Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería Mecánica Eléctrica

AUTOMATIZACIÓN DE LA CAPTURA DE DATOS DE IDENTIFICACIÓN DE MEDIDORES DE ENERGÍA ELÉCTRICA POR MEDIO DE LABVIEW

Wilson Felipe Pérez Hernández
Asesorado por el Ing. Otto Andrino

Guatemala, mayo de 2017

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

AUTOMATIZACIÓN DE LA CAPTURA DE DATOS DE IDENTIFICACIÓN DE MEDIDORES DE ENERGÍA ELÉCTRICA POR MEDIO DE LABVIEW

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

WILSON FELIPE PEREZ HERNANDEZ
ASESORADO POR EL ING.OTTO ANDRINO

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO ELECTRÓNICO

GUATEMALA, MAYO DE 2017

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANO	Ing. Pedro Antonio Aguilar Polanco
VOCAL I	Ing. Angel Roberto Sic García
VOCAL II	Ing. Pablo Christian de León Rodríguez
VOCAL III	Ing. José Milton de León Bran
VOCAL IV	Br. Jurgen Andoni Ramírez Ramírez
VOCAL V	Br. Oscar Humberto Galicia Nuñez
SECRETARIA	Inga. Lesbia Magalí Herrera López

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANO	Ing. Murphy Olympo Paiz Recinos
EXAMINADOR	Ing. Carlos Eduardo Guzmán Salazar
EXAMINADOR	Ing. Julio Rolando Barrios Archila
EXAMINADOR	Ing. José Anibal Silva de los Angeles
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

AUTOMATIZACIÓN DE LA CAPTURA DE DATOS DE IDENTIFICACIÓN DE MEDIDORES DE ENERGÍA ELÉCTRICA POR MEDIO DE LABVIEW

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería Mecánica Eléctrica con fecha 3 de agosto de 2015.

Wilson Felipe Pérez Hernández

Guatemala 22 de marzo de 2017

Ing. Francisco Javier González López
Director Escuela de Mecánica Eléctrica
Universidad de San Carlos de Guatemala
Presente

Estimado Ing. González

Por medio de la presente le informo que he asesorado el trabajo de tesis titulado: **"AUTOMATIZACIÓN DE LA CAPTURA DE DATOS DE IDENTIFICACIÓN DE MEDIDORES DE ENERGÍA ELÉCTRICA POR MEDIO DE LABVIEW"**, desarrollado por el estudiante Wilson Felipe Pérez Hernández, Carné 2004-13106, previo a optar al título de Ingeniero Electrónico.

Con Base en la revisión y corrección de dicho trabajo, considero que ha alcanzado los objetivos propuestos, por lo que el estudiante y asesor, nos hacemos responsables del contenido del mismo.

Atentamente,

Ing. Otto Fernando Andrino Gonzalez
Colegiado No. 4038

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERIA

Ref. EIME 14. 2017
Guatemala, 18 de ABRIL 2017.

Señor Director
Ing. Francisco Javier González López
Escuela de Ingenierìa Mecànica Elèctrica
Facultad de Ingenierìa, USAC.

Señor Director:

Me permito dar aprobación al trabajo de Graduación titulado: AUTOMATIZACIÓN DE LA CAPTURA DE DATOS DE IDENTIFICACIÓN DE MEDIDORES DE ENERGÍA ELÉCTRICA POR MEDIO DE LABVIEW, del estudiante, Wilson Felipe Pérez Hernández, que cumple con los requisitos establecidos para tal fin.

Sin otro particular, aprovecho la oportunidad para saludarle.

Atentamente,
ID Y ENSEÑAD A TODOS

Ing. Otto Fernando Andrino González
Coordinador del Area de Electrotécnia

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DIRECCION ESCUELA DE INGENIERIA MECANICA ELECTRICA
FACULTAD DE INGENIERIA

sro

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERIA

REF. EIME 15. 2017.

El Director de la Escuela de Ingeniería Mecánica Eléctrica, después de conocer el dictamen del Asesor, con el Visto bueno del Coordinador de Área, al trabajo de Graduación del estudiante: WILSON FELIPE PÉREZ HERNÁNDEZ Titulado: AUTOMATIZACIÓN DE LA CAPTURA DE DATOS DE IDENTIFICACIÓN DE MEDIDORES DE ENERGÍA ELÉCTRICA POR MEDIO DE LABVIEW, procede a la autorización del mismo.

Ing. Francisco Javier González López

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DIRECCION ESCUELA DE INGENIERIA MECANICA ELECTRICA
FACULTAD DE INGENIERIA

GUATEMALA, 18 DE ABRIL 2017.

Universidad de San Carlos
de Guatemala

Facultad de Ingeniería
Decanato

DTG. 229.2017

El Decano de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería Mecánica Eléctrica, al Trabajo de Graduación titulado: **AUTOMATIZACIÓN DE LA CAPTURA DE DATOS DE IDENTIFICACIÓN DE MEDIDORES DE ENERGÍA ELÉCTRICA POR MEDIO DE LABVIEW,** presentado por el estudiante universitario: **Wilson Felipe Pérez Hernández,** y después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, autoriza la impresión del mismo.

IMPRÍMASE:

Ing. Pedro Antonio Aguilar Polanco
Decano

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DECANO
FACULTAD DE INGENIERIA

Guatemala, mayo de 2017

/gdech

ACTO QUE DEDICO A:

Dios	Por su inmensa sabiduría, amor y fortaleza, por estar presente en cada día de mi vida.
Mis padres	Othoniel Pérez (q. e. p. d.) quien me dio lecciones valiosas para toda la vida y a mi madre Dora Hernández por su paciencia, cuidado y ser una mujer trabajadora.
Mi esposa	Elida Florián de Pérez por ser mi inspiración, apoyo incondicional, amiga, compañera de aventuras y por compartir su vida junto a la mía.

AGRADECIMIENTOS A:

Universidad de San Carlos de Guatemala	Por abrir sus puertas a los jóvenes de los departamentos de Guatemala y por darme la oportunidad de ser parte de esta gloriosa universidad.
Facultad de Ingeniería	Por todos los conocimientos adquiridos en estos años y por ser parte fundamental en mi carrera profesional.
Mis hermanos y hermanas	Por su apoyo incondicional en los momentos más difíciles.
Ing. Giovanni Salazar e Ing. Elmar Fuentes	Por ser personas importantes e influyentes en mi carrera profesional.
Mis vecinos	Evelyn Catalán y José Azañon por su apoyo incondicional.
Mi asesor	Por el tiempo dedicado a este trabajo de graduación.
Mis amigos de la facultad	Elida Florián, Jose Solis, Adolfo Vasquez, Cristobal Tezen, Carlos Bucu, por estar presentes en cada momento importante de mi vida.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

TABLAS

LISTA DE SÍMBOLOS

Símbolo	Significado
IMAQ	Adquisición de imágenes, paquete de instrumentos virtuales de *National Instruments*
bps	Bits por segundo
CMY	Cian, magenta y amarillo
I2C	Circuito interintegrado
byte	Conjunto de ocho bits
A/D	Conversor analógico digital
DC	Corriente directa
bit	Dígito binario
LED	Diodo emisor de luz
CCD	Dispositivos de carga acoplada
Hz	Hertz
ICSP	Interfaz de programación serial del circuito
Bitmap	Mapa de bits
Mbps	Megabits por segundo
MHZ	Megahertz
RAM	Memoria de acceso aleatorio
EEPROM	Memoria de solo lectura programable y borrable electrónicamente
m	Metro
PIC	Microcontrolador de microchip
mm	Milímetro
PWM	Modulación por ancho de pulso

MCPWM	Módulo de control de modulación de ancho de pulso
nm	Nanómetro
PDI	Procesamiento digital de imágenes
PTMR	Registro de tiempo para pulso programable
RGB	Rojo, verde y azul
CMOS	Semiconductor de óxido metálico complementario
HSL	Tonalidad, saturación y luminancia
HSV	Tonalidad, saturación y valor
MCU	Unidad microcontroladora

GLOSARIO

Array	Conjunto de objetos de la misma clase.
Binario	Es un sistema de numeración en el cual los números se representan utilizando solamente dos cifras: cero y uno (0 y 1).
Bluetooth	Es una especificación industrial para redes inalámbricas de área personal (WPAN) que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia en la banda ISM de los 2,4 GHz.
Bus digital (informática)	Es un sistema digital que transfiere datos entre los componentes de una computadora o entre varias computadoras.
Convolución	Operador matemático que transforma dos funciones, f y g, en una tercera función, filtro lineal.
Difracción	Es un fenómeno característico de las ondas que se basa en su desviación al encontrar un obstáculo o al atravesar una rendija.

Diodos Es un componente electrónico de dos terminales que permite la circulación de la corriente eléctrica a través de este en un solo sentido.

DIP o DIL Formato de encapsulado en construcción de circuitos integrados; bloque con dos hileras paralelas de pines.

Electromagnético Es un campo físico, de tipo tensorial, producido por aquellos elementos cargados eléctricamente que afecta a partículas con carga eléctrica.

Espectro Espectro visible o espectro lumínico: conjunto de fotones dividido por difracción en diversas franjas de colores desde el violeta hasta el rojo (en luz visible) que se observa a partir de la luz blanca cuando la misma atraviesa un prisma óptico.

Estátor Es la parte fija de una máquina rotativa y uno de los dos elementos fundamentales para la transmisión de potencia (en el caso de motores eléctricos) o corriente eléctrica (en el caso de los generadores eléctricos).

Estroboscópicos Efecto óptico que se produce al iluminar, mediante destellos, un objeto que se mueve en forma rápida y periódica.

Ethernet	Es un estándar de redes de área local para computadores con acceso al medio por detección de la onda portadora y con detección de colisiones (CSMA/CD).
Instrumentación	En la industria, es un grupo de elementos que sirven para medir, transmitir o controlar variables de un proceso industrial con el fin de optimizar los recursos.
Interoperabilidad	Habilidad de dos o más sistemas o componentes para intercambiar información y utilizar la información intercambiada.
Matriz	Conjunto de variables del mismo tipo que esta compuesto por filas y columnas.
Microcontroladores	Es un circuito integrado programable, capaz de ejecutar las órdenes grabadas en su memoria.
Modbus	Protocolo de comunicaciones situado en el nivel 7 del Modelo OSI, basado en la arquitectura maestro/esclavo (RTU) o cliente/servidor (TCP/IP).
Píxeles	Es la menor unidad homogénea en color que forma parte de una imagen digital.
Radiación	Es la propagación de energía en forma de ondas electromagnéticas o partículas subatómicas a través del vacío o de un medio material.

Reluctancia magnética	Es la resistencia que posee al paso de un flujo magnético cuando es influenciado por un campo magnético.
Rotor	Es el componente que gira (rota) en una máquina eléctrica: un motor o un generador eléctrico.
Puerto serial	Es una interfaz de comunicaciones de datos digitales, frecuentemente utilizado por computadoras y periféricos, donde la información es transmitida bit a bit, enviando un solo bit a la vez.
Whatchdog	En electrónica (en español perro guardián) es un mecanismo de seguridad que provoca un *reset* del sistema en caso de que este se haya bloqueado.
Labview	*Software* de desarrollo orientado a la programación de objetos, *software* propietario de *National Instrument.*
USB	Bus estándar industrial que define los cables, conectores y protocolos usados en un bus para conectar, comunicar y proveer de alimentación eléctrica a computadoras, periféricos y dispositivos electrónicos.
Wifi	Conexión entre dispositivos de una forma inalámbrica.

CPU	Unidad de procesamiento central, es el *hardware* que interpreta las instrucciones de un programa informático.

RESUMEN

El presente trabajo estudia algunas técnicas de procesamiento digital de imágenes para la extracción de características; reconocimiento de patrones por medio del *software* de desarrollo Labview líder en el procesamiento digital de imágenes. Labview es un *software* que permite a los ingenieros o científicos crear soluciones que son integrables y escalables según sea la necesidad; tiene la capacidad de integrar casi cualquier dispositivo para la captura de imágenes ya sea a través de un puerto serial, USB, wifi o ethernet.

La versatilidad del *software* Labview de poder manejar puertos de comunicación y permitir comunicación con otros dispositivos lo hace un *software* interoperable; en este caso, comunicación a través del puerto serial RS232, con un microcontrolador que, se encarga de controlar los movimientos secuenciales de dos motores, paso, a paso para el posicionamiento preciso de la cámara para que se capture la imagen de un medidor de energía eléctrica y puedan extraerse las características más importantes: número de serie de fábrica, número de serie de la empresa de distribución de electricidad y lectura kWh.

La cámara, el microcontrolador, motores paso a paso y el *software* Labview son parte fundamental de una solución para una empresa de distribución de energía eléctrica, donde se pretende automatizar procesos rutinarios que no aportan un valor significativo y para obtener un mayor provecho de los activos más preciados de una empresa: el recurso humano.

OBJETIVOS

General

Automatizar la captura de datos de identificación de medidores de energía eléctrica por medio de Labview

Específicos

1. Explicar el funcionamiento de Labview y su importancia en la automatización industrial con ayuda de los diferentes módulos.

2. Explicar la integración de microcontroladores PIC 16F877A a Labview por medio del puerto serial RS232.

3. Desarrollar el algoritmo del microcontrolador PIC16F877A para el control de movimiento horizontal y vertical.

4. Desarrollar el algoritmo para la captura de imágenes y procesamiento digital para la extracción de datos característicos.

INTRODUCCIÓN

Labview de *National Instrument* como *software* de desarrollo tiene presencia a nivel mundial y es preferido por ingenieros y científicos para desarrollar soluciones integrales e interoperables; los módulos de funciones de Labview permiten crear soluciones adecuadas a las necesidades de cada industria; las soluciones están orientadas en la adquisición de datos, instrumentación, hardware de control y monitoreo en tiempo real, buses de comunicación industrial y control de instrumentos.

Estas funciones de Labview permiten integrar microcontroladores por medio de puertos de comunicación, capturar imágenes por medio de cámaras USB y, posterior, aplicar técnicas de procesamiento digital de imágenes a la imagen capturada para extraer datos característicos de los medidores de energía eléctrica y realizar una inspección de los medidores.

Con las funciones de Labview se pretende automatizar uno de los procesos en la bodega de almacenamiento de una empresa distribuidora de electricidad. Este proceso consiste en registrar los datos característicos de cada uno de los medidores de energía eléctrica que son ingresados; estos medidores son retirados de los domicilios y llevados a la bodega de la empresa distribuidora, proceso que realiza una persona contratada para este trabajo.

Con los recursos que ofrece Labview se pretende automatizar este proceso con el buen aprovechamiento de la tecnología con la que hoy en día se dispone, automatizando este proceso puede emplearse el recurso humano

en otras actividades que aporten mucho más valor para la empresa distribuidora.

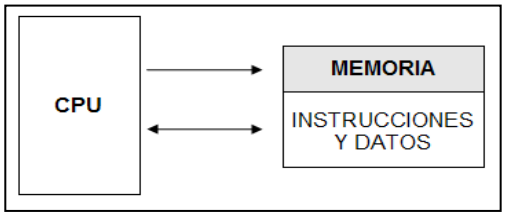
1. PLANTAMIENTO DEL PROBLEMA

1.1. Microcontroladores

Son computadores digitales integrados en un chip, utiliza un procesador en su arquitectura y son capaces de realizar muchas tareas lógicas, un microcontrolador de fábrica no es capaz de realizar tareas, es decir, debe ser programado previo a ejecutar alguna tarea lógica en la actualidad existen diversos *software* y lenguajes de programación para realizar la programación de los microcontroladores, se debe considerar que existen diversos modelos con diferentes capacidades como por ejemplo: el tamaño de memoria, pines de entrada y salida, interrupciones, procesamiento de señales y puertos de comunicación.

La arquitectura de los microcontroladores pueden ser dos, las más utilizadas por las casas fabricantes de estos microcontroladores: la primera es la de Von Nuemman donde la unidad central de procesamiento, conocido como CPU, está conectada a una única memoria con las instrucciones de programa y datos, donde el tamaño de la unidad de datos está fijado por el ancho del bus de la memoria; a continuación, se muestra la arquitectura Von Nuemman.

Figura 1. **Arquitectura Von Nuemman**

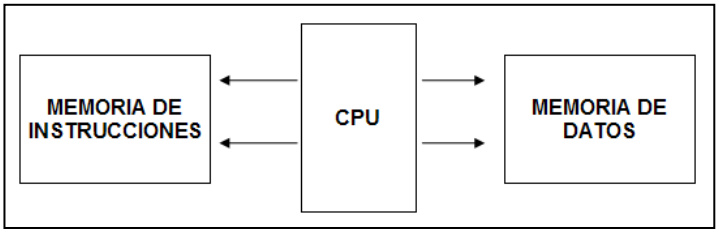


Fuente: SÁNCHEZ, Sergio. *Introducción y arquitectura de microcontroladores*. https://microcontroladoressesv.wordpress.com/arquitectura-de-los-microcontroladores. Consulta: 10 de noviembre de 2016.

Las dos principales deficiencias de esta arquitectura son: la longitud de instrucciones que está limitada por la unidad de longitud de datos lo que significa que el procesador debe ingresar varias veces a la memoria para ejecutar una instrucción muy compleja, y el cuello de botella que se genera por tener un único bus para ingresar a las instrucciones y los datos.

La segunda arquitectura y las más utilizada hoy en día es la arquitectura harvard consiste en una memoria para las instrucciones y otra para datos, conectadas por un bus independiente una de la otra al CPU; lo que permite un rápido acceso a los datos y las instrucciones. La memoria con las instrucciones es llamada memoria de programa y la memoria que contiene los datos; es llamada memoria de datos y solo almacena los datos, los buses que se conectan a las memorias son independientes uno de otro y pueden ser de distintos anchos. Para un procesador de set de instrucciones reducido, por sus siglas en ingles RISC, puede diseñarse de tal forma que una instrucción tenga una única posición de memoria de programa de longitud; como los buses son independientes el CPU puede estar accediendo a los datos para ejecutar una instrucción.

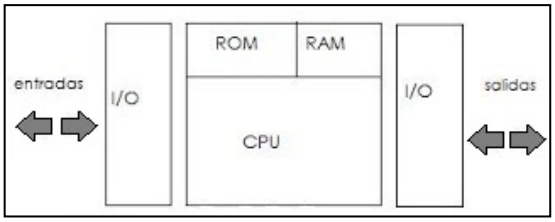
Figura 2. **Arquitectura harvard**



Fuente: SÁNCHEZ, Sergio. *Introducción y arquitectura de microcontroladores*. https://microcontroladoressesv.wordpress.com/arquitectura-de-los-microcontroladores. Consulta: 11 de noviembre de 2016.

Las ventajas saltan a la vista ya que la memoria de instrucciones y datos pueden ser de diferente tamaño por su arquitectura; también, se puede acceder a ambas memorias al mismo tiempo; una de las desventajas de esta arquitectura es que debe poseer instrucciones especiales para acceder a tablas de valores constantes que puedan incluir en los programas, ya que estas tablas se encuentran físicamente en la memoria del programa.

Figura 3. **Arquitectura interna de un microcontrolador**



Fuente: SÁNCHEZ, Sergio. *Introducción y arquitectura de microcontroladores*. https://microcontroladoressesv.wordpress.com/arquitectura-de-los-microcontroladores. Consulta: 15 de noviembre de 2016.

1.1.1.1. CPU (unidad central de proceso)

Es el núcleo del microcontrolador; es el encargado de ejecutar las instrucciones que se almacenan en la memoria, regularmente, llamado procesador o microprocesador; término que no se debe confundir con el microcontrolador ya que este último contiene un microprocesador y sin el cual no sería útil; el microcontrolador es un sistema completo que puede llevar a cabo de forma autómata una labor.

1.1.2. Memoria

Parte importante de un microcontrolador que almacen información durante un determinado tiempo; una es la encargada de almacenar el código del programa, las instrucciones; la otra es la encargada de almacenar los datos que usaremos durante la ejecución del programa, por lo tanto, se tiene memoria de programa y memoria de datos, respectivamente.

Las memorias, por su propio uso en un microcontrolador, se hace necesario diferenciar las y el uso de diferentes tipos de memoria; habrá que tener en cuenta la clasificación de las memorias: volátil, aquella que pierde la información cuando se desconecta de la alimentación y típicamente contiene la información que es la memoria de programa donde se almacena información que se usará en la ejecución de un programa; y la no volátil, aquella que no pierde la información cuando es desconectada de la alimentación y normalmente, es utilizada para la memoria de programa.

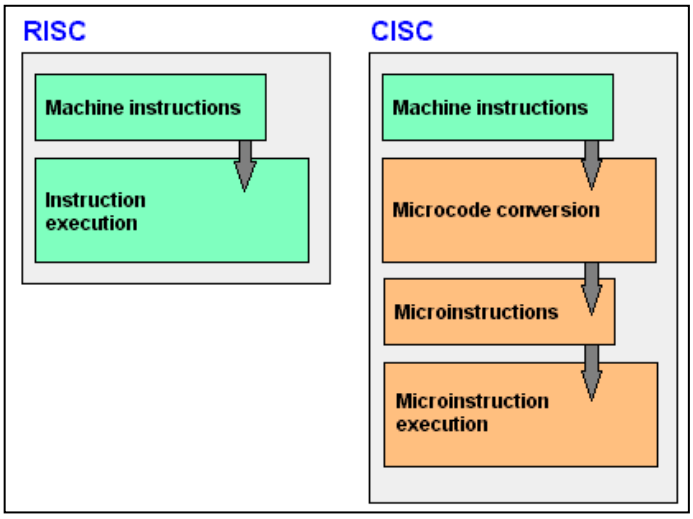
1.1.3. Unidades de entrada y salida

Son sistemas que emplean el microcontrolador para comunicarse con el exterior; estos sistemas permiten introducir u obtener información del microcontrolador hacia otros sistemas.

1.1.4. Arquitecturas RISC y CISC

Para diseñar un microprocesador es muy importante decidir cuál será el juego de instrucciones del cual dependerá el tamaño físico del conjunto de instrucciones que deberán ejecutarse en el microcontrolador que debe ser capaz de describirse en el lenguaje de estas instrucciones.

Figura 4. **Arquitectura RISC y CISC**



Fuente: SÁNCHEZ, Sergio. *Introducción y arquitectura de microcontroladores*. https://microcontroladoressesv.wordpress.com/arquitectura-de-los-microcontroladores. Consulta: 20 de noviembre de 2016.

