PIC 16F887	Arduino UNO	Raspberry Pi 4
Hardware	Cerrado	Abierto	Cerrado
Software	MikroC PRO for PIC	Arduino IDE	Broadcom BCM2711
Pines	40	14	40
Memoria ROM	8 [kB] Flash	32 [kB] Flash	No
Memoria RAM	368 [bytes]	2 [kB]	2 [GB]
Memoria EEPROM	256 [bytes]	1 [kB]	No
Resolución	10 [bits]	8 [bits]	8 [bits]
Voltaje	2 [V] - 5.5 [V]	7 [V] – 12 [V]	3.3 [V] y 5 [V]
Corriente	220 [µA]	20 [mA]	3 [A]
Comunicación serie	UART I2C SPI	UART I2C (TWI) SPI	UART I2C SPI
Wi-Fi	No	No	802.11ac de 2.4 [GHz] y 5.0 [GHz]
Bluetooth	No	No	5.0
Sistema operativo	No	No	Raspberry Pi OS (Debian)
SD	No	No	Si

TABLA 4. 1. COMPARATIVA DE CARACTERÍSTICAS BÁSICAS DE LAS TECNOLOGÍAS. PIC 16F887, ARDUINO UNO Y RASPBERRY PI 4

Las placas PIC 16F887, Arduino UNO y Raspberry Pi 4 tienen diferentes características, se exponen en la Tabla 4.1, donde se hace una comparación de las tecnologías.

Plataforma	Tamaño de datos	Arquitectura interna	Arquitectura de procesador
PIC 16F887	8 bits	Harvard	Arquitectura RISC (35 instrucciones)
Arduino UNO	8 bits	Harvard	Arquitectura RISC (35 instrucciones)
Raspberry Pi 4	64 bits	Von Neumann	Arquitectura RISC (32 instrucciones)

TABLA 4. 2. COMPARATIVA DE LA ARQUITECTURA DE LAS TECNOLOGÍAS PIC16F887, ARDUINO UNO Y RASPBERRY PI 4.

Hardware de la Plataforma	OS	Lenguaje
PIC 16F887	Ninguno	Lenguaje C
Arduino UNO	Ninguno	Lenguaje C++
Raspberry Pi 4	GNU/Linux ARM (Debían, Raspberry Pi OS), Fedora, Arch Linux, RISC OS)	Java Ruby Perl Python

TABLA 4. 3. COMPARATIVA DEL OS Y LENGUAJE QUE UTILIZAN LAS TECNOLOGÍAS PIC 16F887, ARDUINO UNO Y RASPBERRY PI 4.

Plataforma	Número total de pines		Pines disponibles
PIC 16F887	40	35	Propósito general
			Bidireccional
			Multifuncional
Arduino UNO	14		Propósito general
Raspberry Pi 4	40	28	Propósito general
			Bidireccional
			Multifuncional

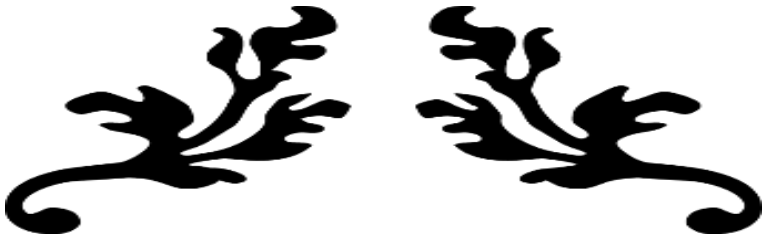
TABLA 4. 4. COMPARATIVA DE LOS PINES DISPONIBLES EN LAS PLACAS PIC 16F887, ARDUINO UNO Y RASPBERRY PI 4.

Plataforma	PWM	
	Pines	Resolución
PIC 16F887	16 y 17	10 bits
Arduino UNO	3, 5, 6, 9, 10 y 11	8 bits
Raspberry Pi 4	12, 13, 18 y 19	8 bits

TABLA 4. 5. PINES DISPONIBLES PARA LA FUNCIÓN PWM EN LAS TECNOLOGÍAS PIC 16F887, ARDUINO UNO Y RASPBERRY PI 4.