- RISC: por sus siglas en inglés, *reduce instruction set computer,* el microcontrolador reconoce y ejecuta solo operaciones básicas, por ejemplo: sumar, restar, copiar, etc. Las operaciones más complicadas, como la multiplicación, se llevan a cabo realizando una adición sucesiva. Las instrucciones más complicadas se ejecutan a una velocidad de tal forma que el usuario no percibe este tipo de instrucciones más complicadas.

- CISC: por sus siglas en inglés, *complex instruction set computer*, los microcontroladores son diseñados para reconocer más de 200 instrucciones diferentes; pueden ejecutar muchas instrucciones a alta velocidad, claro está, se debe poseer un amplio conocimiento en el lenguaje que no siempre es tan fácil.

1.2. Microcontroladores PIC 16F877A

Los PIC son una familia de microcontroladores desarrollados y fabricados por Microchip Technologies Inc., que cuentan con una tecnología RISC y poseen características especiales según sea el modelo del PIC.

Los microcontroladores PIC se asemejan mucho a una computadora: cuentan con los mismos recursos, poseen memoria RAM para el almacenamiento temporal de los datos; memoria ROM para el programa, también, existe la que puede ser modificables pero en este caso se le llama EEPROM, puertos de entrada o salida, temporizadores; algunos con otras características especiales: convertidores A/D, comparadores, USAR y comunicación serie I2C, entre muchos más.

El microcontrolador PIC es el encargado de dirigir todos los procesos de un circuito electrónico, con base en las instrucciones de programa o rutinas que definen funciones específicas de control o supervisión, por tal razón es necesario conocer la arquitectura interna del microcontrolador PIC que se desea programar.

El microcontrolador PIC 16F877A de microchip es uno de los más populares y de bajo costo en Guatemala; se pueden comprar en el mercado guatemalteco a un costo de Q 100,00 aproximadamente y son más

económicos que los microcontroladores de 16 y 32 bits que son empleados en aplicaciones de instrumentación y operaciones matemáticas complejas. Un microcontrolador de 8 bits que en la actualidad domina el mercado y por su características los hace apropiados para la gran mayoría de aplicaciones industriales; se prefiere emplear el microcontrolador PIC 16F877A para estas aplicación por las siguientes características:

- Costo: como se ha mencionado, el PIC 16F877A se puede comprar a un precio de aproximadamente Q 100,00 en cualquier tienda de venta de electrónicos de Guatemala.

- *Software*: hay que tomar en cuenta las herramientas de desarrollo que hay disponible en el mercado: emuladores, compiladores, simuladores, etc. Para el PIC 16F877A existe una gran variedad de *software* como el de *oshonsoft*.

- Aplicación: el PIC 16F877A tiene puertos de entradas y salidas digitales de 8 bits, ancho de palabra de 8 bits e interfaz serial RS232, característica de vital importancia para el desarrollo del autómata; este PIC tiene más características que se detallan más adelante.
- Encapsulado: DIP o DIL es uno de los más comunes.

A continuación, se detallarán algunas de sus características e identificación de los pines:

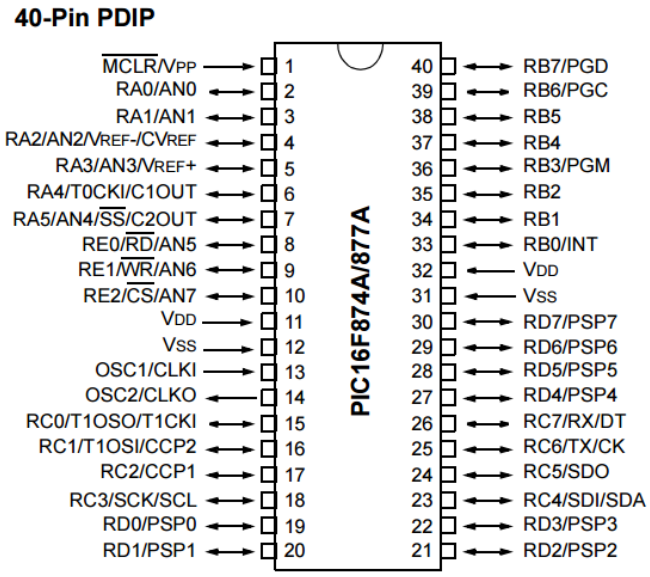
- Memoria de programa tipo Flash.
- Memoria de programa 14 kB.
- Velocidad del CPU (MIPS) 5.
- RAM 368 bytes.

- EEPROM 256 bytes.
- Comunicaciones digitales 1-UART, 1-A/E/USART, 1-SPI, 1-I2C-MSSP(SPI/I2C).
- PWM 2CCP.
- Temporizadores 2 x 8 bit, 1 x 16-bit.
- ADC 8ch, 10 bit.
- 2 comparadores de voltaje.
- Temperatura de funcionamiento -40 a 125 °C.
- Voltaje de operación 2 a 5.5 V DC.
- Pines 40.

1.2.1. Configuración de pines

Los pines de entrada y salida del microcontrolador están configurados en cinco puertos: el puerto A tiene 6 líneas, los puertos B, C y D tienen 8 líneas y el puerto E tiene 3 líneas; cada uno de los puertos descritos anteriormente pueden configurarse como entrada o salida y pueden programarse independientemente. Si los registros de los puertos están configurados con un bit "0" es salida, si es bit "1" es entrada.

Figura 5. **Configuración de pines microcontrolador PIC 16F877A**



Fuente: *Microchip Technology Inc*., http://ww1.microchip.com/downloads/en/DeviceDoc/39582C.pdf. Consulta: 22 de noviembre de 2016.

Como se verá más adelante, los pines de los puertos de entradas y salidas digitales pueden configurarse también de forma especial, los puertos A y E también puede trabajar como entradas analógicos a digitales; esta señal puede venir de un sensor o de una entrada analógica y el microcontrolador convertirla a su equivalente digital y luego realizar un proceso de control; el pin RB0/INT puede configurarse por *software* para que funcione como interrupción externa, para configurar se utiliza uno de los bits de los registros que controla las interrupciones.

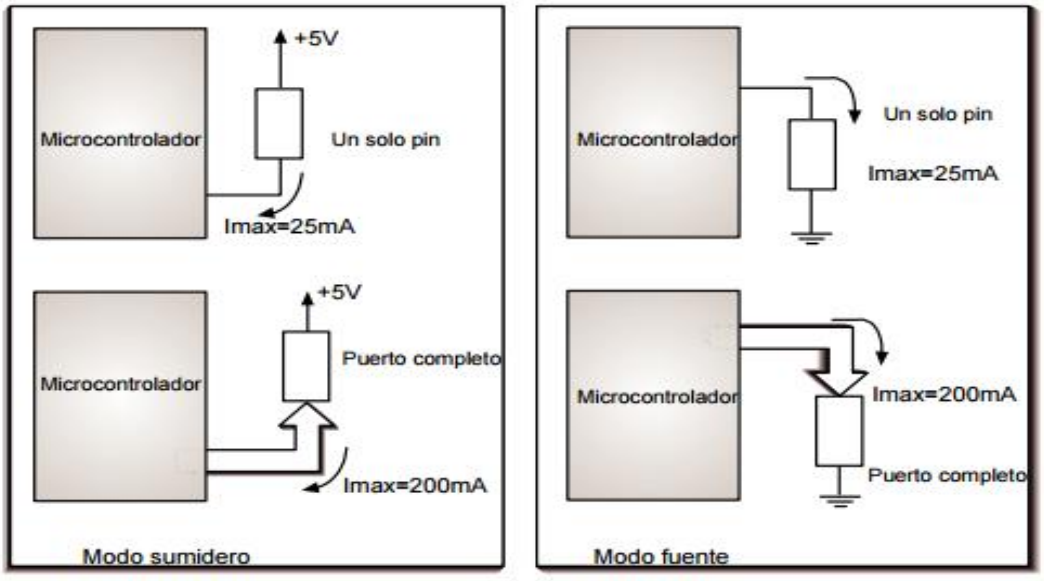
El pin R4/TOCK1 del puerto puede ser configurado como un pin de entrada/salida o como entrada del temporizador/contador; cuando se programa como entrada digital, funciona como disparador Schmitt, transforma señales

distorsionadas a señales de niveles lógicos como "1" o "0", cuando se usa como salida digital, se comporta como salida colector abierto quiere decir que debe de colocarse una resistencia de *pull-up* y la salida funciona como lógica inversa, es decir cuando se escribe "0" al puerto del pin en realidad el puerto entrega "1".

El puerto E puede configurarse para controlar la conexión en modo microprocesador con otros dispositivos utilizando las líneas RE0/RD/AN5 para lectura, RE1/WP/AN6 para escritura y RE2/CS/AN7 selección de chip; en este modo el puerto D funciona como un bus de 8 bits.

La máxima capacidad de corriente que soportan los pines del microcontrolador en modo sumidero o en modo fuente es de 25 mA; la máxima capacidad de corriente total en los puertos se presenta a continuación.

Figura 6. **Capacidad de corriente de los pines y puertos**



Fuente: *El microcontrolador PIC16F877.* http://www.utp.edu.co/~eduque/arquitec/PIC16F877.pdf. Consulta: 23 de noviembre de 2016.

Figura 7. **Descripción de los pines del microcontrolador PIC 16F877A**

Nombre pin	Pin	Descripción
RA0/AN0	2	E/S Digital o Entrada análoga 0.
RA1/AN1	3	E/S Digital o Entrada análoga 1.
RA2/AN2 V_{ref} -	4	E/S Digital o Entrada análoga 2.
RA3/AN3/V_{ref} +	5	E/S Digital o Entrada análoga 3.
RA4/T0CKI	6	Bit 4 del puerto A (E/S bidireccional). También se usa como entrada de reloj al temporizador/contador TMR0. Salida de colector abierto.
RA5/SS/AN4	7	E/S Digital o Entrada análoga 4. También lo usa el puerto serial síncrono.
RB0/INT	33	Bit 0 del puerto B (E/S bidireccional). Buffer E/S: TTL/ST. También se usa como entrada de interrupción externa (INT).
RB1	34	Bit 1 del puerto B (E/S bidireccional). Buffer E/S: TTL
RB2	35	Bit 2 del puerto B (E/S bidireccional). Buffer E/S: TTL
RB3/PGM	36	Bit 3 del puerto B (E/S bidireccional). Buffer E/S: TTL (Programación en bajo voltaje)
RB4	37	Bit 4 del puerto B (E/S bidireccional). Buffer E/S: TTL. Interrupción por cambio del pin.
RB5	38	Bit 5 del puerto B (E/S bidireccional). Buffer E/S: TTL. Interrupción por cambio del pin.
RB6/PGC	39	Bit 6 del puerto B (E/S bidireccional). Buffer E/S: TTL/ST. Interrupción por cambio del pin. Entrada de reloj para programación serial.
RB7/PGD	40	Bit 7 del puerto B (E/S bidireccional). Buffer E/S: TTL/ST. Interrupción por cambio del pin. Entrada de datos para programación serial.
RC0/T1OSO/T1CKI	15	E/S Digital. Salida del oscilador Timer 1 o entrada de reloj Timer 1.
RC1/T1OSI/CCP2	16	E/S Digital. Entrada del oscilador Timer 1. Entrada Captura 2; Salida Compara 2; Salida PWM 2
RC2/CCP1	17	E/S Digital. Entrada Captura 1; Salida Compara 1; Salida PWM 1
RC3/SCK/SCL	18	E/S Digital. Línea de reloj serial asíncrono en el modo SPI y el modo I²C
RC4/SDI/SDA	23	E/S Digital. Línea de datos en el modo SPI o en el modo I²C
RC5/SDO	24	E/S Digital.
RC6/TX/CK	25	E/S Digital. Transmisión asíncrona (USART) o reloj síncrono (SSP).
RC7/RX/DT	26	E/S Digital. Recepción asíncrona (USART) o línea de datos (SSP).
V_{DD}	11,32	Voltaje de alimentación DC (+)
V_{SS}	12,31	Referencia de voltaje (GND).
MCLR	1	Entrada de RESET al microcontrolador. Voltaje de entrada durante la programación. En nivel bajo resetea el microcontrolador.
OSC1/CLKIN	13	Entrada oscilador cristal oscilador / Entrada fuente de reloj externa.
OSC2/CLKOUT	14	Salida oscilador cristal. Oscilador RC: Salida con un ¼ frecuencia OSC1
RD0/PSP0	19	E/S Digital. Puede ser puerto paralelo en bus de 8 bits.
RD1/PSP1	20	E/S Digital. Puede ser puerto paralelo en bus de 8 bits.
RD2/PSP2	21	E/S Digital. Puede ser puerto paralelo en bus de 8 bits.
RD3/PSP3	22	E/S Digital. Puede ser puerto paralelo en bus de 8 bits.
RD4/PSP4	27	E/S Digital. Puede ser puerto paralelo en bus de 8 bits.
RD5/PSP5	28	E/S Digital. Puede ser puerto paralelo en bus de 8 bits.
RD6/PSP6	29	E/S Digital. Puede ser puerto paralelo en bus de 8 bits.
RD7/PSP7	30	E/S Digital. Puede ser puerto paralelo en bus de 8 bits.
RE0/RD/AN5	8	E/S Digital. Puede se pin de lectura (*read*) en modo microprocesador.
RE1/WR/AN6	9	E/S Digital. Puede ser pin de escritura (*write*) en modo microprocesador.
RE2/CS/AN7	10	E/S Digital. Puede ser pin de selección de chip (*chip select*) en modo microprocesador.

Fuente: *El microcontrolador PIC16F877A,* http://www.utp.edu.co/~eduque/arquitec/PIC16F877A.pdf, Consulta: 7 de noviembre de 2016.

1.2.2. El oscilador externo

Todo microcontrolador requiere de un circuito externo que le indique la velocidad a la cual debe trabajar; es conocido comúnmente como reloj u

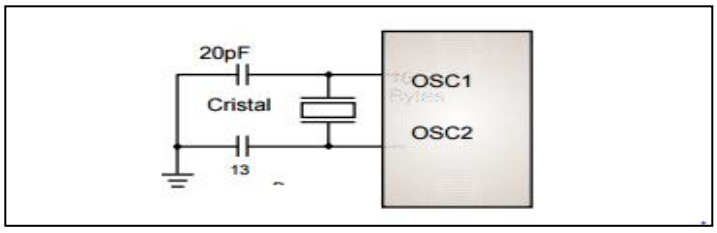
oscilador y es de vital importancia para el buen funcionamiento del microcontrolador, el PIC 16F877a puede utilizar cuatro tipos diferentes de osciladores:

- RC: oscilador basado en resistencias y capacitores
- XT: cristal a unas frecuencias de 1 a 4MHz
- HS: cristal de alta frecuencia a unas frecuencias de 10 a 20 MHz
- LP: cristal de baja frecuencia y bajo consumo de potencia

Es muy importante recordar que en el momento de programar, o como comúnmente se llama "quemar", el microcontrolador se debe especificar cuál es el oscilador que se está utilizando, esto se realiza en el *software* de programación del microcontrolador.

El tipo de oscilador que se sugiere es el XT con un cristal de 4 MHz; garantiza precisión y, además, es muy comercial; dentro del microcontrolador esta frecuencia es dividida por 4 que hace una frecuencia efectiva de 1 MHz, por lo tanto, cada instrucción se ejecuta en un microsegundo; el cristal debe de ir acompañado de dos condensadores.

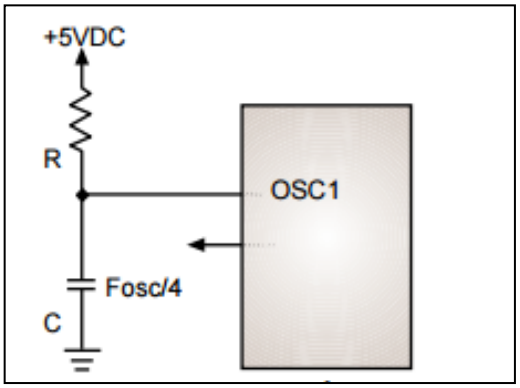
Figura 8. **Oscilador XT**



Fuente: *El microcontrolador PIC16F877.* http://www.utp.edu.co/~eduque/arquitec/PIC16F877.pdf, Consulta: 23 de noviembre de 2016.

Si no se requiere precisión en el oscilador se puede utilizar una resistencia y un condensador, la configuración que se sugiere es la siguiente.

Figura 9. **Oscilador RC**



Fuente: *El microcontrolador PIC16F87.*
http://www.utp.edu.co/~eduque/arquitec/PIC16F877.pdf, Consulta: 23 de noviembre de 2016.

1.2.3. *Reset*

Se requiere un pin de *reset* cuando sea necesario reiniciar el funcionamiento del microcontrolador, ya sea por una falla en el funcionamiento o porque así fue diseñado el sistema; el pin de reset en los PIC 16F877A se llama MCLR que quiere decir *master reset*; existen varias formas de reiniciar el microcontrolador:

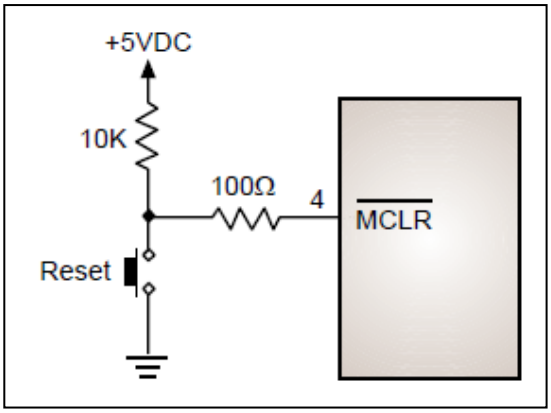
- Al encendido. *Power on reset.*
- Pulso en el pin *reset* durante funcionamiento normal.
- Pulso en el pin *reset* durante el modo de bajo consumo.
- Cuando el contador del *watchdog* se rebalsa durante la operación normal.

- Cuando el contador del *watchdog* se rebalsa durante operación bajo consumo.

El *reset* al encendido se consigue gracias a dos temporizadores OST y PWRT; el primero encargado de mantener en *reset* al microcontrolador hasta que el oscilador de cristal sea estable y el segundo provee un retardo fijo de 72 ms al encendido únicamente, está diseñado para mantener al microcontrolador en *reset* hasta que la fuente de alimentación se estabilice, para utilizar estos dos temporizadores es necesario conectar la fuente de alimentación al *pin reset* sin conectar resistencias como típicamente se realiza.

El *reset* por MCLR es cuando se lleva este pin a un estado lógico "0", mientras el *watchdog* produce un *reset* cuando el contador se rebalsa, es decir, el contador pasa de un estado FFh a 00h; cuando se requiere control en el sistema se pude conectar un botón para esta acción.

Figura 10. ***Reset* cuando se aplica estado bajo en el pin MCLR**

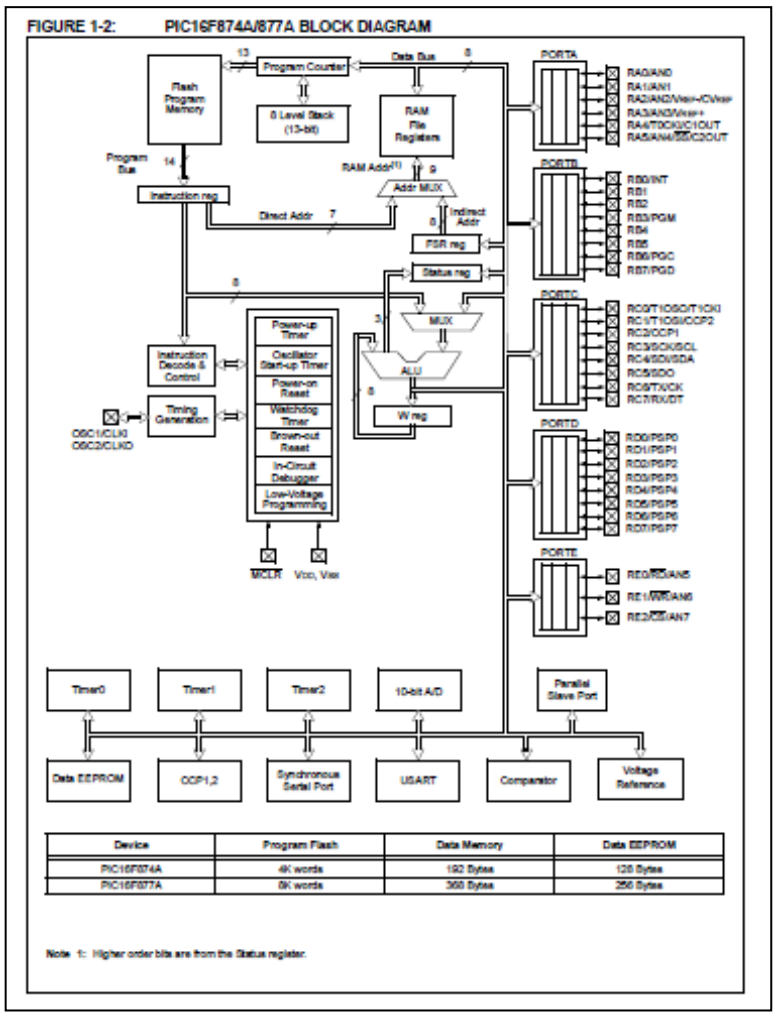


Fuente: *El microcontrolador PIC16F877.*
http://www.utp.edu.co/~eduque/arquitec/PIC16F877.pdf, Consulta: 23 de noviembre de 2016.

1.2.4. Arquitectura interna del microcontrolador PIC 16F877A

Se refiere a los bloques funcionales internos del microcontrolador y la manera en la que están conectados; por ejemplo, la memoria FLASH, la memoria RAM, los puertos, la lógica de control que permite que todo funcione correctamente.

Figura 11. **Arquitectura del microcontrolador PIC 16F877A**



Fuente: Microchip Technology Inc. http://ww1.microchip.com/downloads/en/DeviceDoc/39582C.pdf. consulta: 23 de noviembre de 2016.

En la figura 11 se pueden apreciar sus diferentes bloques y la forma en que se conectan; se muestra la conexión de los puertos, las memorias de datos y programa, los bloques especiales como el *watchdog*, temporizador y el oscilador.

Todos los elementos se conectan entre sí por medio de buses. Un bus es un conjunto de líneas que transportan información entre dos o más módulos. El microcontrolador PIC 16F877A tiene un bloque especial de memoria de datos de 256 bytes del tipo EEPROM, además de los bloques principales que son de memoria del programa y el de datos o registro.

El microcontrolador PIC 16F877A se basa en la arquitectura *Harvard* en el cual los datos y el programa se pueden trabajar con buses y memorias separadas, lo que posibilita que los datos y las instrucciones tengan longitudes diferentes. Esta estructura permite superponer ciclos de búsqueda y ejecución de instrucciones que permite el rendimiento en velocidad del microcontrolador.

1.2.5. Memoria de programa FLASH

Es una memoria de 8K de longitud con datos de 14 bits en cada posición, al ser tipo FLASH se puede borrar y programar eléctricamente; en esta memoria se almacena el programa o códigos que el PIC 16F877A debe ejecutar.

La memoria está divida en cuatro campos de longitud 2K cada una: el primero va de la posición de memoria 000h a la 07FFh, el segundo de la 0800h a la 0FFFh, el tercero 1000h al 17FFh y el cuarto del 1800h al 1FFFh.

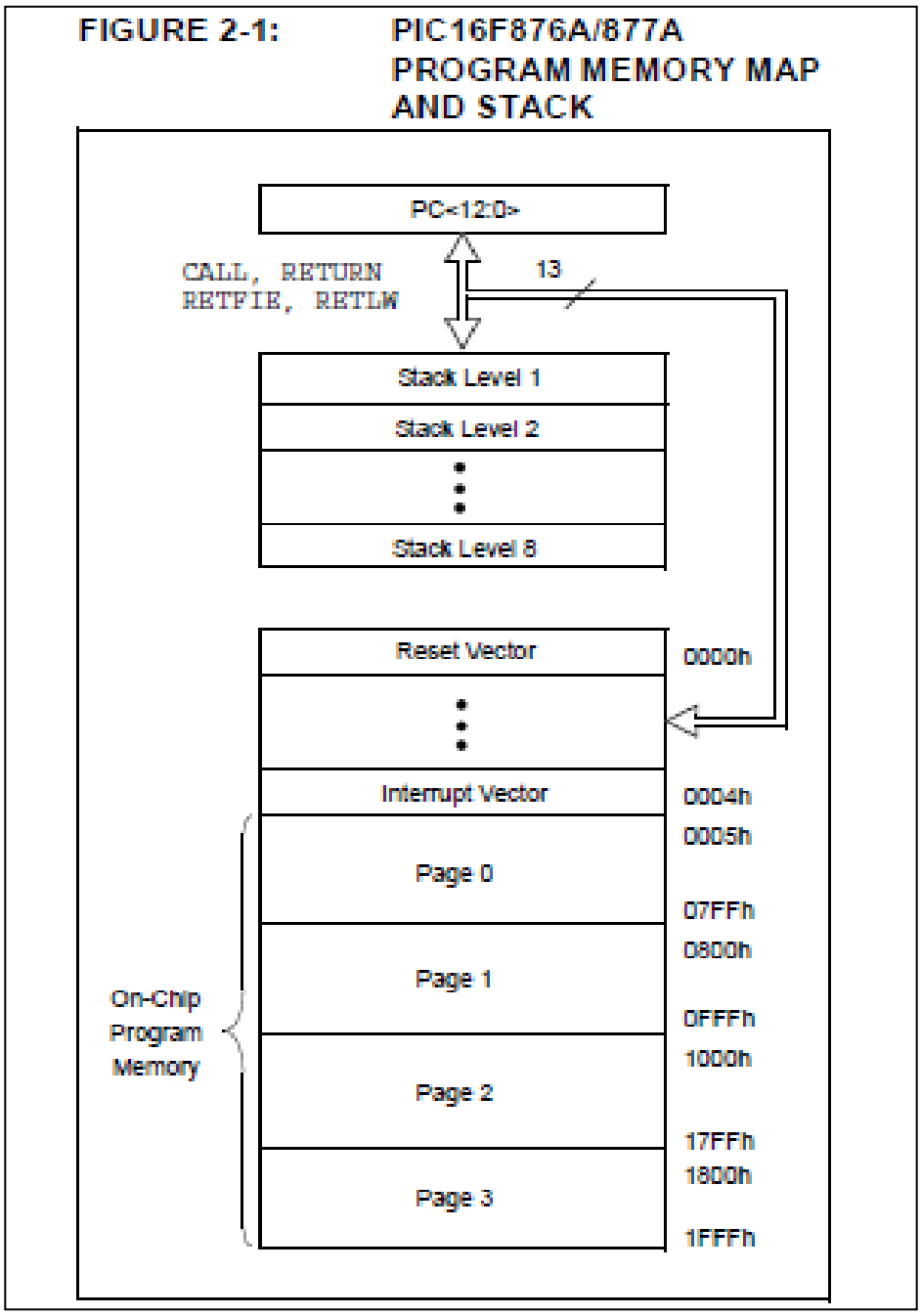
- Vector de *reset*: cuando ocurre un *reset* al PIC, el contador de programa se pone en ceros que quiere decir 0000H; en la primera dirección del

programa se debe escribir todo lo relacionado con la inicialización del programa.

- Vector de interrupción: cuando recibe una señal de interrupción, el contador de programa apunta al 04H de la memoria de programa; ahí se escribe toda la programación necesaria para atender dicha interrupción.

- Pila (*stack*): estos registros no forman parte de la memoria y no permiten acceso por parte del usuario. Se utiliza para guardar el valor del contador del programa al hacer un llamado a una subrutina o cuando se atiende una interrupción, el contador de programa recupera su valor leyéndolo en la pila. El PIC tiene una pila de 8 niveles, por lo que pueden ser llamadas 8 subrutinas sin problemas.

Figura 12. **Memoria de programa PIC 16F877a**

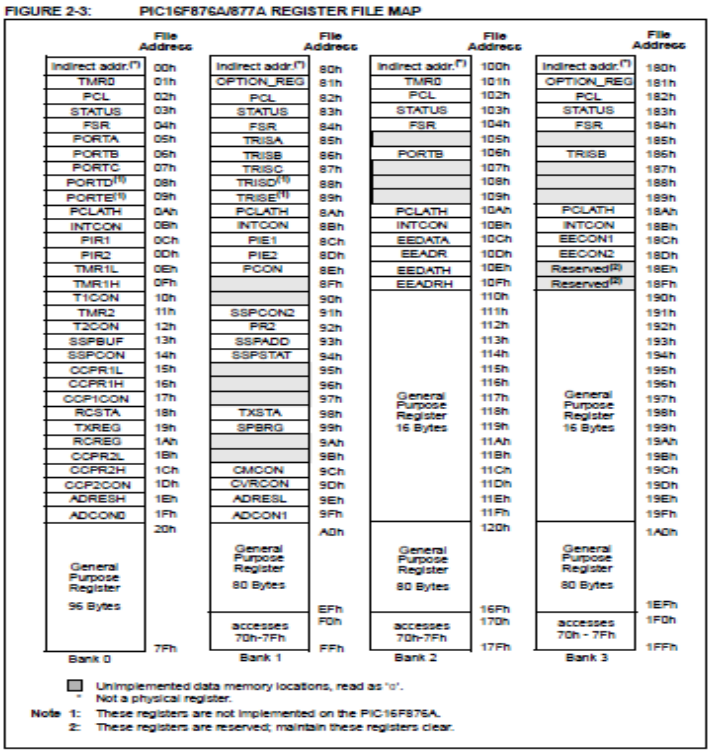


Fuente: *Microchip Technology Inc*. http://ww1.microchip.com/downloads/en/DeviceDoc/39582C.pdf. Consulta: 26 de noviembre de 2016.

1.2.6. Memoria de datos RAM

El PIC 16F877A posee cuatro bancos de memoria RAM, cada banco posee 128 bytes. Los primeros 32 son registros que cumplen un propósito especial en el control del microcontrolador y en su configuración. Los 96 siguientes son registro de uso general que se pueden usar para guardar los datos temporales de la tarea que se está ejecutando. A las posiciones o registros se pueden acceder directamente o indirectamente. Para seleccionar la página o porción de memoria se trabaja en un momento determinado se utilizan los bits RP0 y RP1 del registro STATUS.

Figura 13. **Mapa de los registros memoria RAM**



Fuente: *Microchip Technology Inc*. http://ww1.microchip.com/downloads/en/DeviceDoc/39582C.pdf, Consulta: 26 de noviembre de 2016.

1.2.7. Comunicación serial

El microcontrolador PIC 16F877A tiene integrado el módulo de comunicación básico llamado USART, que no es más que una comunicación serial utilizando el protocolo RS232, lo que permite al microcontrolador una comunicación entre dos dispositivos que utilicen el mismo protocolo.

El módulo USART puede ser configurado como un sistema asíncrono *full-duplex* que puede comunicarse con otros dispositivos periféricos, como terminales CRT y computadoras personales, o puede configurarse como sistema *half-duplex* síncrono que puede comunicarse con dispositivos periféricos tales como circuitos integrados D/A o A/D, memorias EEPROM seriales, etc.

El módulo USART puede ser configurado en los siguientes modos:

- Asíncrono (*full-duplex*)
- Síncrono master (*half-duplex*)
- Síncrono slave (*half-duplex*)

Los pines RC6 corresponden al TX transmisor y el RC7 corresponde al RX receptor en el módulo USART.

1.2.8. Resumen de algunos registros de configuración

- BANCO 0
    - TMR0: registro del temporizador/contador de 8 bits.
    - PCL: byte menos significativo del contador de programa (PC).

- STATUS: contiene banderas (bits) que indican el estado del procesador después de una operación aritmética/lógica.
- FSR: registro de direccionamiento indirecto.
- PORTA, PORTB, PORTC, PORTD, PORTE: registro de puertos de E/S de datos. Conectan con los pines físicos del micro.
- PCLATH: byte alto (más significativo) del contador de programa (PC).
- INTCON: registro de control de las interrupciones.
- ADRESH: parte alta del resultado de la conversión A/D.
- ADCON0: controla la operación del módulo de conversión A/D

- BANCO 1:

  - OPTION: registro de control de frecuencia del TMR0.
  - TRISA, TRISB, TRISC, TRISD. TRISE: registros de configuración de la operación de los pines de los puertos.
  - ADRESL: parte baja del resultado de la conversión A/D.
  - ADCON1: controla la configuración de los pines de entrada análoga.

- BANCO 2:

  - TMR0: registro del temporizador/contador de 8 bits
  - PCL: byte menos significativo del contador de programa (PC)
  - FSR: registro de direccionamiento indirecto
  - EEDATA: registro de datos de la memoria EEPROM
  - EEADR: registro de dirección de la memoria EEPROM
  - PCLATH: byte alto (más significativo) del contador de programa (PC)

- INTCON: registro de control de las interrupciones

- BANCO 3:

  - OPTION: registro de control de frecuencia del TMR0
  - EECON1: control de lectura/escritura de la memoria EEPROM de datos
  - EECON2: no es un registro físico

1.2.9. Resumen de algunas características especiales del PIC 16F877A

- Memoria de programa: FLASH de 8K de instrucciones de 14 bits C/U.
- Memoria de datos: 368 bytes RAM, 256 bytes EEPROM.
- Pila: 8 niveles 14 bits.
- Fuentes de interrupción: 13
- Instrucciones: 15
- Encapsulado: DIP de 40 pines
- Frecuencia oscilador : 20 MHz (máxima)
- Temporizadores/Contadores: 1 de 8 bits (timer 0); 1 de 16 bits (timer 1); 1 de 8 bits (timer 2) con pre y post escalador.
- Perro guardián *watchdog* (WDT)
- Líneas de E/S : 6 del puerto A, 8 del puerto B, 8 del puerto C, 8 del puerto D y 3 del puerto E, además de 8 entradas análogas.
- Dos módulos de captura, comparación y PWM: captura: 16 bits. Resolución máx. = 12,5 nseg.
- Comparación: 16 bits. Resolución máx. = 200 nseg.
- PWM: Resolución máx. = 10 bits.
- Convertidor análogo/digital de 10 bits multicanal (8 canales de entrada).

- Puerto serial síncrono (SSP) con bus SPI (modo maestro) y bus I²C(maestro/esclavo).
- USART (*universal synchronous asynchronous receiver transmitter)* con dirección de detección de 9 bits.
- Corriente máxima absorbida/suministrada (sink/source) por línea (pin): 25 mA.
- Oscilador: soporta 4 configuraciones diferentes: XT, RC, HS, LP.
- Tecnología de fabricación: CMOS.
- Voltaje de alimentación: 3,0 a 5,5 V DC.

1.3. Lenguaje de programación para PIC 16F877A

Para realizar los programas se debe trabajar con un programador de alto nivel, comúnmente se les llama entornos de desarrollo y existe una gran variedad de estos que funcionan en sistemas operativo Windows, Linux, Mac, etc.

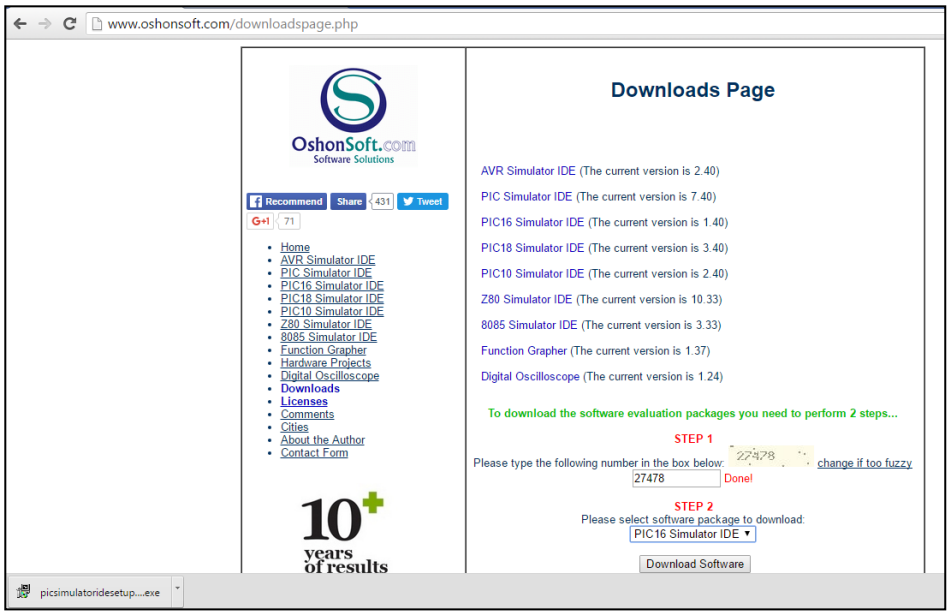
Este *software* permiten crear un código de programación y hay que ser cuidadoso en escoger el *software* ya que este debe incluir todas las librerías de funciones o instrucciones que soporta el microcontrolador PIC 16F877A; se va a utilizar el *software* PIC16 *Simulator* IDE de *OshonSoft;* puede descargarse del siguiente link: OshonSoft.com. Aquí también se encuentra el costo de la licencia y las instrucciones de programación que el microcontrolador entenderá y ejecutará. El sotware PIC 16 Simulator IDE es una herramienta de programación para la gama de microcontroladores de *Microchip* de 8 bits que incluye entorno de desarrollo gráfico fácil de utilizar y puede ser instalado en sistemas operativos *Windows,* este *software* cuenta con un emulador, compilador, ensamblador, desensamblador, depurador de código, etc. Cabe resaltar la característica importante de este *software* que es el emuladores de

PIC, motor pasos a paso, LCD, *hardware* HUART, *software* HUART, terminal para puertos serial, etc. Con este emulador se puede probar el código de programa y verificar que esté funcionando según el criterio de programación; debido a estas características se utilizará este *software*.

1.3.1. Proceso de instalación PIC16 Simulator IDE.

Ingresar a la página oshonsoft.com y luego en la opción descargas (*downloads*); en la página de descargas seguir los pasos 1 y 2: el primero es ingresar el código en el cuadro; el siguiente; escoger el *software* deseado, en este caso se escoge PIC 16 Simulator IDE y se presiona descargar *software* (*download software*) y comenzará la descarga.

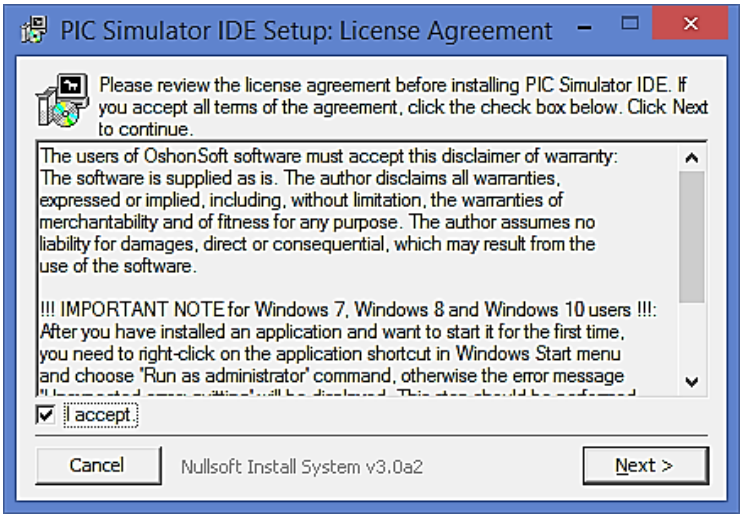
Figura 14. **Proceso de descarga PIC 16 Simulator IDE**



Fuente: elaboración propia, utilizando *software* Labview.

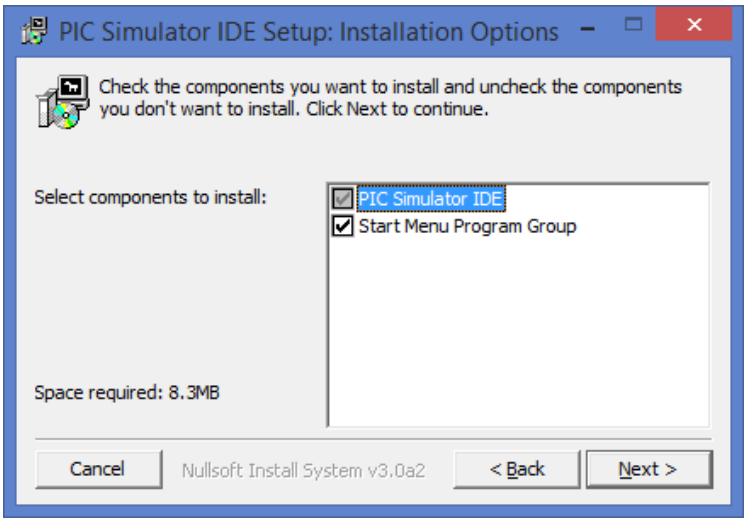
Una vez finalizada la descarga se procede con la instalación.

Figura 15. **Proceso de instalación, aceptar los términos**



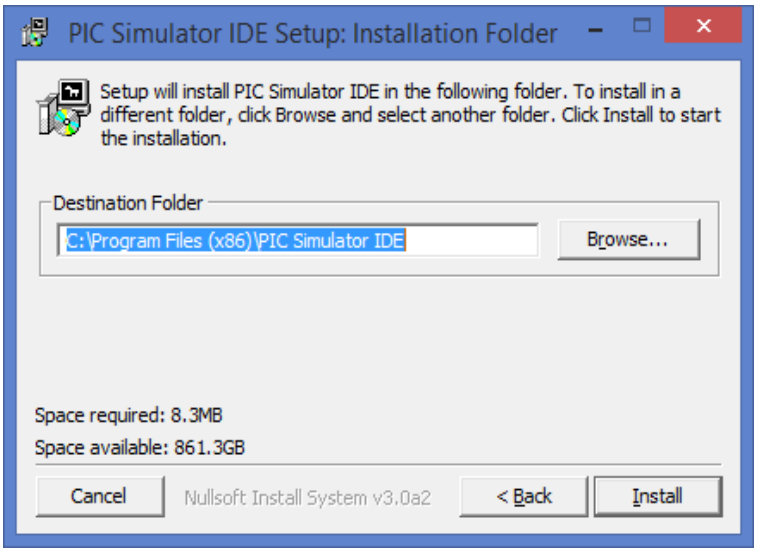
Fuente: elaboración propia, utilizando *software* Labview.

Figura 16. **Componentes que se instalarán**



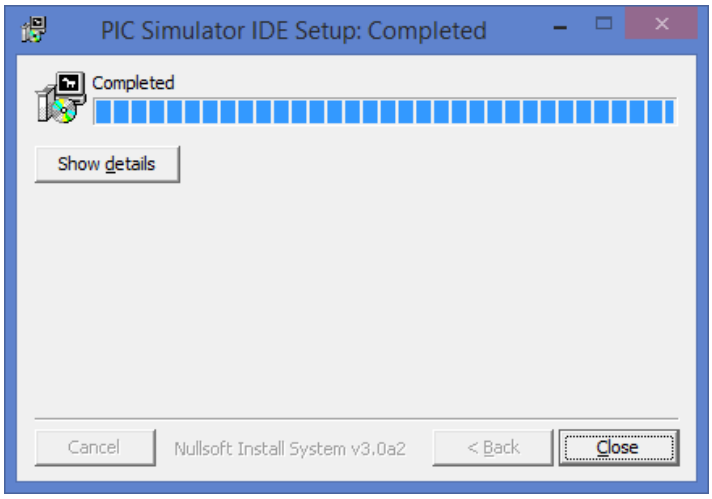
Fuente: elaboración propia, utilizando *software* Labview.

Figura 17. **Carpeta de destino**



Fuente: elaboración propia, utilizando *software* Labview.

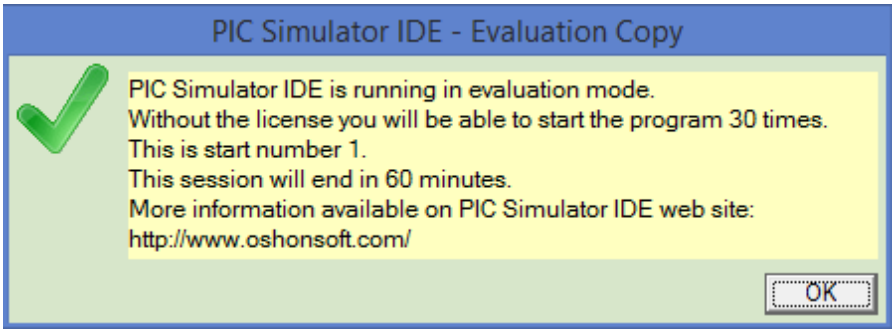
Figura 18. **Instalación completa sin ningún problema**



Fuente: elaboración propia, utilizando *software* Labview.