Plataforma	UART		SPI		I^2C	
	Pines	Velocidad de transmisión	Pines	Velocidad de transmisión	Pines	Velocidad de transmisión
PIC 16F887	25 y 26	9600 [bps] 19200 [bps] 38400 [bps] 57600 [bps] 115200 [bps]	18, 23 y 24	100 [kbps] 1 [Mbps] 10 [Mbps]	18 y 23	100 [kHz] 400 [kHz] 1 [Hz]
Arduino UNO	0 y 1	1200 [bps] 2400 [bps] 4800 [bps] 9600 [bps] 19200 [bps] 38400 [bps] 57600 [bps] 115200 [bps]	10, 11, 12 y 13	10 [Mbps]	A4 y A5	100 [kHz] 400 [kHz] 3.4 [Mbps]
Raspberry Pi 4	8 y 10	1 [Mbps]	19, 21 y 23	60 [Mbps]	3 y 5	100 [kbit/s] 400 [kbit/s] 3 [Mbps]

TABLA 4. 6. COMPARACIÓN DE COMUNICACIÓN ENTRE LAS TECNOLOGÍAS PIC 16F887, ARDUINO UNO Y RASPBERRY PI 6



Conclusiones

Concluimos que para el Internet de las Cosas la opción más recomendable es usar Raspberry Pi.

Por los siguientes puntos:

PIC, Arduino y Raspberry Pi son tecnologías muy similares en funcionalidad y composición. Las tres tienen entradas y salidas, con las tres se pueden desarrollar aplicaciones, pero la manera en la que cada una realiza estas funciones es donde está su mayor diferencia.

Raspberry Pi genera compatibilidad con PIC y Arduino. Pero para un mejor funcionamiento es necesario cambiar el voltaje de su fuente principal para cubrir con el voltaje requerido en los componentes de las tecnologías PIC y Arduino.

Raspberry Pi es un microprocesador que cuenta con la característica de tener puertos de entrada y salida de propósito general.

En sentido de la conectividad. Raspberry cuenta con conectividad Bluetooth, WIFI y Ethernet integradas y aunque a PIC y Arduino también se les puede añadir conectividad esto representaría agregar placas de expansión, por lo que gastaríamos parte de sus puertos, lo que implica consumo de recursos, además de que si a la arquitectura le agregamos accesorios el manejo de la información se vuelve lento y genera una latencia y para nosotros en telecomunicaciones la latencia es importante.

La Raspberry Pi tiene una mayor capacidad de procesamiento a costa de consumir más corriente. Por lo tanto, es la menos recomendable para proyectos portátiles a menos de que necesite la capacidad de cómputo.

La capacidad de memoria en Raspberry Pi la podemos expandir, lo cual puede ser una ventaja desventaja. Al tener más memoria se permite almacenar más datos y programas, pero al tener más datos implica tener una latencia cuando se leen.

La fortaleza en PIC y Arduino, que son microcontroladores es que están diseñados para realizar una tarea en específico en un menor tiempo, pero si vamos a manejar muchos datos nos conviene utilizar Raspberry Pi.

En cuanto al Software, al encender PIC o Arduino ejecuta inmediatamente la tarea para lo que lo hemos programado, mientras que Raspberry para usarlo requiere un sistema operativo, por lo que es un poco más lento, pues el tiempo de arranque de Raspberry Pi 4 va de 10 a 30 [s] dependiendo del sistema operativo y de los servicios que se estén ejecutando al inicio. Por otra parte, el hecho de tener un sistema operativo hace que la Raspberry Pi sea más adecuada para garantizar la conectividad y la comunicación de los dispositivos.