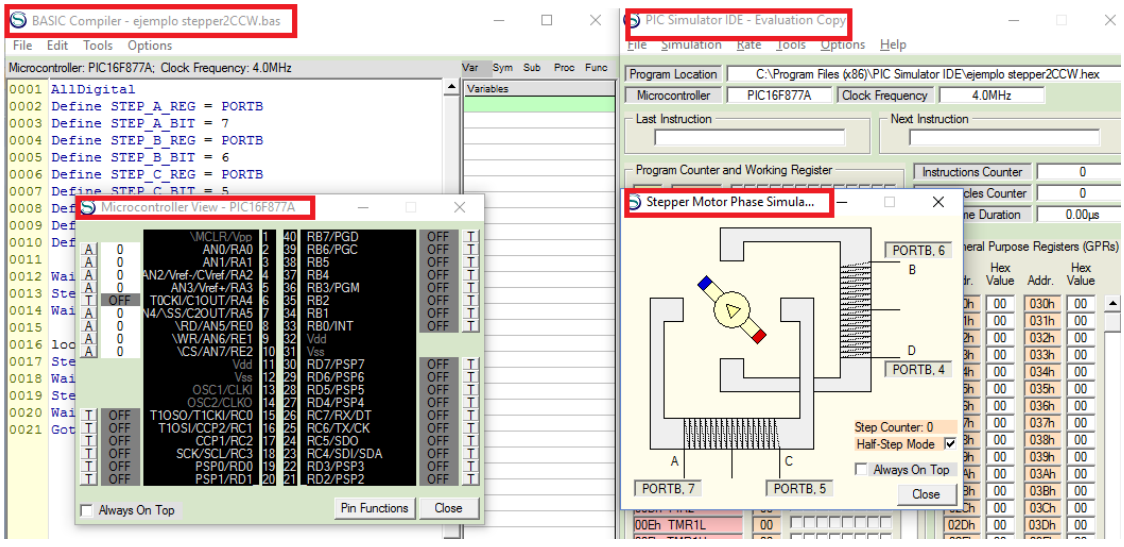
Cuando se abre el sotware PIC Simulator IDE mostrará un mensaje indicando que no se tiene habilitada una licencia y que está en modo de evaluación. Se podra abrir el softwar 30 veces con una duracion de 60 minutos.

Figura 19. **Mensaje de *software* de evaluación PIC Simulator IDE**



Fuente: elaboración propia, utilizando *software* Labview.

Figura 20. ***Software* para programación PIC Simulator IDE**



Fuente: elaboración propia, utilizando *software* Labview.

1.3.2. Instrucciones para programacion del PIC 16F877A

Se detallarán algunas de las instrucciones soportadas por el microcontrolador PIC 16F877A y se utilizará el *software* de desarrollo PIC Simulator IDE para realizar la programación; en la página de OshonSoft se puede encontrar toda ayuda relacionada con la programación del PIC.

Existen 6 tipos de variables que son soportados:

- Bit: 1 bit que puede ser 0 o 1
- Byte: 1 byte, entero en el rango de 0 a 255
- Word: 2 byte, entero en el rango de 0 a 65 535
- Long: 4 bytes, entero en el rango de 0 a 4 294 967 295
- Single: 4 bytes, precisión simple punto flotante, 7 dígitos de precisión
- String: arreglos de bytes que contiene códigos de caracteres ASCII

Las variables pueden ser globales (declaradas en el programa principal antes de la instrucción *end*) o locales (declaradas en subrutinas, procedimientos o funciones); el nombre de la variable global se puede utilizar de nuevo para las variables locales. El compilador reservará locaciones de memoria para ellos. El número total de variables es limitado por la capacidad de la memoria RAM del microcontrolador. Estas variables son declaradas utilizando la sentencia DIM, por ejemplo:

- DIM i As bit
- DIM j As byte
- DIM K As Word
- DIM x As Long

Es posible utilizar *arrays* de una dimensión para *byte*, *long* o *single* variables, por ejemplo:

- Dim c(10) As Byte

  Esta sentencia indica un *array* de 10 variables de un *byte* con índice de matriz en el intervalo [0-9]. Un *array* de variables *byte* puede contener hasta 96 elementos, hasta 48 *arrays* están disponibles para variables Word; para variables *Long* y *Single* el límite es 24 *array*, estos valores, también, dependen de la memoria RAM del microcontrolador.

  Operaciones matemáticas y lógicas: cinco operaciones aritméticas están disponibles en el microcontrolador para datos de tipo enteros. Las operaciones MOD no están disponibles para variables de datos tipo *single*. El compilador puede operar todo tipo de expresión compleja, incluyendo funciones matemáticas complejas, por ejemplo:

  - Dim j As Word
  - Dim i As Word
  - Dim x As Word
  - i=123
  - j=I * 234
  - x=2
  - x=(j*x-12345)/(i+x)

  - Dim j As bit
  - Dim i As bit
  - Dim x As bit
  - x = not i

- x = i and j

- Elementos básico de programación: son los utilizados comúnmente en casi todo ambiente de programación; cuatro sentencias básicas de programación son soportadas: *if-else, goto, while, for,* etc. Se darán ejemplos de cuál es la sintaxis utilizada para cada una.

  - Ejemplo de *IF-Else*:

    - loop:
    - If PORTA.0 Then
    - PORTB.0 =1
    - Else
    - PORTB.0 =0
    - Endif
    - Goto loop:

  - Ejemplo de While:
    - a=10
    - while a>0
    - PORTD=a
    - a=a-1
    - waitMs 100
    - wend

  - Ejemplo de For:
    - TRISB=0
    - For f=0 to 100 step 2
    - PORTB= f

        - Nest f

    - Ejemplo de Case:
        - loop:
        - Select case b
        - Case 10
        - b=1
        - Case <= 100
        - b= b+10
        - Case else
        - b=255
        - EndSelect
        - Goto loop

- Goto: son saltos incondicionales, utiliza el nombre de la línea de etiqueta como argumento donde el nombre de la etiqueta tiene que terminar con ":", ejemplo:

    - Dim f As word
    - f=0
    - ejemplo:
    - f+1
    - goto ejemplo

- Configuración de los pines del microcontrolador PIC 16F877A: la configuración de los pines del microcontrolador puede realizarse de dos maneras, utilizando la sintaxis "**TRISx**" donde x es el registro A, B, C, D, E; o utilizarse la sintaxis CONFIGPIN. A continuación, ejemplos de sintaxis:

- ConfigPin PORTE = Output
- ConfigPin RD7 = Output
- ConfigPin RD = input
- TrisB= 0X00 Output
- TrisC=0XFF Input

- Elementos especiales de programación: las sintaxis WAITUS o WAITMS se pueden utilizar para forzar al programa a que espere ya sea en milisegundos o microsegundos, tomar en consideración cuando se realiza una simulación deberíamos utilizar WAITUS para no forzar al procesador de la computadora, pero si estamos escribiendo programas para un dispositivo real deberíamos considerar el uso del WAITMS, para un segundo son 1000 milisegundos, por ejemplo:

  - Dim a As Word
  - a= 100
  - WaitMS a

- Uso de la memoria interna EEPROM: para acceder a la memoria del microcontrolador puede ser usadas las sentencias de programación READ y WRITE; el primer argumento es la dirección de un byte en la memoria EEPROM y puede ser una constante o una variable, el segundo argumento es la información que está siendo leída o escrita.

  - Dim x As Byte
  - Dim y As Byte
  - Read 10,x
  - Write 11,y

- Convertidor analógico-digital A/D: la sentencia ADCIN está disponible como un soporte para el convertidor interno A/D; en el primer argumento indicamos el número de canal a utilizar y el segundo es una variable donde almacenaremos el resultado de la conversión analógico a digital. La sentencia ADCIN utiliza dos parámetros ADC_CLOCK y ADC_SAMPLEUS los valores por defecto son 3 y 20. los valores por defecto pueden ser cambiados utilizando los directiva DEFINE, ADC_CLOCK puede configurarse con los valores de 0-3 o 0-7 y ADC_SAMPLEUS puede configurarse con tiempos de adquisición dados en microsegundos entre 0-255; para configurar el pin en el microcontrolador que hará la captura se utiliza la sentencia TRIS y ADCON1 para configurar como entrada analógica, a continuación un ejemplo:

```
Dim v(5) As Byte
Dim vm As Word
Dim i As Byte
Define ADC_CLOCK = 3
Define ADC_SAMPLEUS = 50
TRISA = 0xff
TRISB = 0
ADCON1 = 0
For i = 0 To 4
Adcin 0, v(i)
Next i
vm = 0
For i = 0 To 4
vm = vm + v(i)
Next i
```

- vm = vm / 5
- PORTB = vm.LB

- Módulo interno UART: la comunicación serial es soportada por ambas maneras por *software* y por hardware, HSEROPEN, HSEROUTE, HSERIN, HSERGET, son los comandos que pueden ser utilizados con dispositivos PIC UART. Todas las velocidades en el rango de transmisión 100-200000 Baudios son aceptadas, pero se sugiere utilizar las velocidades de transmisión estándar 300, 600, 1 200, 2 400, 4 800, 9 600, 14 400, 19 200, 28 800, 31 250, 38 400, 56 000 o 57 600 y si no se declara la sentencia UART se utiliza por defecto la velocidad 9 600, si el argumento ALLOW_MULTIPLE_HSEROPEN se fija en 1 usando la directiva DEFINE podrá utilizarse la sentencia HSEROPEN más de una vez en el programa para cambiar al velocidad en baudios.

- La sentencia HSEROUT es usada para la transmisión en serie y contiene múltiples argumentos separados por ","; se pueden utilizar variables tipo cadenas o constantes, variables numéricas y constantes numéricas tipo byte, el signo "#" es usado antes de la variable y, luego, la representación decimal tipo cadena es enviada por el puerto serial.

- La sentencia HSERIN se utiliza para cargar una lista de variables de tipo de datos numéricos con los valores recibidos en el puerto en serie. Esta declaración espera hasta que se recibe el número necesario de bytes en el puerto serie. La sentencia HSERGET tiene un argumento que debe ser una variable Bbyte. Si hay un carácter esperando en el búfer de entrada, lo cargará en la variable, de lo contrario se cargará el valor 0. A continuación un ejemplo:

- Dim i As Byte
- Hseropen 38400
- WaitMs 1000
- For i = 20 To 0 Step -1
- Hserout "Number: ", #i, CrLf
- WaitMs 500
- Next i

- Comunicación serial por *software* UART: en todos los PIC es soportada la comunicación serial y es implementada por las declaraciones SEROUT y SERTIN; en ambos, el primer argumento después de la sentencia debe ser el pin del microcontrolador y el segundo es la velocidad; todos los rangos de velocidad son soportados por el microcontrolador 100-60 000, pero se sugiere utilizar velocidades estándares 300, 600, 1 200, 2 400, 4 800 o 9 600; si se utilizan velocidades altas con frecuencias de reloj más bajas, puede provocar errores de sincronización y de encuadre inexactas.

- La sentencia SEROUT es usada para la transmisión en serie y contiene múltiples argumentos separados por ","; se pueden utilizar variables tipo cadenas o constantes, variables numéricas y constantes numéricas tipo byte, el signo "#" es usado antes de la variable y luego la representación decimal tipo cadena es enviada por el puerto serial.

- La sentencia SERIN se utiliza para cargar una lista de variables de tipo de datos numéricos con los valores recibidos en el puerto en serie. Esta declaración espera hasta que se recibe el número necesario de bytes en el puerto serie, a continuación un ejemplo.

- Dim i As Byte
- loop:
- Serin PORTC.7, 9600, i
- Serout PORTC.6, 9600, "Number: ", #i, CrLf
- Goto loop

- Interfaz para motores *stepper*, antes de utilizar las sentencias para configurar un puerto para que funcione como motor paso a paso se debe conocer su conexión y el modo de funcionamiento del motor *stepper*, para establecer el funcionamiento del motor hay 8 parámetros disponibles que definen la conexión de las bobinas. A, B, C y D:

    - STEP_A_REG: define el puerto donde la bobina A está conectada
    - STEP_A_BIT: define el pin donde la bobina A está conectada
    - STEP_B_REG: define el puerto donde la bobina B está conectada
    - STEP_B_BIT: define el pin donde la bobina B está conectada
    - STEP_C_REG: define el puerto donde la bobina C está conectada
    - STEP_C_BIT: define el pin donde la bobina C está conectada
    - STEP_D_REG: define el puerto donde la bobina D está conectada
    - STEP_D_BIT: define el pin donde la bobina D está conectada

Las bobinas A y C son partes de una bobina simple con conexión común al igual que las bobinas B y D. Hay otro parámetro utilizado STEP_MODE que define el avance del motor si está configurado, en 1 el motor funcionará como paso completo. El valor 2 debería ser utilizado para medio paso. La sentencia STEP_HOLD configurará los puertos como salidas y energizará las bobinas A y B para que esté el motor en su posición inicial; el movimiento del rotor puede configurarse en contra de las macillas del reloj o a favor de las manecillas del reloj, las sentencias para configurar son STEPCW y STEPCCW,

respectivamente, estas sentencias tienen dos argumentos separados por coma (,): el primero define la cantidad de pasos y puede ser una constante tipo byte o una variable; el segundo define el tiempo entre cada paso y puede ser una constante tipo byte o una variable. A continuación, un ejemplo:

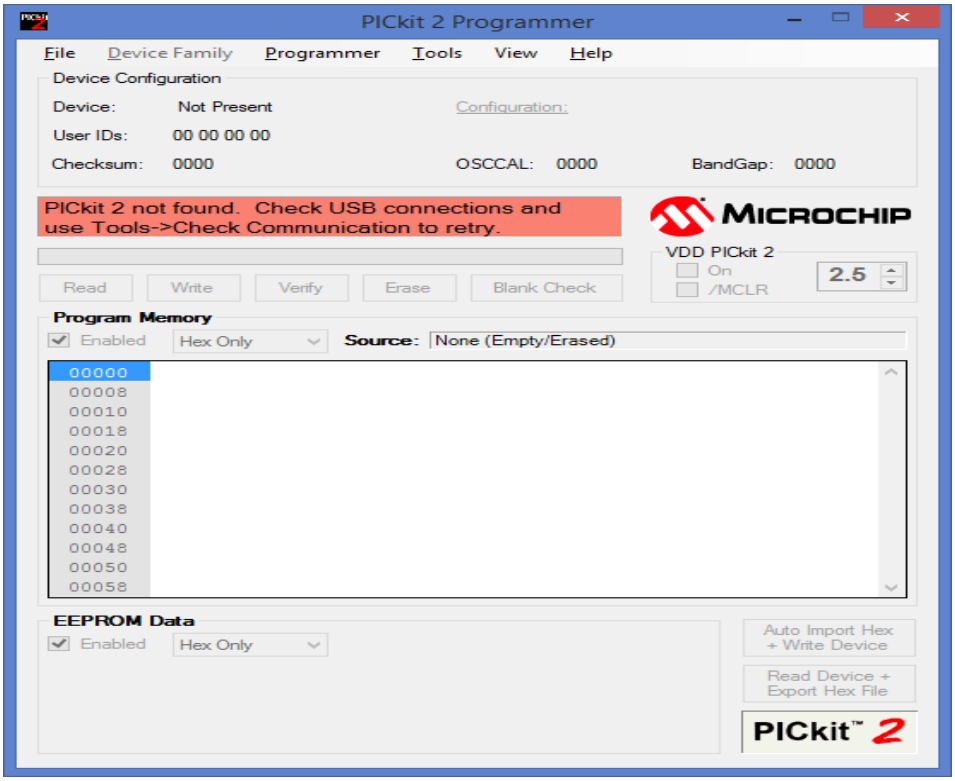
- all Digital
- Define STEP_A_REG = PortB
- Define STEP_A_BIT = 7
- Define STEP_B_REG = PortB
- Define STEP_B_BIT = 6
- Define STEP_C_REG = PortB
- Define STEP_C_BIT = 5
- Define STEP_D_REG = PortB
- Define STEP_D_BIT = 4
- Define STEP_MODE = 2
- WaitUs 100
- Step Hold
- WaitUS 100
- ciclo:
- StepCCW 16,200
- StepCW 30,500
- goto ciclo:

1.3.3. Grabación del microcontrolador PIC 16F877A

Una vez compilado el código de programación, el *software* de programación y simulación creará un archivo con extensión. HEX el cual debe cargarse al microcontrolador ya que este solo entiende leguaje binario: unos (1) y ceros (0).

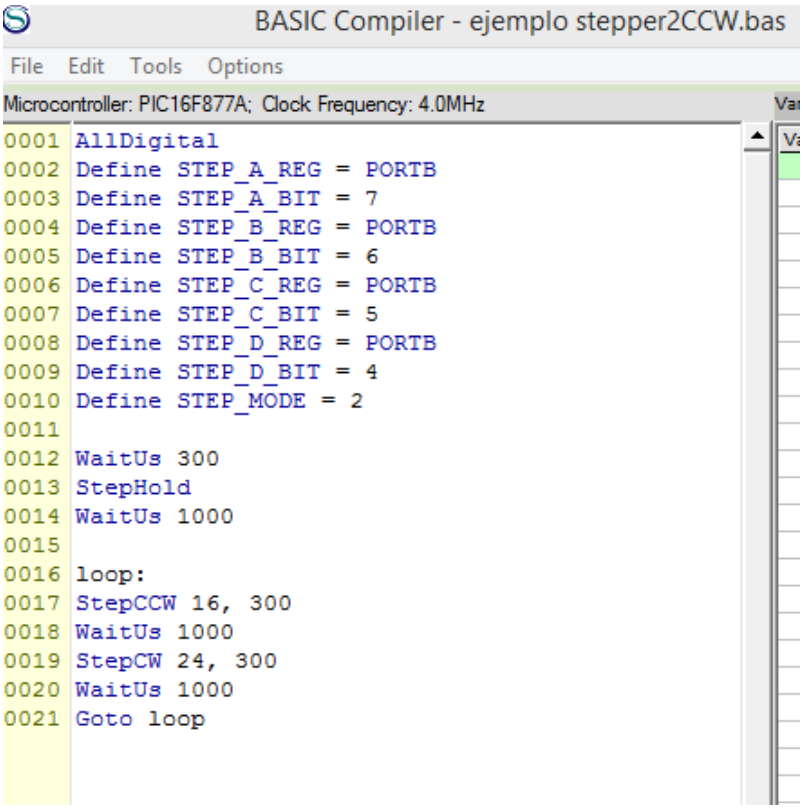
Para cargar o grabar el programa al microcontrolador se utiliza el *software* PICkit2 *Programmer* desarrollado por *Microchip Technology* Inc. el cual permite cargar el archivo con extensión .HEX al microcontrolador pero antes de iniciar la programación se verifica la conexión con la programadora USB, se debe configurar: tipo de oscilador, tipo de dispositivo y frecuencia del reloj; una vez configurados estos parámetros se procede a borrar y verificar que esté en blanco y por último la carga del archivo. A continuación, en la figura 21 se muestra el programa PICkit2 y en la figura 22 el código de programa para el movimiento de un motor stepper.

Figura 21. **Programadora de PIC**



Fuente: elaboración propia, utilizando *software* Labview.

Figura 22. **Programa de para movimiento de Stepper**



Fuente: elaboración propia, utilizando *software* Labview.

1.4. Motores paso a paso o *stepper*

Los motores paso a paso son dispositivos ideales para construir mecanismos donde se requieren movimientos muy precisos. Las características especiales es que se pueden mover un paso a la vez por cada pulso que se aplique; los pasos pueden variar de desde 90° hasta pequeños pasos de 1.8°. Para el primer caso se necesitan 4 pasos para completar una revolución o lo que es igual a 360°; para el segundo caso se necesitan 200 pasos para completar una revolución.

Figura 23. **Motor paso a paso**

Fuente: *Motor paso a paso*. https://es.wikipedia.org/wiki/Motor_paso_a_paso#/media/File:Motori_passo-passo.jpg, Consulta: 26 de noviembre de 2016.

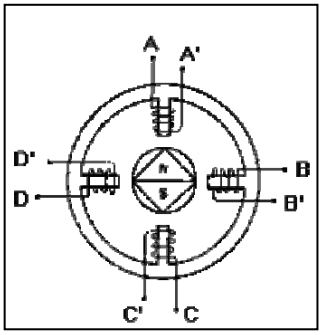
1.4.1. Principio de funcionamiento

Están construidos normalmente por un rotor sobre el que están aplicados distintos imanes permanentes y por un cierto número de bobinas excitadoras en su estator; las bobinas son parte del estator y el rotor es de imán permanente, toda la excitación de las bobinas se debe realizar por un conmutador manejada por un controlador que puede ser un microcontrolador.

1.4.2. Motores de imán permanente

Estos tienen rotores de imán permanente sin dientes y son magnetizados perpendicularmente al eje; energizando en secuencia, el rotor es atraído por los polos magnéticos; estos motores tienen pasos de 90 grados si la secuencia de alimentación de la bobina es ABCD. Estos motores tienen pasos de ángulos entre 45 y 90 grados y la velocidad de pasos es relativamente baja, par alto y buenas características de amortiguamiento.

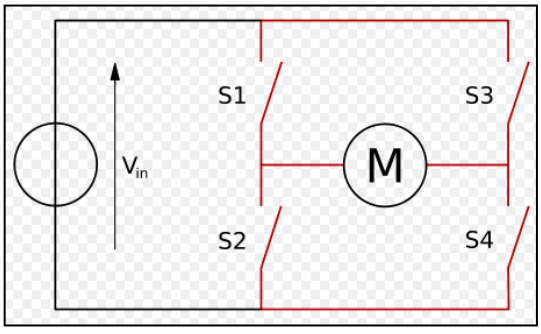
Figura 24. **Motor de imán permanente**



Fuente: CANTO, Carlos. *Motores de paso - fc-uaslp*. http://galia.fc.uaslp.mx/~ canto car/ microcontroladores/SLIDES_8051_PDF/21_MOTOR.PDF. Consulta: 27 de noviembre de 2016.

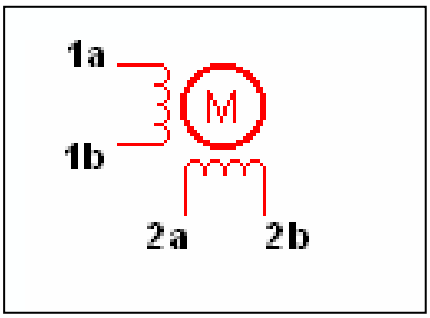
Motor paso a paso bipolar: normalmente tienen cuatro cables de salida, requieren el cambio de flujo de corriente a través de las bobinas y es necesario una secuencia apropiada para realizar un movimiento. Mediante el uso de un puente H puede controlarse el flujo de corriente en la bobina y es necesario un puente H por cada bobina del motor.

Figura 25. **Puente H**



Fuente: *Puente H. (electrónica)*. https://es.wikipedia.org/wiki/Puente_H_(electr%C3%B3nica). Consulta: 27 de noviembre de 2016.

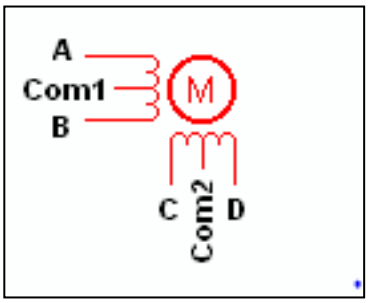
Figura 26. **Motor paso a paso imán permanente bipolar**



Fuente: *Motores paso a paso unipolares y bipolares.* http://www.forosdeelectronica.com/f16/motores-paso-paso-unipolares-bipolares-tutorial-13284/, Consulta: 28 de noviembre de 2016.

Motor paso a paso unipolar: estos motores comúnmente tienen 5 a 6 cables de salida por su conexión interna este es más sencillo de controlar y se debe a que tienen un terminal común a ambas bobinas; una forma de identificar cada uno de los cables es analizar la forma de conexión interna de este tipo de motor.

Figura 27. **Motor paso a paso imán permanente unipolar**



Fuente: *Motores paso a paso unipolares y bipolares*. http://www.forosdeelectronica.com/f16/motores-paso-paso-unipolares-bipolares-tutorial-13284/, Consulta: 28 de noviembre de 2016.

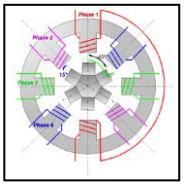
Los dos bobinados del motor se encuentran separados, pero tienen una terminal central, la cual se llamará común.

1.4.3. Motores paso a paso de reluctancia variable

La expresión motor de reluctancia variable hace referencia a un motor eléctrico del tipo paso a paso, cuyo funcionamiento se basa en la reluctancia variable mediante un rotor dentado en hierro dulce que tiende a alinearse con los polos bobinados del estator. El rotor es de material magnético, pero no es un imán permanente, presenta una forma dentada con salientes. El estator consiste en una serie de piezas polares conectadas a 3 fases.

En todo momento, el rotor buscará alinearse de forma tal que minimice la reluctancia rotor-estator, circunstancia que se da cuando el espacio entre los polos del estator queda lo más ocupado posible por material del rotor, es decir, orientando los salientes o dientes hacia los polos energizados del estator. Este tipo de motor puede diseñarse para funcionar con pasos más pequeños que los pasos de un motor paso a paso de imán permanente. Por otra parte, su rotor es de baja inercia, con lo cual mejora su respuesta dinámica, aunque tiene la desventaja de tener menor por motor que un motor eléctrico de imán permanente de similar tamaño.

Figura 28. **Motor de reluctancia variable**

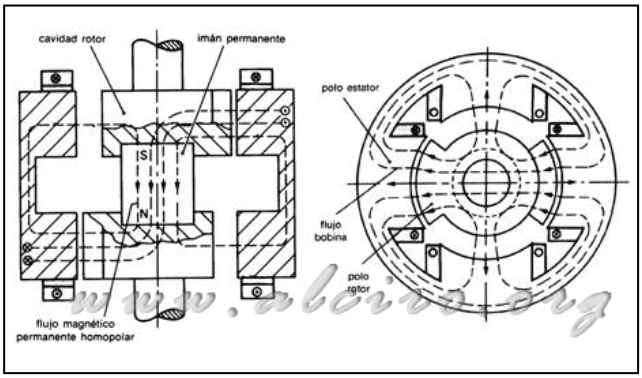


Fuente: *Motor de reluctancia variable.* https://es.wikipedia.org/wiki/Motor_de_reluctancia_variable. Consulta: 28 de noviembre de 2016.

1.4.4. Motores paso a paso híbridos

Combinan las cualidades de los motores de reluctancia variable y los de imán permanente; los motores híbridos tienen cualidades deseables de cada uno, tienen un par alto de reten, un excelente par de sostenimiento y dinámico, pueden ser operados a una alta velocidad de pasos, normalmente tienen pasos entre 0,9 a 5 grados.

Figura 29. **motor híbrido**



Fuente: *Motor de reluctancia variable*. http://www.alciro.org/alciro/Plotter-Router-Fresadora-CNC_1/Motores-hibridos_158.htm. Consulta: 29 de noviembre de 2016.

1.4.5. Secuencia para manejar motores paso a paso bipolares y unipolares

Como se mencionó anteriormente, estos motores necesitan que el flujo de corriente en una bobina cambie en una determinada secuencia cada inversión en la polaridad provoca el movimiento de un paso, la siguiente secuencia determina el paso del motor y hacia donde gira. En la siguiente tabla (Tabla I) está la secuencia para controlar los pasos de un motor bipolar.

Tabla I. **Secuencia de motor paso a paso bipolar**

Giro A favor de los punteros del reloj ↓

Paso	1a	1b	2a	2b
1	+	-	+	-
2	+	-	-	+
3	-	+	-	+
4	-	+	+	-

↑ Giro en contra de los punteros del reloj

Fuente: elaboración propia.

Para mantener el giro del motor es necesario repetir la secuencia de pasos y para invertir el giro basta con ejecutar la secuencia de forma inversa. Para los motores unipolares paso a paso existen tres posibles secuencias:

- Secuencia doble paso: es la secuencia más utilizada; en ésta el motor avanza paso por paso, como siempre hay dos bobinas activadas se obtiene un alto torque de paso y de retención, pero los pasos son más bruscos debido a que el campo magnético es mayor.

Tabla II. **Secuencia doble paso motor paso a paso unipolar**

	Nombre de la bobina			
Paso	**A**	**B**	**C**	**D**
1	ON	ON	OFF	OFF
2	OFF	ON	ON	OFF
3	OFF	OFF	ON	ON
4	ON	OFF	OFF	ON

ON: bobina encendida; OFF: bobina apagada

Fuente: elaboración propia.

- Secuencia paso simple: en esta secuencia se activa solo una bobina a la vez. En algunos motores esto brinda un funcionamiento más suave. Pero al estar solo una bobina activada, el torque de paso y retención es menor.

Tabla III. **Secuencia paso simple motor paso a paso Unipolar**

	Nombre de la bobina			
Paso	**A**	**B**	**C**	**D**
1	ON	OFF	OFF	OFF
2	OFF	ON	OFF	OFF
3	OFF	OFF	ON	OFF
4	ON	OFF	OFF	ON

ON: bobina encendida; OFF: bobina apagada

Fuente: elaboración propia.

- Secuencia tipo medio paso: es la más simple, en esta secuencia se activan las bobinas de tal forma de brindar un movimiento igual a la mitad

del paso real. Para lo cual se activan primero 2 bobinas y luego solo 1 y así sucesivamente. La secuencia completa consta de 8 movimientos en lugar de 4.

-

Tabla IV. **Secuencia medio paso motor paso a paso Unipolar**

	Nombre de la bobina			
Paso	**A**	**B**	**C**	**D**
1	ON	OFF	OFF	OFF
2	ON	ON	OFF	OFF
3	OFF	ON	OFF	OFF
4	OFF	ON	ON	OFF
5	OFF	OFF	ON	OFF
6	OFF	OFF	ON	ON
7	OFF	OFF	OFF	ON
8	ON	OFF	OFF	ON

ON: bobina encendida; OFF: bobina apagada

Fuente: elaboración propia.

2. PROCESAMIENTO DIGITAL DE IMÁGENES

Se presenta de manera breve una explicación de la visión, las imágenes digitales y cómo son capturadas por medios electrónicos, el campo encargado del procesamiento digital de imágenes (DIP) y sus respectivas técnicas. El procesamiento digital de imágenes es un campo de investigación abierto y el progreso en esta área no ha sido por sí mismo; al contrario, se ha logrado en conjunto con otras áreas: matemática, computación y conocimiento cada vez más avanzado de ciertos órganos del cuerpo humano que intervienen en la percepción y en la manipulación de las imágenes.

La inquietud de la humanidad por tratar de imitar y usar ciertas características del ser humano como apoyo en la solución de ciertos problemas, ha permitido el avance en el procesamiento digital de imágenes y se ve reflejado en medicina, astronomía, geología, microscopía, información meteorológica y ahora en el campo de la energía eléctrica, donde se utilizarán técnicas de procesamiento digital de imágenes para extraer datos característicos de los medidores de energía eléctrica. El campo de la visión artificial está abierto a brindar muchas soluciones en la industria y las aplicaciones están en crecimiento a tal punto que procesos críticos o peligrosos realizados por el ser humano están siendo reemplazados por estos sistemas de visión artificial.

2.1. Sistema de visión humano

La vista es uno de los sentidos más complejos y especializados del cuerpo humano; por medio de la visión, el ser humano es capaz de captar forma,

volumen, colores, luminosidad, tamaño y todas las demás cualidades de un objeto.

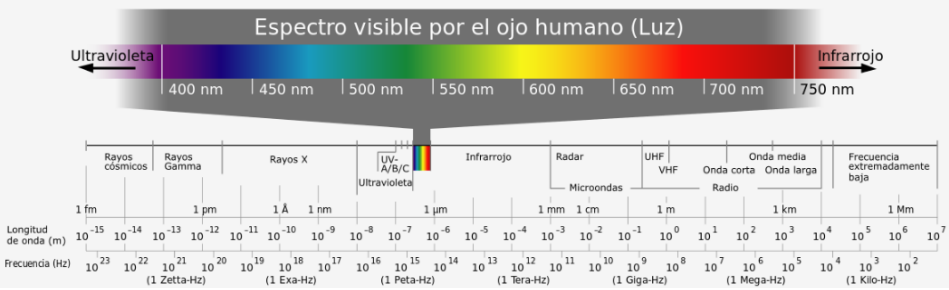
2.1.1. Percepción de la luz por el ojo humano

Para la percepción de los colores, el ojo humano presenta dos tipos de sensores denominados células foto-receptoras, estas células se dividen en conos y bastones: los primeros son sensibles a los colores en diferentes proporciones, rojo 65 %, verde 33 % y azul el 2 %; mientras que los bastones tienen una respuesta espectral menos selectiva pero más sensible, como es el caso de la visión nocturna. De esta forma el ojo humano percibe los estímulos de los 3 colores, todo esto por acción de los conos que actúan como filtro de los estímulos que llegan al ojo humano y la sensación de color es la respuesta aditiva de estos tres colores.

2.1.2. Luz visible por el ojo humano

La luz visible es una radiación electromagnética cuya longitud de onda pertenece a una porción del espectro electromagnético, en el cual la retina del ojo humano exhibe una respuesta; la gama normal de longitud de onda de visión humana oscila entre 390 nm y 780 nm aproximadamente. En la figura 30 se muestra el espectro electromagnético y el rango de frecuencias y longitud de onda de los distintos colores.

Figura 30. **Espectro electromagnético**



Fuente: *Espectro electromagnético*. https://es.wikipedia.org/wiki/Espectro_electromagn%C3%A9tico, Consulta: 7 de noviembre de 2016.

Las tres dimensiones básicas de la luz son:

- Intensidad (brillantez o amplitud) que está asociada a la percepción visual del flujo luminoso que emite o refleja un objeto.

- Frecuencia o longitud de onda, percibida por el ojo humano como color de la luz.

- Polarización o ángulo de vibración que no es perceptible por los humanos bajo circunstancias ordinarias

2.1.3. Dispositivos para la adquisición de imágenes

Un sistema de adquisición de imágenes debe contar con tres elementos:

- Fuente de luz: para iluminar la escena
- Cámara: dispositivo para capturar imágenes
- Interfaz entre el computador y el sensor (cámara)

- Iluminación: un sistema de iluminación es muy importante para un sistema de visión artificial; una adecuada iluminación permite resaltar características de interés del objeto, reducir la complejidad de la imagen a analizar y mejorar el tiempo de respuesta del procesamiento digital.

Existen varias técnicas de iluminación y se debe utilizar la técnica que más se adecúe a nuestra necesidad y a la característica que se desee resaltar del objeto; además de la técnica de iluminación, también, hay que considerar el tipo de fuente de luz.

Existen varios factores que deben considerarse al momento de decidir el tipo de fuente de luz para iluminación del ambiente: intensidad lumínica, tiempo de vida útil, flexibilidad del diseño y precio. A continuación, se detallan las fuentes lumínicas más utilizadas.

- Lámpara de tungsteno: económicas, pero poco convenientes en el procesamiento de imágenes, especialmente cuando la frecuencia de captura de la cámara no es múltiplo de la frecuencia de la red eléctrica (60 Hz para América); cuando la frecuencia de la fuente y la captura de la cámara difieren, provoca interferencia en la captura que hace que aparezcan franjas más claras u oscuras en la imagen capturada, afectando la calidad de la imagen.

- Lámparas fluorescentes: iluminación muy homogénea, pueden ser operados con rectificadores de frecuencia para prevenir la modulación de la intensidad de la luz y así evitar interferencia, no disipan calor. Una de las desventajas es que no presentan un balance de color uniforme, las longitudes de onda son mayoritariamente azules y deben funcionar a alta

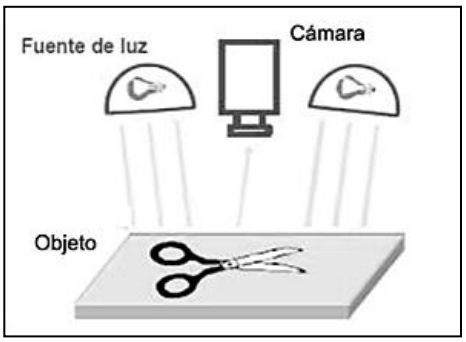
frecuencia (25 kHz) para que el efecto parpadeo no afecte la calidad de la captura.

- Diodos emisores de luz: una mayor respuesta que las lámparas fluorescentes y los halógenos del orden de algunos microsegundos lo cual lo hace ideal para trabajar en sistemas estroboscópicos (destellos), funcionan a baja tensión, disipan poco calor al ambiente, son pequeños, consumen poca potencia y tienen un tiempo de vida prolongado. Se pueden encontrar en el mercado diodos con diferentes longitudes de onda: infrarrojos, ultravioleta, rojos, verdes, ámbar, blanco. El infrarrojo es el más empleado en escenarios donde se requiere eliminar por completo la influencia de la luz del día y de las fuentes de luz artificiales cercanas al sistema. Por todas estas ventajas el diodo de luz se ha popularizado en las diferentes aplicaciones industriales de los sistemas de visión artificial.

Elegir la fuente de luz es tan solo una parte del diseño del sistema de iluminación, debe seleccionarse la forma de iluminar la escena y el objeto de interés, resaltar ciertas características. Entre las técnicas de iluminación se tienen las siguientes:

- Iluminación direccional frontal: se utiliza en objetos planos con superficies mates, la cámara se ubica en la misma dirección de la luz y recibe la luz reflejada por el objeto; el tipo de fuente de luz que se utiliza con esta técnica son los LED y fuentes de fibra óptica;

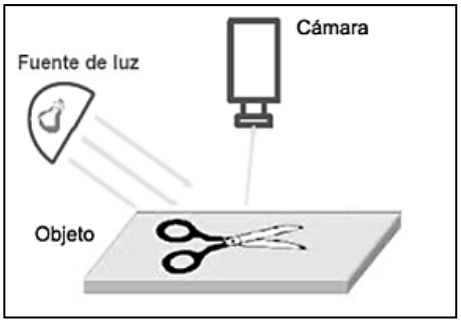
Figura 31. **Iluminación direccional frontal**



Fuente: OTINIANO, Carlos. *Detección de material articulado en suspensión en ambientes de baja iluminación, mediante el procesamiento digital de imágenes*. https://decibel.ni.com/content/docs/DOC-31879. Consulta: 7 de noviembre de 2016.

- Iluminación lateral: se emplea para mostrar laterales de los objetos, estructura superficial, pero produce problemas con los efectos de la sombra, debido a que con esta técnica de iluminación la luz incide a un determinado ángulo.

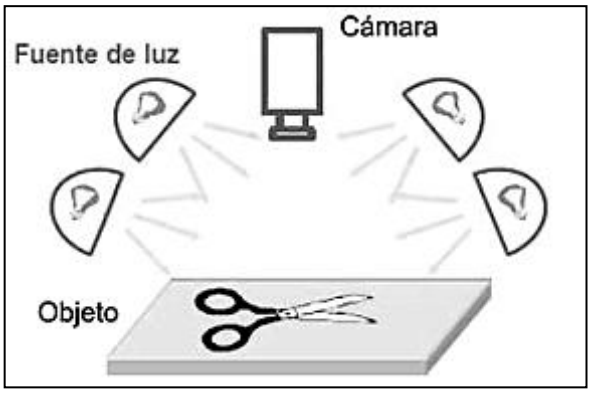
Figura 32. **Iluminación lateral**



Fuente: OTINIANO, Carlos. *Detección de material articulado en suspensión en ambientes de baja iluminación, mediante el procesamiento digital de imágenes*. https://decibel.ni.com/content/docs/DOC-31879. Consulta: 7 de noviembre de 2016.

Iluminación frontal axial difusa: se utiliza para todo tipo de objetos, sin importar que sean mates o brillantes, como fuente de luz pueden emplearse lámparas que proporcionen una luz difusa y uniforme, como los focos incandescente, fluorescente, etc.; hay que tomar en consideración que este método no puede ser utilizado en espacios reducidos por la gran cantidad de fuentes de luz.

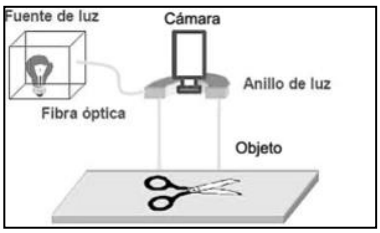
Figura 33. **Iluminación frontal axial difusa**



Fuente: OTINIANO, Carlos. *Detección de material articulado en suspensión en ambientes de baja iluminación, mediante el procesamiento digital de imágenes*. https://decibel.ni.com/content/docs/DOC-31879. Consulta: 7 de noviembre de 2016.

- Iluminación coaxial: se emplea una luz difusa y uniforme en objetos mates o brillantes para todo tipo de inspección, pero la luz debe incidir al objeto desde el mismo eje de la cámara, la iluminación se genera mediante el uso de anillos de luz o divisores de haz, estas condiciones pueden representar un alto coste y difícil de montar.

Figura 34. **Iluminación coaxial**



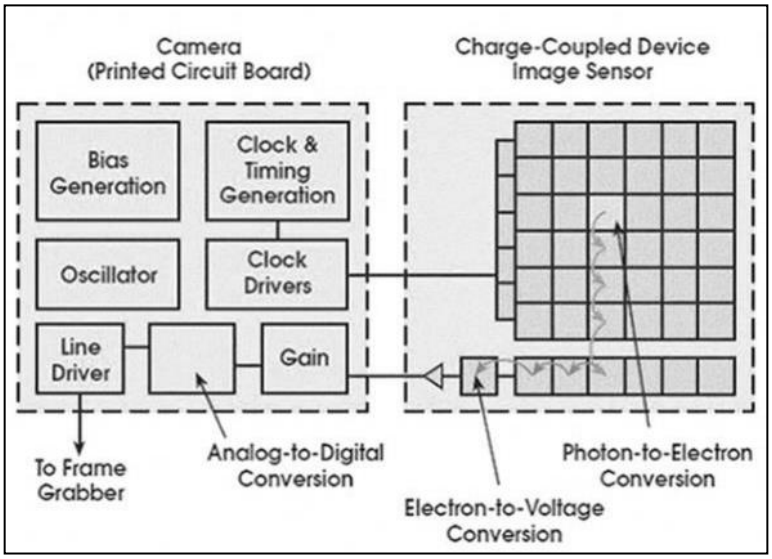
Fuente: OTINIANO, Carlos. *Detección de material articulado en suspensión en ambientes de baja iluminación, mediante el procesamiento digital de imágenes*. https://decibel.ni.com/content/docs/DOC-31879. Consulta: 7 de noviembre de 2016.

- Cámara: el sensor es considerado el corazón de la cámara digital, se clasifican de acuerdo al tipo de sensor que se emplea. Existen dos tipos de tecnologías utilizadas en la fabricación de sensores de cámaras digitales: los *charge couple device* (CCD) y los *complementary metal oxide semiconductor* (CMOS); ambos están conformados por semiconductores de metal-óxido (MOS) y están distribuidos en forma de matriz. La función de los sensores es acumular una carga eléctrica en cada una de las celdas de esta matriz, estas celdas son los llamados pixeles. La carga eléctrica de cada pixel dependerá de la luz que incide en este.

- Sensor CCD *(charge couple device)*: convierte las cargas eléctricas en voltajes y entrega una señal analógica a la salida que posteriormente será digitalizada por la cámara; luego, esta señal es procesada por un convertidor analógico a digital que los convierte en datos binarios. La estructura interna del sensor es muy simple, pero tienen como inconveniente que necesita un chip que se encargue del tratamiento de

la información proporcionada por el sensor lo cual impacta en el tamaño y costo.