La principal ventaja que tiene la Raspberry Pi 4 al tener un sistema operativo en comparación con el Arduino Uno y el PIC16F887 es su capacidad para ejecutar un sistema operativo completo como Raspberry Pi OS, lo que le brinda una funcionalidad similar a la de una computadora de escritorio, pero de forma pequeña y de bajo costo.

Por tal motivo la opción más recomendable es usar una Raspberry Pi para IoT debido a que se puede utilizar como una plataforma de servidor propio de modo que se tendrá el control total de los datos.

Podemos utilizar la relación maestro esclavo, donde Raspberry Pi es el maestro y PIC o Arduino esclavos, sin embargo, nos conviene utilizar como esclavo al PIC por ser más robusto y económico, debido a que a Arduino está pensado para la educación.

Una propuesta para retomar esta Tesis es mediante la investigación y aplicación a una de las tecnologías con las placas especializadas en el IoT, puesto que las placas que se analizaron en esta Tesis, son las más comerciales o más comunes.

Mediante estas nuevas tecnologías se podrá tener un ahorro en costos y espacio.

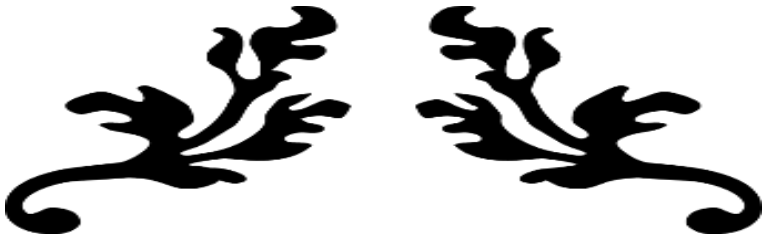
- Existen placas especializadas en el tema de Internet de las Cosas, tales como:
  - La placa Arduino MKR WiFi 1010 de la familia de Arduino cuenta con varios sensores.
  - Arduino MKR NB 1500 donde integra Wifi, GSM, LoRa y SigFox.

- También se cuentan con módulos especializados como:
  - Arduino MKR ETH Shield
  - Arduino MKR Vidor 4000
  - Arduino UNO WiFi Rev.2.

- Por otro parte la tecnología de Raspberry Pi a inicios de este año introdujo la nueva placa Raspberry Pi 5, la cual cuenta con una fuente de alimentación USB-C de 5[V] y de 5 [A], esta placa funciona con un enfriamiento activo y una de las mejoras en esta placa es la implementación de un botón de on/off, el cual es importante para encender y apagar Raspberry Pi de una forma conveniente y rápida, sin necesidad de desconectarla de manera física de la fuente de alimentación y de esta manera se protege la Raspberry Pi de posibles daños causados por daños abruptos o desconexiones bruscas de la alimentación.

- En cuanto a la tarjeta de desarrollo que utilizamos para desarrollar las prácticas en PIC EasyPIC V6, actualmente la versión más reciente es EasyPIC V8 que a diferencia de su predecesora tiene integrado el módulo WiFi.

- Otra de las opciones en placas es la desarrollada por Microchip Technology, la placa Microchip IoT AC164164, la cual cuenta con herramientas de desarrollo WiFi.



Glosario

PICmicro: *Peripheral Interface Controller*, Controlador de interfaz periférica.

EEPROM: *Electrically Erasable Programmable Read Only Memory*, Memoria de solo lectura programable y borrable eléctricamente.

ROM: *Read Only Memory*, Memoria de solo lectura.

RAM: *Random Access Memory*, Memoria de acceso aleatorio.

CPU: *Central Processing Unit*, Unidad central de procesamiento.

ALU: *Arithmetic and Logic Unit*, Unidad lógica y aritmética.

IDE: *Integrated Development Environment*, Entorno de desarrollo integrado.

DMA: *Direct Memory Access*, Acceso Directo de Memoria.

PROM: *Programmable read only memory*, Memoria programable de solo lectura.

EPROM: *Erasable Programmable read only Memory***,** Memoria de solo lectura programable y borrable eléctricamente.