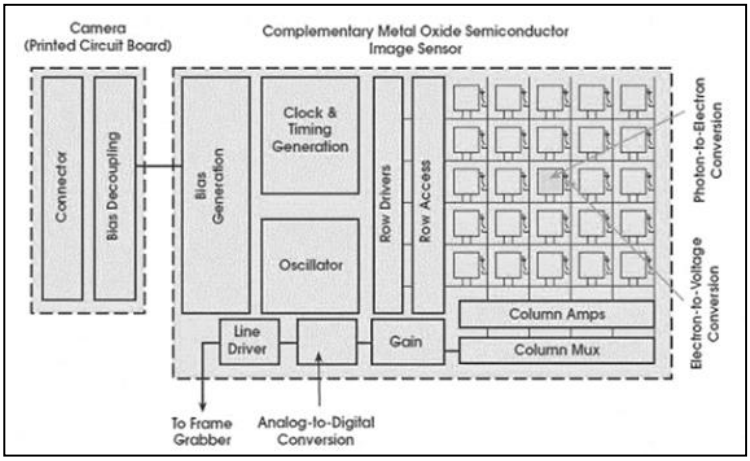
Figura 35. **Estructura interna cámara digital con sensor CCD**



Fuente: OTINIANO, Carlos. *Detección de material articulado en suspensión en ambientes de baja iluminación, mediante el procesamiento digital de imágenes*. https://decibel.ni.com/content/docs/DOC-31879. Consulta: 7 de noviembre de 2016.

- Sensor CMOS (*complementary metal oxide semiconductor*): la digitalización de los pixeles se realiza mediante los transistores que tiene cada celda, por lo cual todo el trabajo se realiza en el sensor y no es necesario un chip externo, con esto se consigue reducir los costos y tamaños; adicional, estos sensores son más sensibles a la luz debido a que los amplificadores se encuentran en la propia celda.

Figura 36. **Estructura interna de la cámara digital con sensor CMOS**



Fuente: OTINIANO, Carlos. *Detección de material articulado en suspensión en ambientes de baja iluminación, mediante el procesamiento digital de imágenes*. https://decibel.ni.com/content/docs/DOC-31879. Consulta: 7 de noviembre de 2016.

2.2. Representación de una imagen digital

Una imagen es una representación visual de un objeto y está definida como una función bidimensional f (x,y), donde "x" y "y" son coordenadas espaciales y la amplitud f en un par de coordenadas (x,y) es la intensidad o nivel de grises de la imagen en ese punto. Una imagen digital es definida como una matriz cuyos valores de filas y columnas identifican un punto en la imagen y su valor identifica la intensidad luminosa; el tamaño de la matriz es lo que define la resolución de la imagen, es decir, con cuantos pixeles se representará la imagen.

En el caso de que las imágenes sean en escala de grises, cada elemento de la matriz representa el valor de gris. El valor que pueda tomar cada elemento va desde 0 (negro) hasta 255 (blanco), 256 escalas de intensidad, esta

restricción se basa en que el ojo humano es sensible y puede detectar 256 diferentes niveles de intensidad de un color; además, 256 es un valor manejable por el computador ya que puede ser representado por un byte. El número de bits que se utilizan para codificar el valor de un pixel es lo que se conoce como la definición de una imagen.

2.2.1. Tipos de imágenes

Para representar imágenes a color se pueden utilizar modelos de colores, tales como el RGB (rojo, verde, azul), CMY (cían, magenta, amarillo), HSL, HSV, HSI. Cuando la imagen se presenta en mapa de bits (bitmap) usando el modelo RBG, cada elemento de la matriz es un valor de 3 bytes, una para cada color (rojo, verde, azul), cuya combinación es el color del pixel. Esta característica se le menciona como planos de imágenes, cuando se trabaja con imágenes en escala de grises se usa solo un plano, a diferencia de cuando se trabaja una imagen a color en el cual se utilizan los tres planos, sin importar el modelo de color que se use porque siempre se utilizan tres parámetros.

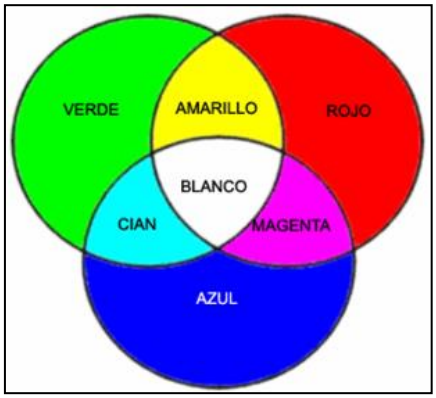
2.2.2. Representación vectorial de los colores

En los diferentes modelos de color, estos se representan como combinaciones lineales de tres factores: en el modelo RGB son la combinación del rojo, verde y azul; en el CMY son el cían, magenta y amarillo; en el HSL son el matiz, saturación y luminiscencia.

Para facilitar el análisis matemático de los colores se representan como vectores, por ejemplo, en el modelo RGB; la base vectorial del espacio de color la constituyen los colores, rojo, verde y azul; resultando que cualquier color puede ser expresado como una combinación lineal de la base:

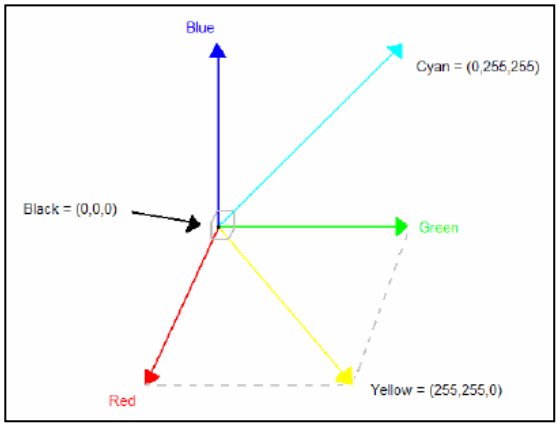
Color = a*Rojo + b*Verde + c*Azul, Color = (a,b,c)

Figura 37. **Combinación de colores**



Fuente: OTINIANO, Carlos. *Detección de material articulado en suspensión en ambientes de baja iluminación, mediante el procesamiento digital de imágenes*. https://decibel.ni.com/content/docs/DOC-31879. Consulta: 7 de noviembre de 2016.

Figura 38. **Representación vectorial de los colores**



Fuente: OTINIANO, Carlos. *Detección de material articulado en suspensión en ambientes de baja iluminación, mediante el procesamiento digital de imágenes*. https://decibel.ni.com/content/docs/DOC-31879. Consulta: 7 de noviembre de 2016.

Donde a, b, y c son valores enteros que pertenecen al intervalo (0, 255), entonces un tono amarillo se puede expresar como la siguiente ecuación Amarillo = 255*Rojo + 255*Verde + 0*Azul, Amarillo (255,255,0).

Representar a los colores en forma vectorial hace posible utilizar conceptos matemáticos como la norma, el producto punto, la proyección vectorial, distancia, rotación y transformaciones lineales. Estos conceptos permiten implementar técnicas de procesamiento.

2.2.3. Procesamiento básico

La representación de las imágenes a través de matrices de pixeles en función de coordenadas "x" y "y" perteneces al dómino espectral bidimensional, permitiendo realizar transformaciones lineales a las imágenes, simplemente son matrices que se pueden sumar, restar o multiplicar.

Cada color puede diferenciarse entre sí, ya que cada color puede expresarse como un vector y obteniendo la distancia vectorial entre dos vectores da como resultado la diferencia de colores, por ejemplo: sean dos colores D1 = (r1,v1,a1) y D2 = (r2,v2,a2); la distancia se define como:

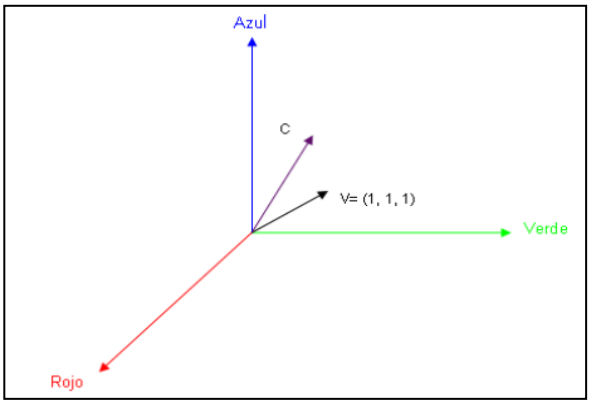
$$D(D1, D2) = \sqrt{(r1 - r2)^2 + (v1 - v2)^2 + (a1 - a2)^2}$$

La comparación de color es un proceso que requiere que se comparen pixel a pixel de la imagen, la comparación de colores comúnmente se utiliza para la identificación de colores, por ejemplo, mover bloques de determinado color. Se puede utilizar el algoritmo siguiente para buscar un color en una imagen:

- Definir el color de interés.
- Establecer el umbral K de la distancia entre los colores del mapa de bits y el color de interés.
- Para cada pixel de la imagen se obtienen las distancias y si la distancia es menor al umbral K de comparación entonces marca en blanco, de lo contrario se marca en negro.

En el modelo RGB, la escala de grises se caracteriza por tener el mismo valor para rojo, verde y azul. En el espacio vectorial dichos colores se encuentran en el vector (1,1,1). Para convertir un color a escala de grises, se debe determinar la proyección de cada componente sobre el valor vector V y así determinar el nivel de gris para ese color.

Figura 39. **Representación vectorial de colores y vector gris**



Fuente: LENIN GORDILLO, Jorge Yánes. *Aplicación de visión con Labview para la detección de frascos con turbiedades*. p. 65.

2.3. Técnicas de procesamiento digital de imágenes

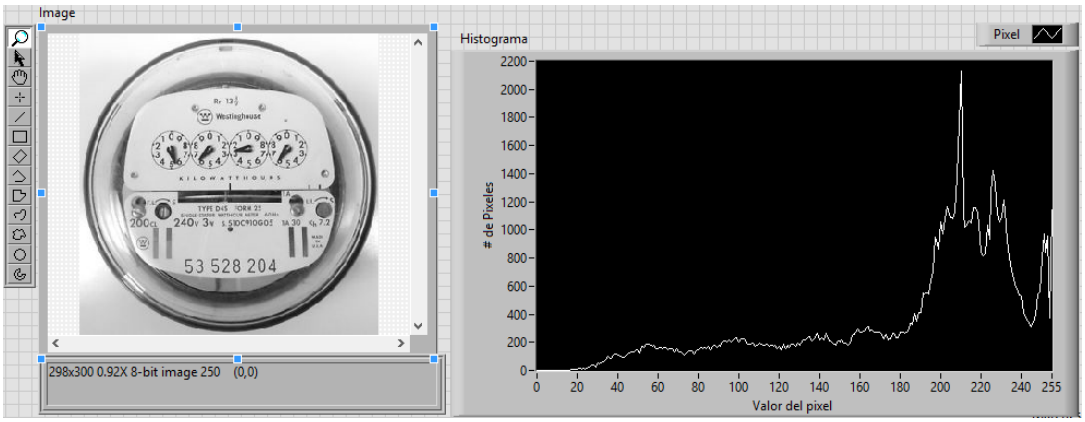
El campo que se encarga del procesado de las imágenes digitales por medio de una computadora digital es el procesamiento digital de imágenes. la mayor parte de las técnicas DIP actúan tratando a la imagen como una señal de dos dimensiones (2D) y después aplicando técnicas estándar de procesado de señales de una dimensión (1D). Dentro de las operaciones más comunes de procesamiento se encuentran transformaciones geométricas (reducción, rotación, alargamiento), correcciones de color (ajustes de brillo y contraste), alineación de imágenes, segmentación, interpolación, reconocimiento de patrones o características en una imagen, entre otros.

El desarrollo de la tecnología en el campo de la adquisición de imágenes y sistemas de cómputo más eficientes, han logrado que las aplicaciones de DIP se extiendan a muy diversas áreas como las aplicaciones médicas, reconocimiento de objetos, visión por computadora, manipulación de fotografías, aplicaciones militares, entre otros.

2.3.1. Histogramas

El histograma de una imagen consiste en una gráfica donde se muestra el número de pixeles n_k de cada nivel de gris r_k que aparecen en la imagen, en el eje de las "X" están todos los niveles de grises; es 0 negro y 255 blanco y en el eje de las "Y" la cantidad de pixeles que corresponden cada tonalidad.

Figura 40. **Histograma de una imagen**



Fuente: elaboración propia.

El análisis estadístico del histograma sirve para comparar contrastes e intensidades entre imágenes; el histograma puede ser alterado para producir cambios en la imagen, a partir del histograma se puede binarizar una imagen que consiste en convertir una imagen en blanco y negro. Se escoge un umbral y a partir de este valor todo lo que es menor es negro y todo lo que es mayor es blanco, con esto se logran resaltar las características esenciales en una imagen.

2.3.2. Filtros de convolución

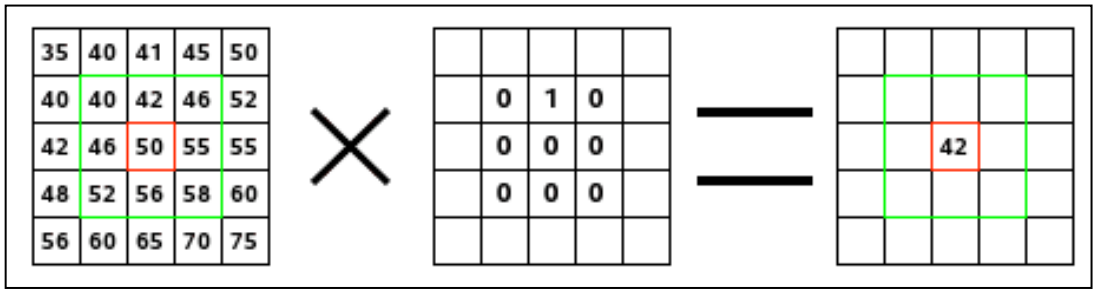
Los filtros de convolución son muy utilizados en el mejoramiento de los detalles de las imágenes con el propósito de resaltar características especiales, atenuar o resaltar bordes para detectarlos.

La mayoría de los filtros usan una matriz de convolución; esta puede ser personalizada para conseguir el resultado deseado, comúnmente se le llama

kernel a la matriz de convolución y puede ser de cualquier dimensión MxN, aunque las más comunes son de 5x5, 3x3 y 7x7.

Los kernel de convolución son utilizados para representar algoritmos que modifiquen el valor de un pixel basado en el valor del mismo pixel y el de sus vecinos. Las matrices de convolución se deslizan sobre la imagen y el centro del kernel de convolución se coloca sobre el pixel a ser analizado, se obtiene el resultado y se coloca en la matriz de la imagen de salida.

Figura 41. **Efecto del kernel en una matriz**

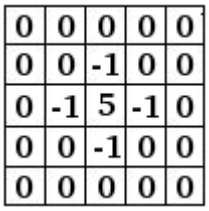


Fuente: *8.2. Matriz de convolución.* https://docs.gimp.org/es/plug-in-convmatrix.html. Consulta: 7 de noviembre de 2016.

A la izquierda la matriz original, en el centro el kernel y a la derecha el resultado de convolución; el pixel inicial tiene un borde rojo y el área de acción del kernel tiene un borde verde.

Se pueden encontrar kernel completamente perfeccionado para obtener un resultado; a continuación, se muestran los más comunes

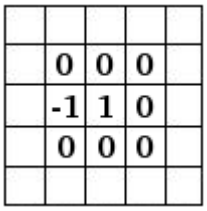
Figura 42. **Kernel para enfocar**



0	0	0	0	0
0	0	-1	0	0
0	-1	5	-1	0
0	0	-1	0	0
0	0	0	0	0

Fuente: *8.2. Matriz de convolución.* https://docs.gimp.org/es/plug-in-convmatrix.html. Consulta: 7 de noviembre de 2016.

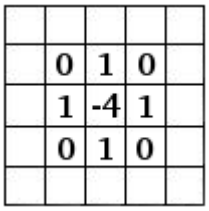
Figura 43. **Kernel para realizar bordes**



	0	0	0	
	-1	1	0	
	0	0	0	

Fuente: *8.2. Matriz de convolución.* https://docs.gimp.org/es/plug-in-convmatrix.html. Consulta: 7 de noviembre de 2016.

Figura 44. **Kernel para detectar bordes**



	0	1	0	
	1	-4	1	
	0	1	0	

Fuente: *8.2. Matriz de convolución.* https://docs.gimp.org/es/plug-in-convmatrix.html. Consulta: 7 de noviembre de 2016.

2.3.3. Interpolación bicúbica

Es el proceso de estimar valores intermedios en un evento continuo a partir de muestras discretas. Este método es ampliamente utilizado en DIP para ampliar o reducir imágenes y también para corregir distorsiones geométricas.

El algoritmo de interpolación bicúbica es uno de los más utilizados aun hoy en día, debido a su bajo costo computacional y los resultados que se obtienen en comparación de la interpolación bilineal y la interpolación del vecino más próximo. Este método estima un pixel analizando la vecindad de 4 x 4 alrededor de este mediante un algoritmo de polinomios cúbicos definidos en sub intervalos.

3. *SOFTWARE* DE DESARROLLO LABVIEW

3.1. Introducción a Labview

Labview es un *software* de desarrollo con enfoque para aplicaciones de ingeniería y ciencia, tiene participación en una amplia variedad de industrias. La sintaxis de programación es gráfica, en forma de bloques, fácil de entender e intuitiva, por lo cual facilita visualizar, crear y codificar sistemas de ingeniería que permitan ser más eficientes en procesos que ya están establecidos o rediseñarlos de tal forma que permite ser más competitivos a nivel mundial.

Debido a su gran capacidad de integración con otros *software* y de integrar cualquier dispositivo, Labview es preferido en distribuidoras de energía eléctrica, industrias del plástico, metal, sector alimenticio, azucareras, medicina, procesos de manufactura de componentes electrónicos, entre otros.

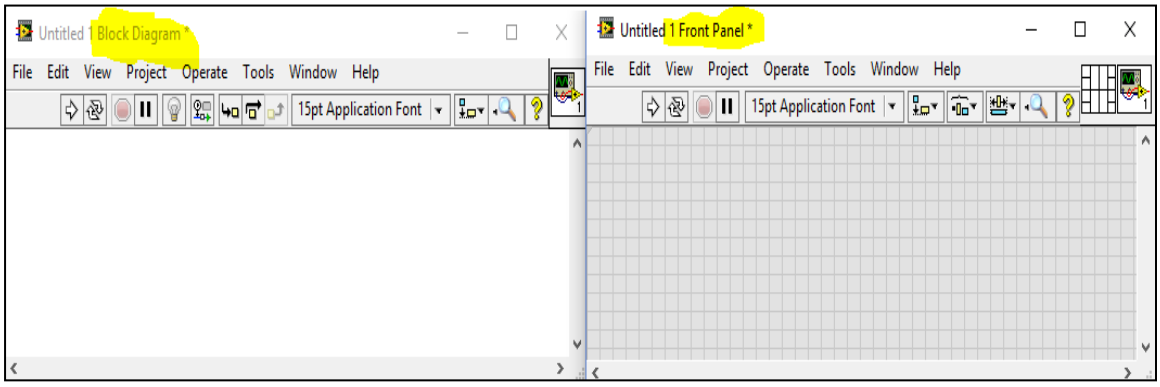
Hoy en día Labview tiene una muy amplia participación en el sector eléctrico ayudando desde la generación, transmisión y distribución de energía eléctrica; Labview combinado con *hardware* modular y que además sea reconfigurable ayuda a resolver la creciente complejidad involucrada en proporcionar sistemas de medida y control en tiempo real.

3.1.1. Entorno de desarrollo

Labview es una herramienta de programación gráfica, comúnmente llamada programación en bloques o G; es utilizado en los sistemas de instrumentación, lo que se conoce instrumentación virtual, los programas

creados en Labview son guardados en ficheros llamados VI y con la misma extensión, por ejemplo: prueba.vi. Tienen dos ventanas principales: diagrama de bloques y panel frontal.

Figura 45. **Diagrama frontal y panel frontal**



Fuente: elaboración propia, utilizando *software* Labview.

- Diagrama de bloques: donde se realiza la programación
- Panel Frontal: es la parte que visualiza el usuario

El panel frontal y diagrama de bloques están conectados a través de las terminales, estas terminales sirven como entradas o salidas de datos.

En la parte superior de cada una de estas ventanas se pueden encontrar las barras de herramientas, en el diagrama de bloques se encuentran más opciones que en la de panel frontal.

Figura 46. **Barra de herramientas diagrama de bloques**

Fuente: elaboración propia, utilizando *software* Labview.

Las primeras cuatro opciones sirven para controlar la ejecución del programa, el primer botón indica si hay errores en el programa con una flecha rota y cuando no hay errores la flecha está completa y se ejecuta una sola vez el programa. El segundo ejecuta de forma continua el programa, se debe tener mucho cuidado con este botón ya que puede crear un bucle en el programa. El cuarto detiene por completo el programa y el último permite realizar una pausa en la ejecución.

Figura 47. **Flecha rota cuando hay error**

Fuente: elaboración propia, utilizando *software* Labview.

El segundo grupo de herramientas es para la depuración del programa; el primer botón permite una ejecución lenta ya que permite observar como los datos llevan la información a otros instrumentos del programa, el segundo botón es una punta de prueba que permite conocer el valor del dato que pasa por ese punto donde fue colocada la punta de prueba, los otros tres restantes permite la ejecución del programa paso a paso.

Figura 48. **Herramientas de depuración**

Fuente: elaboración propia, utilizando *software* Labview.

En la figura 49 se observa el menú desplegable que permite dar el formato a los textos, en la figura 50 las herramientas que ayudan a ordenar, alinear y controlar el tamaño de los objetos en el diagrama de bloques o panel frontal.

Figura 49. **Menú desplegable para formatos de texto**

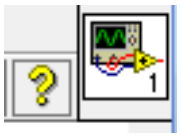
Fuente: elaboración propia, utilizando *software* Labview.

Figura 50. **Herramientas para ordenar objetos**

Fuente: elaboración propia, utilizando *software* Labview.

En la parte lateral derecha del panel frontal o diagrama de bloques se pueden encontrar la ayuda que es el icono izquierdo de la figura 51 y del lado derecho de las propiedades del VI que pueden ser modificadas haciendo un click derecho sobre este icono.

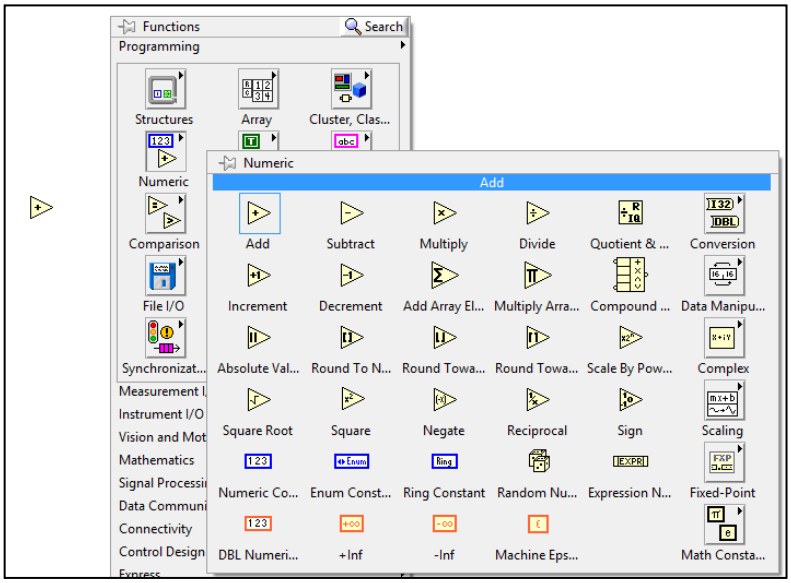
Figura 51. **Ayuda y propiedades del VI**



Fuente: elaboración propia, utilizando *software* Labview.

Para acceder a las paletas de controles es necesario realizar un click derecho en cualquier parte del diagrama de bloques o en el panel frontal, aparecerán los diferentes controles y funciones, basta con arrastrar un control o función desde la paleta de controles hacia el diagrama de bloques o panel frontal.

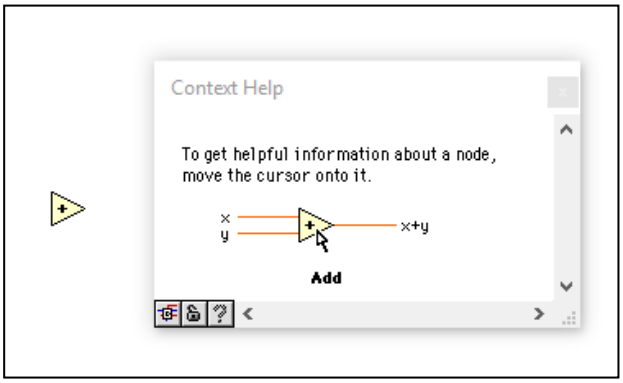
Figura 52. **Paleta de controles**



Fuente: elaboración propia, utilizando *software* Labview.

La ayuda contextual puede abrirse en el Help > ShowContexHelp, esta ventana muestra la información del objeto así como el tipo de entrada y salida que acepta.

Figura 53. **Context Help**



Fuente: elaboración propia, utilizando *software* Labview.

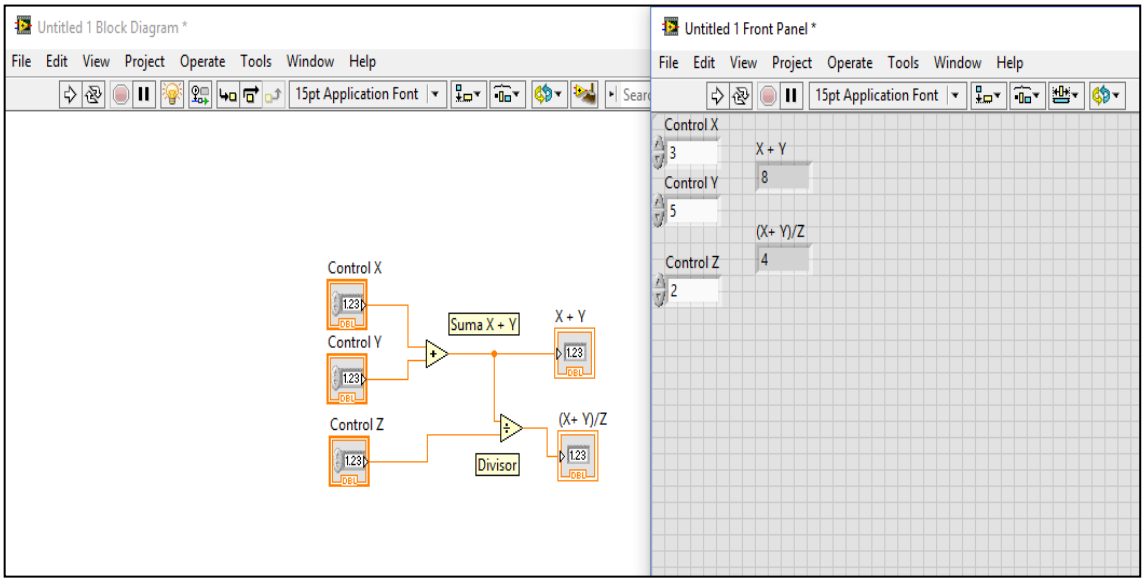
3.1.2. Flujo de ejecución

Como lo se ha indicado anteriormente, el lenguaje de programación que utiliza Labview también es llamado lenguaje G; la ejecución del código fuente está basada en el flujo de datos.

Se llamará programa en Labview a las funciones que están unidas mediante cables en los cuales los datos circulan o fluyen por los cables y una función solo puede ejecutarse si esta tiene todos los datos de entrada para procesarla; esta es la mejor forma y la más apropiada para sistemas multiprocesador y multihilo.

En la figura 54 se muestra un programa en Labview que consiste en dos operaciones matemáticas básicas: la primera operación es la suma de dos número "X" y "Y" y el resultado de esta operación es divida por Z.

Figura 54. **Programa en Labview suma y división**

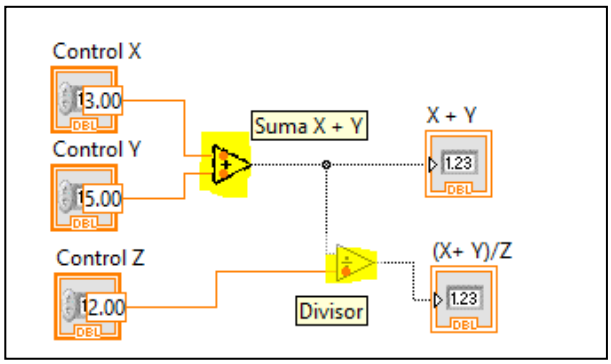


.

Fuente: elaboración propia, utilizando *software* Labview.

Los datos de entrada se ven en el panel frontal y las operaciones en el diagrama de bloques figura 54; la ejecución inicia llevando los datos de los controles "X", "Y" y "Z" a las entradas de la operación suma y divisor como se muestra en la figura 55.

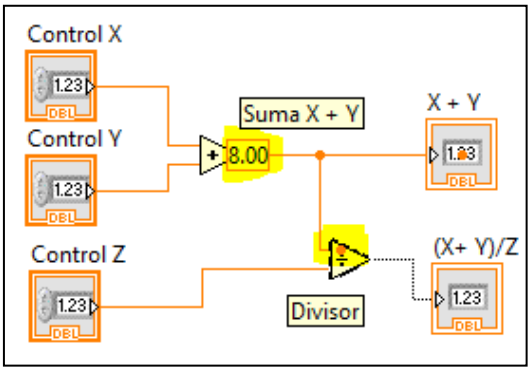
Figura 55. **Llegada de los datos a las entradas**



Fuente: elaboración propia, utilizando *software* Labview.

El punto naranja en los operadores indica la llegada de los datos, como se puede observar el operador suma tiene completa sus entradas por lo cual se ejecuta la suma figura 56.

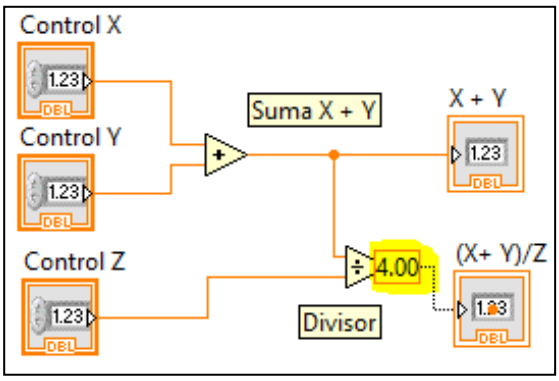
Figura 56. **Ejecución de la suma**



Fuente: elaboración propia, utilizando *software* Labview.

Ahora las dos entradas del operador división están completas lo cual permite la ejecución de la división figura 57.

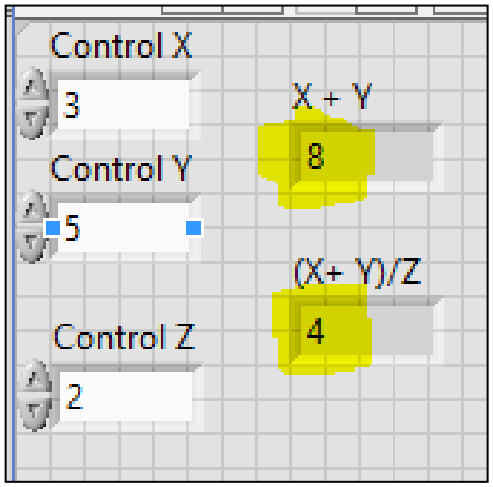
Figura 57. **Ejecución de la división**



Fuente: elaboración propia, utilizando *software* Labview.

Por último, el resultado de la suma y división se muestran en los indicadores del panel frontal figura 58.

Figura 58. **Resultado de las operaciones**



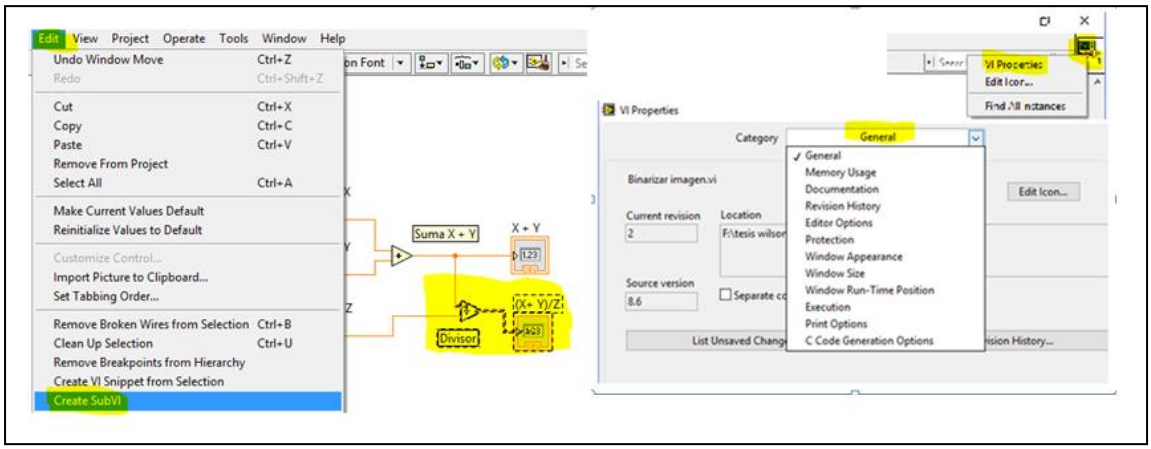
Fuente: elaboración propia, utilizando *software* Labview.

3.1.3. VI (instrumentos virtuales)

Los programas creados en Labview se guardan en ficheros llamados VIs del acrónimo *virtual instrument*, cuando el programa sea de un tamaño muy grande será necesario separarlo en varios ficheros o en otras ocasiones parte del programa no ser utilizado con frecuencia y convendrá tenerlo por aparte, un VI puede contener a otro VI, este segundo será llamado SubVI.

Se va a mostrar una de tantas maneras para crear un SubVI; es necesario marcar la parte del programa que se desea convertir en SubVI y luego ir a Editar > Crear SubVI.

Figura 59. **Creación de SubVI**



Fuente: elaboración propia, utilizando *software* Labview.

Muchas veces se hace necesario cambiar las propiedades del VI, esto se puede hacer dando click derecho en el icono derecho como se muestra en la de la figura 59; las propiedades que puede configurarse son las siguientes:

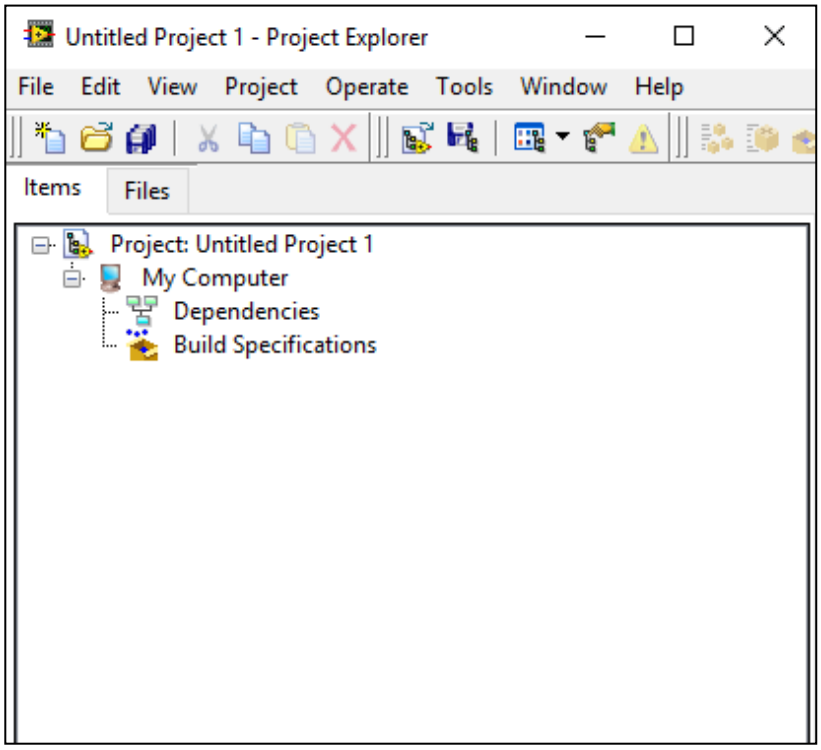
- General: información de la versión y cambios sin guardar.
- *Memory usage*: espacio que ocupa las diferentes partes del VI.
- Documentation: información sobre el VI.
- *Revision history*: configuración e información sobre el historial del VI.
- *Editor option*: parámetros que afectan al crear el VI.
- *Security*: permite bloquear y proteger con contraseña el código del fichero.
- *Windows appearance*: configuración de la ventana del panel frontal que se mostrará al usuario.
- *Windows size*: tamaño de la ventana del programa cuando se ejecute.
- *Windows run-time position*: posición de la venta del programa cuando se ejecute.
- *Execution*: afecta la forma de ejecutarse de un VI.
- *Print option*: configura la forma en que se imprimirá el VI.

3.1.4. Proyectos

Antes de la versión 8.x de Labview la organización de proyectos medianos y grandes era muy complicado y toda la responsabilidad de la organización era del programador el cual debería de guardar los ficheros en directorios en el disco.

A partir de la versión 8.0, Labview introdujo la opción de agrupar todos los ficheros en un proyecto, el proyecto consiste en un fichero en formato XML y con extensión .LVPROJ; los ficheros que componen un proyecto pueden ser VIs, controles, documentos y cualquier otro fichero; a continuación, se muestra el aspecto de un proyecto.

Figura 60. **Proyecto en Labview**



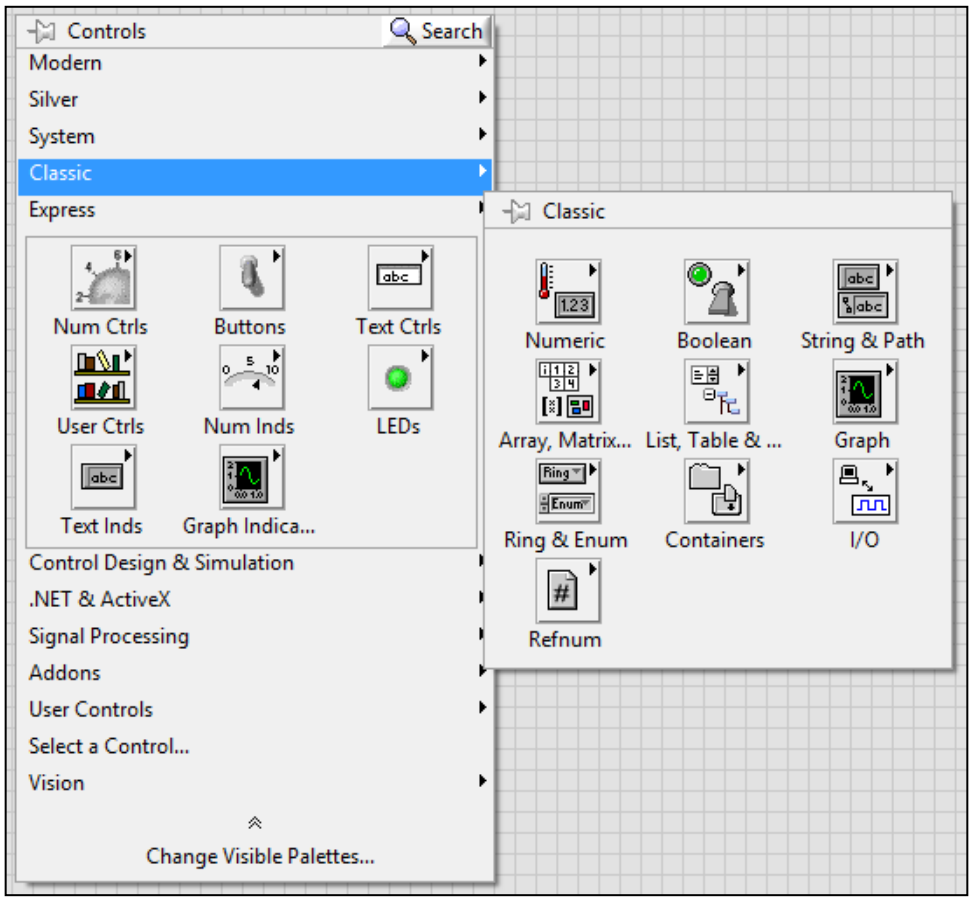
Fuente: elaboración propia, utilizando *software* Labview.

3.2. Menú de instrumentos virtuales de Labview

Los controles e indicadores normalmente se les llama controles a ambos; estos pueden accederse haciendo click derecho sobre el fondo del panel frontal o del diagrama de bloques.

Los controles son las entradas de datos y los indicadores, las salidas de datos; están clasificados según su estilo en varios submenús: Modern, Systems y Classic; dentro de cada submenú hay otros menús que clasifican por el tipo de dato.

Figura 61. **Submenús en el panel frontal**

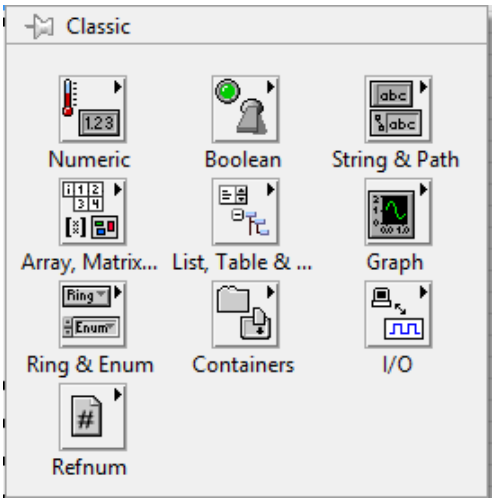


Fuente: elaboración propia, utilizando *software* Labview.

3.2.1. Menú de controles, indicadores y funciones

Si se quiere utilizar un control, hay que seleccionar el terminal y llevarlo al lugar deseado en el panel frontal; hay indicadores numéricos, booleanos, cadenas, arreglos, matrices, gráficas, entre otros. Hay que tener cuidado al seleccionar el tipo de control para una entrada, ya que el control solo puede ser del mismo tipo de entrada, por ejemplo, control booleano con entrada booleana.

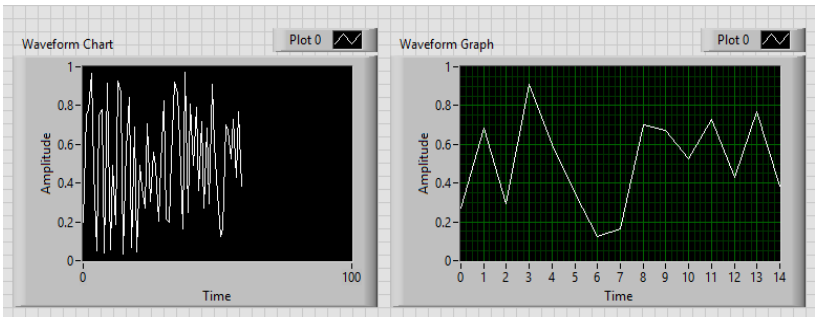
Figura 62. **Menús de submenú Classic**



Fuente: elaboración propia, utilizando *software* Labview.

Para los indicadores tipo gráficas hay varios tipos: los más importantes son los *waveform chart* y *waveform graph*. La gran diferencia entre los indicadores *waveform chart* y *waveform graph*, es que estos últimos no tienen memoria y dibujan totalmente la gráfica cuando llegan los nuevos datos y los waveform chart dibujan los datos existentes junto con los nuevos datos.

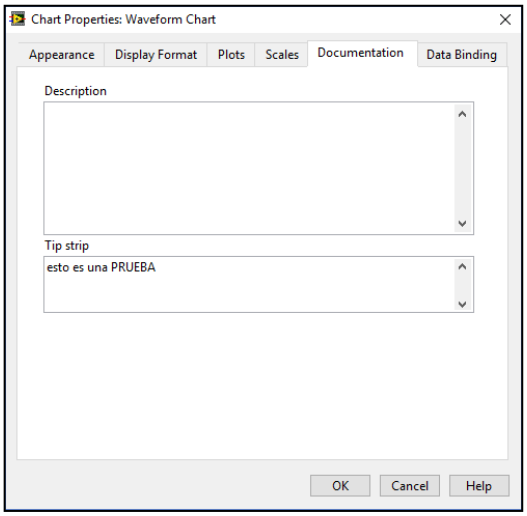
Figura 63. ***Waveform chart* y *waveform graph***



Fuente: elaboración propia, utilizando *software* Labview.

Una buena práctica es añadir a todos los controles e indicadores una descripción de lo que realizan; esto se puede hacer ingresando a las propiedades del control y luego en la pestaña documentación y esta pestaña tiene dos opciones: documentación y *tip strip*. En *tip strip* se puede escribir el texto para indicar lo que realiza.

Figura 64. **Pestaña documentación y *tip strip***



Fuente: elaboración propia, utilizando *software* Labview.