E/S:

XTAL: Cristal de cuarzo.

WDT: *Watchdog Timer*, Perro guardián.

CISC: *Complex Instruction Set Computer*, Computadora con conjunto de instrucciones complejas.

RISC: *Reduced Instruction Set Computer*, Computadora con conjunto de instrucciones reducidas.

W: *Working Register*, Registro de Trabajo.

LIFO: *Last In, First Out.*

XT: Cristal de cuarzo.

RC: Oscilador con resistencia y capacitor.

HS: Cristal de alta velocidad.

LP: Cristal para baja frecuencia y bajo consumo de corriente.

EIA: *Electronics Industries Association*, Alianza de Industrias Electrónicas.

DTE: *Data Terminal Equipment*, Equipos Terminales de Datos.

DCE: *Data Communication Equipment*, Equipos de Comunicación de Datos.

RS-232C: *Recommended Standard número 232 revisión C*, Estándar Recomendado 232.

I^2C: *Inter-Integrated Circuit*, Intercomunicación con Circuitos Integrados.

SDA: *Serial Data Line.* Línea de datos

SCL: *Serial Clock Line, Línea de reloj*

USART: *Universal Synchronous Asynchronous Receiver Transmitter*

EUSART: *Enhanced Universal Synchronous Asynchronous Receiver Transmitter*

SCI: *Serial communication Interface*

PWM: *Pulse Width Modulated*, Modulación por ancho de pulso.

CAN: *Controller Area Network*, Red de área del controlador.

RETFIE: *Return From Interrupt*

Open source: código abierto.

GPL: *General Public License,* Licencia Pública General.

LGPL: *GNU Lesser General Public License*, Licencia Pública General Reducida de GNU.

I^2C: *Inter-Integrated Circuit* también conocido con el nombre de TWI (*TWo-wIre*)

SPI: *Serial Peripheral Interface.*

GPIO: *General Purpose Input/Output*, Entradas/Salidas de propósito general.

GPU: *Graphics Processing Unit*, Unidad de Procesamiento Gráfico.

SCL: señal de reloj

SDA: señal de datos

MOSI: *Master Output Slave Input*, Salida de datos del maestro y entrada de datos del esclavo.

MISO: *Master Input Slave Output*, Salida de datos del esclavo y entrada de datos del maestro.

SS: *Select*

CS: *Chip Select*

UART: *Universal Asynchronous Receiver Transmiter*, transmisor/receptor asíncrono universal.

ARPANET: *Advanced research projects agency network*, Red de agencias de proyectos de investigación avanzada.

RFID: Radio Frequency Identification, Red Identificación por Radio Frecuencia.

IAB: *Internet Architecture Board*, Consejo de Arquitectura de internet.

UIT, ITU: *International Telecommunication Union,* Unión Internacional de Telecomunicaciones.

ISO: Organización Internacional de Normalización

IEEE: Instituto de Ingenieros Eléctricos y Electrónicos.

IA: Inteligencia Artificial

IAB: *Internet Architecture Board*, Consejo de Arquitectura de internet.

IETF Journal: Internet Engineering Task Force, Grupo de trabajo de ingeniería de Internet.

TCP: *Transmission Control Protocol,* Protocolo de control de transmisión.

UDP: *User Datagram Protocol,* Protocolo de datagramas de usuario.

HTTP: *Hypertext Transfer Protocol,* Protocolo de transferencia de Hipertexto.

TLS: *Transport Layer Security*, seguridad de la capa de transporte.

IIoT: *Industrial Internet of Things*, Internet industrial de las cosas.

IT: *Information Technology*, Tecnología de la información.

OT: *Operational Technology,* Tecnología de la Operación.

IoMT: *Internet of Medical Things,* Internet de las cosas médicas.

TIC: Tecnologías de la Información y la Comunicación.

P2P: *people to people*, persona a persona.

M2P: *machine to people*, máquina a persona.

M2M: *machine to machine*, máquina a máquina.