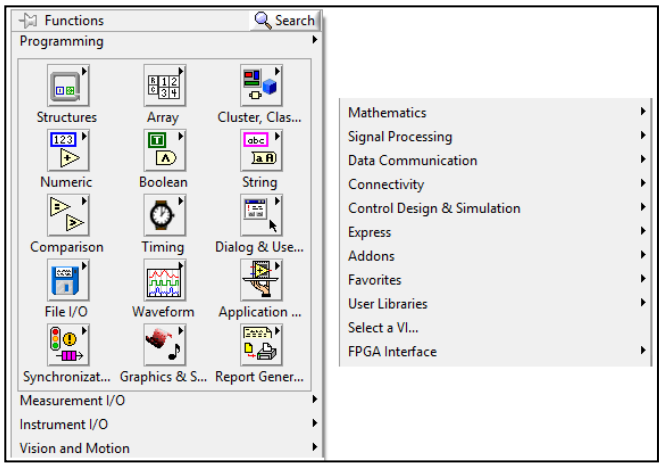
Las funciones son las que se muestran al trabajar sobre el diagrama de bloques, se puede acceder a diferentes funciones sub VI y estructuras disponibles.

La paleta de funciones también tiene submenús que se dividen dependiendo de la aplicación; se pueden encontrar submenús de: programación, instrumentación, modbus, conectividad, dnp3, *vision and motion*, entre otros.

Las funciones más utilizadas son las del submenú, programación donde se puede encontrar funciones para: comparación, manejo de archivos, bucles, además de muchas otras funciones.

Existen muchas funciones más que puede utilizarse, pero debido a que son funciones específicas deben ser adquiridas como una licencia para utilizarse tal es el caso de DNP3, *Vision an Motion*, *Data Base*, entre otros.

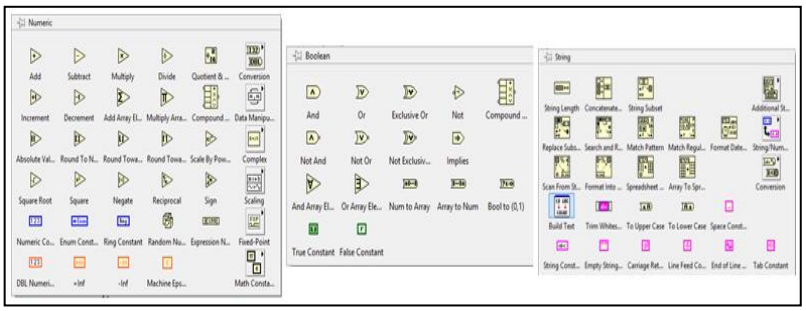
Figura 65. **Submenús de funciones de diagrama de bloques**



Fuente: elaboración propia, utilizando *software* Labview.

Los datos pueden ser de tres tipos: booleanos, numéricos y cadenas. En la figura 66 se ven las funciones para trabajar con cada uno de estos tipos de datos.

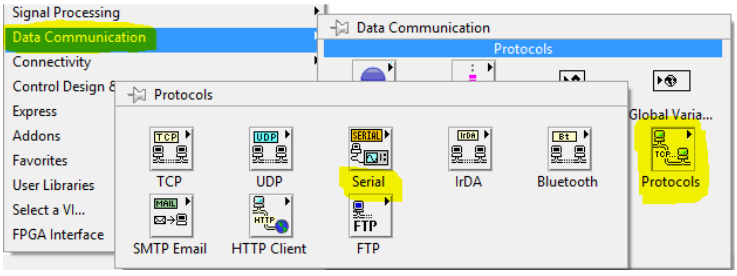
Figura 66. **Funciones numéricas, booleanos y cadenas**



Fuente: elaboración propia, utilizando *software* Labview.

Labview tiene muchas funciones y una es la comunicación con otros dispositivos; esta puede ser ethernet, serial, *bluetooth*, entre otros. Maneja muchos protocolos de comunicación. El submenú se llama *data comunication* el cual contiene todos los recursos para comunicarse con estos dispositivos.

Figura 67. **Funciones para comunicación**



Fuente: elaboración propia, utilizando *software* Labview.

3.2.2. Creación de programas

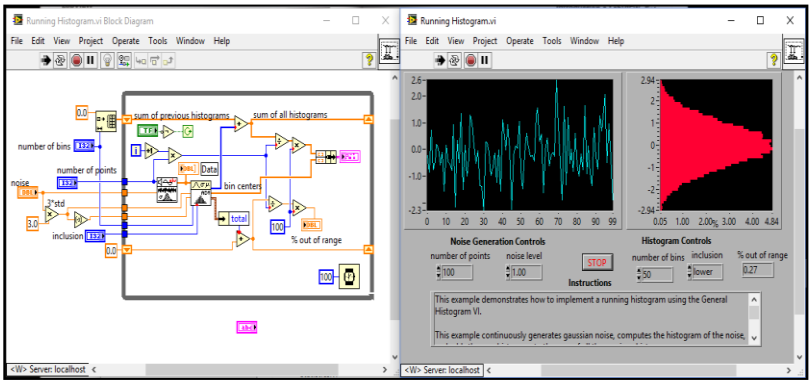
En Labview, la programación normalmente se realiza en el diagrama de bloques y está conformado por:

- Controles: entrada de datos.
- Funciones, sub VIs y estructuras: ayudan a realizar operaciones con los datos.
- Indicadores: salidas de datos.

Los datos circulan por el programa mediante los cables que unen controles, funciones e indicadores. Para realizar la conexión se utiliza la herramienta *connect wire* de la paleta de herramientas, los cables tienen una única fuente pero pueden tener varios destinos que pueden ser indicadores o funciones y se aconseja que los cables sean lo más corto posible para mantener claridad en el programa.

En el panel frontal se introducen los datos iniciales del programa una vez se finalice el programa, para ejecutarla se presiona el botón Run y cuando el programa termine de ejecutarse mostrará los resultados en los indicadores que correspondan.

Figura 68. **Programa en Labview**



Fuente: elaboración propia, utilizando *software* Labview.

3.2.3. Depuración de código

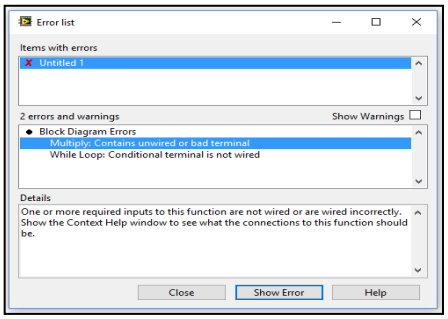
Cuando se crean programas, muchas veces se cometen errores; Labview tiene muchas herramientas para la depuración de errores que ayudan al programador para corregir los errores del programa.

Como se indicó anteriormente, cuando hay un error en el programa el botón de run de la barra de herramientas aparecerá como una flecha rota, al realizar click en la fecha rota aparecerá una ventana llamada lista de errores la cual consta de tres partes:

- Ítem con errores: lista todos ítems afectados por el error.
- Errores y precauciones: lista todos los errores y precauciones asociadas al VI, si el VI es seleccionado.
- Detalle: indica en detalle el error seleccionado en errores y precauciones.

En la misma barra de herramientas se puede encontrar otro icono en forma de bombilla; si se hace click en este icono, la ejecución será lenta que permita ver el flujo de datos en todo el programa del diagrama de bloques.

Figura 69. **Lista de errores**



Fuente: elaboración propia, utilizando *software* Labview.

Otra herramienta que ayuda en la depuración de errores son los *probes*, que pueden ser activados sobre un control, cable o indicador; cuando son activados, aparecerá una ventana flotante que indica los valores en tiempo real.

Para conocer la explicación del error existe una herramienta que amplía la información del error, la cual se puede acceder en Help - Explain error.

3.2.4. Herramientas adicionales

Se puede acceder a otras configuraciones para modificar parámetros del programa, se puede acceder a través de *Tools - options*. A continuación, la descripción de las opciones más importantes:

- Paths: la carpeta de de instalación de Labview.
- Front panel: opciones relacionadas con el panel frontal y controles.
- Diagrama de bloques: manejo automático de errores y otras ayudas al programador.
- Controles y funciones: forma de presentar las funciones y controles.
- Seguridad: permite restringir el acceso a Labview.
- Variables compartidas: servidores que pueden manejar variables compartidas.

3.3. Módulo de *Vision and Motion*

El módulo de *Vision and Motion* de Labview es una herramienta poderosa y hoy en día se encuentran muchas aplicaciones de visión artificial en la industria: fábricas de semiconductores, embotelladoras, industria eléctrica, fábricas de productos alimenticios, entre otros Labview es preferido debido a sus muchas funciones en el procesamiento digital de imágenes y lo rápido que

pueden crearse programas enfocados a resolver problemas que impiden que las fábricas mejoren en sus procesos y la vez impacte en sus producciones.

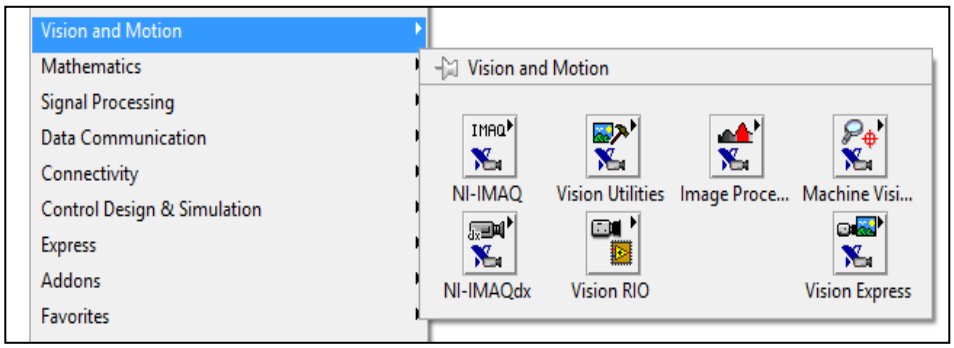
Para el procesamiento digital se realiza directamente en el diagrama de bloques utilizando las funciones del módulo *Vision and Motion* que contiene técnicas de procesamiento digital más comunes, adquisición de imágenes, *vision express*, entre otros. El resultado o salida del programa puede visualizarse en el panel frontal y de esta forma conocer si se obtiene el resultado deseado.

3.3.1. Funciones de *Vision and Motion*

La paleta de funciones de *Vision and Motion* tiene varios módulos de funciones:

- NI-IMAQ: para la aquisición y captura de imágenes.
- *Vision utilities*: crear y manipular imágenes en NI *vision*.
- *Image processing*: analizar, filtrar y aplicar procesamiento digital de imágenes.
- *Machine vision*: máquinas con visión artificial para realizar tareas de búsqueda de patrones y objetos en una imagen y medición de dimensiones de partes si se encuentra según la especificación.
- NI-IMAQdx: para adquisición y captura de imágenes por medio de un dispositivo USB.
- Vision RIO: aplicaciones para control de líneas de entrada y salida en un NI IMAQ I/O device.
- *Vision express*: para desarrollos rápidos de adquisición y procesamiento.

Figura 70. **Módulos de *Vision and Motion***

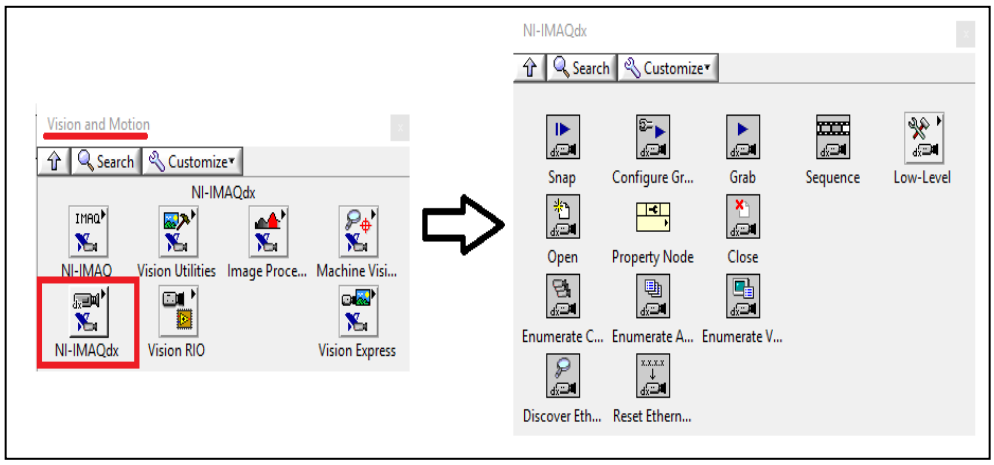


Fuente: elaboración propia, utilizando *software* Labview.

3.3.2. Adquisición y captura de imágenes

Para la captura de imágenes por medio de una cámara USB es necesario contar con las funciones necesarias que lo permitan; la librería que permite la captura desde una *webcam* es llamado IMAQdx y está disponible entre las funciones del módulo vision and motion; este módulo de funciones se llama NI-IMAQdx. con este driver se pueden adquirir imágenes con cualquier cámara USB del mercado.

Figura 71. **Funciones de IMAQdx**



Fuente: elaboración propia, utilizando *software* Labview.

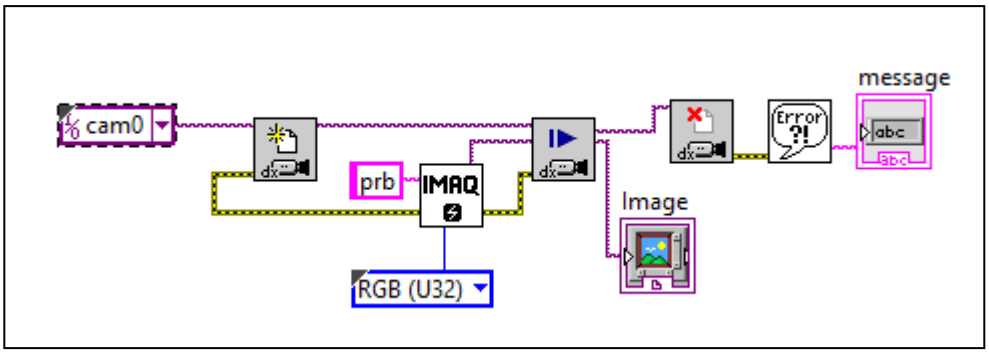
Como se ha mencionado anteriormente, la programación en Labview es fácil e intuitiva; crear un programa para capturar imágenes es completamente fácil pero también es necesario conocer algunas funciones de NI-IMAQdx y *vision utilities*.

- IMAQ *create*: crea un espacio de memoria temporal en la RAM para la imagen.
- IMAQ *Dispose*: destruye una imagen y libera el espacio ocupado en memoria.
- *Open*: abre una cámara y solicita las capacidades, carga en el archivo la configuración de una cámara y crea una única referencia para esta cámara.
- *Snap*: configura, inicia y adquiere una captura sin utilizar un botón.
- *Close*: detiene el proceso de captura y libera los recursos de la cámara.
- *Enumerate camera*: retorna un listado de las maras disponibles.

- Imagen: muestra la imagen capturada o procesada.

Los modos de captura pueden ser snap o continuo; el tipo Snap permite la adquisición de una sola captura de imagen y es almacenada en la memoria RAM, ideal para aplicacciones de baja velocidad y de un procesamiento lento.

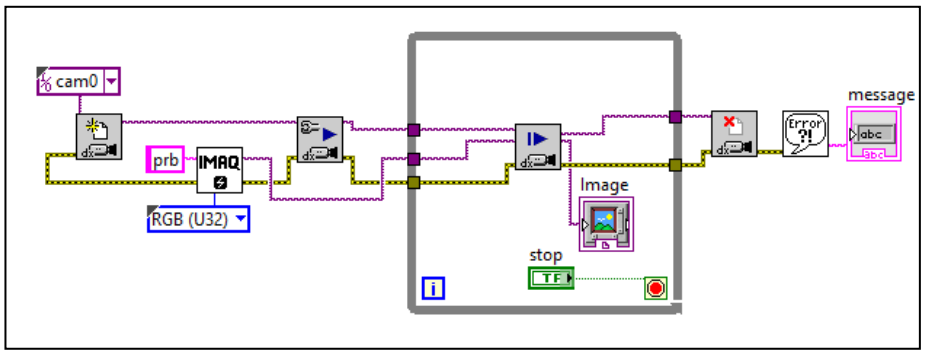
Figura 72. **Adquisición modo Snap**



Fuente: elaboración propia, utilizando *software* Labview.

Para el modo continuo es necesario la inserción de un lazo y la función IMAQdx *Configure grab* como se muestra en la figura 62. Este se utiliza para una alta adquisición de datos enviados a la memoria RAM

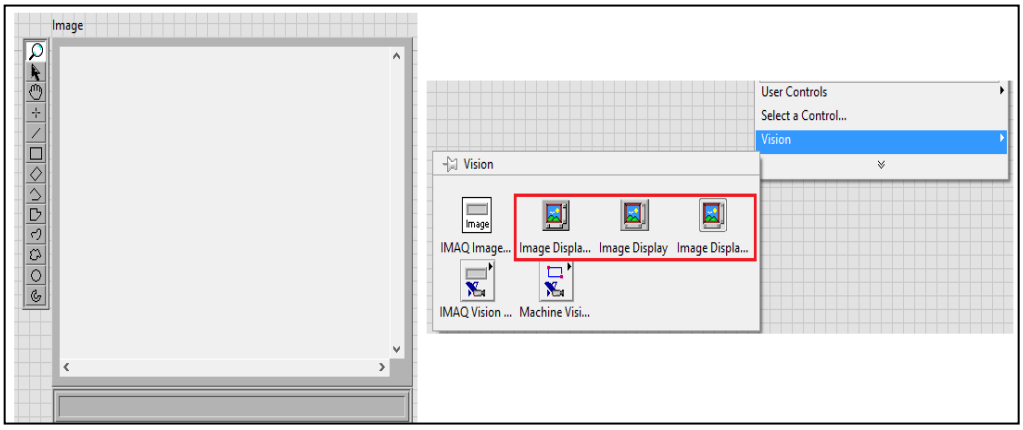
Figura 73. **Modo continuo o Grab**



Fuente: elaboración propia, utilizando *software* Labview.

El instrumento virtual utilizado en el panel frontal es imagen display que se puede encontrar en el módulo visión.

Figura 74. **Instrumento Virtual Imagen Display**

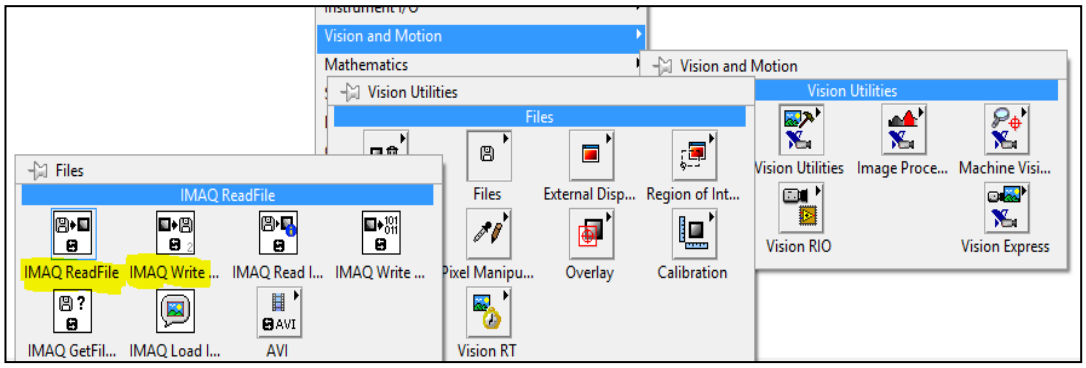


Fuente: elaboración propia, utilizando *software* Labview.

3.3.3. Lectura y escritura de imagen

Cuando se necesita importar una imagen al programa para el procesamiento; existe una función para realizar esta acción, la función se puede encontrar en vision utilities -> files -> IMAQ ReadFile. Los formatos de la imagen pueden ser los siguientes: JPEG2000, BMP, TIFF, PNG, y AIPD.

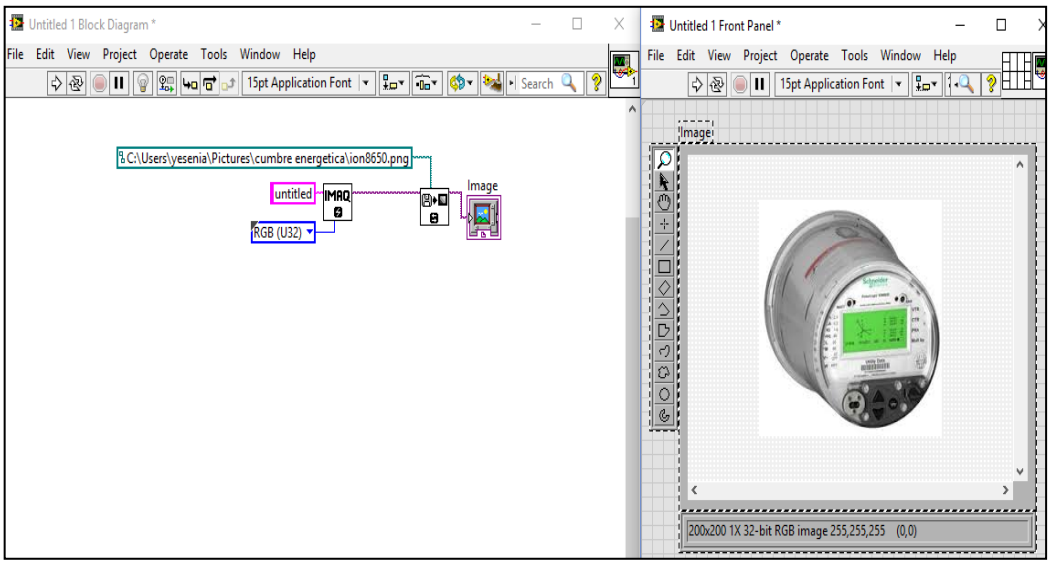
Figura 75. **Función IMAQ *ReadFile* y IMAQ *WriteFile***



Fuente: elaboración propia, utilizando *software* Labview.

Las entradas necesarias a completar son: la ubicación de la imagen a importar y el espacio de memoria que se debe reservar. También, se pueden guardar imágenes en los formatos que ya se han mencionado; para la escritura se utiliza la función IMAQ *WriteFile* y las entradas necesarias con la dirección donde se guardará la imagen; estas pueden servir como plantillas para otros programas donde se utilicen como referencia para realizar una búsqueda; las plantillas son importantes para la visión artificial.

Figura 76. **Programa para leer imágenes**

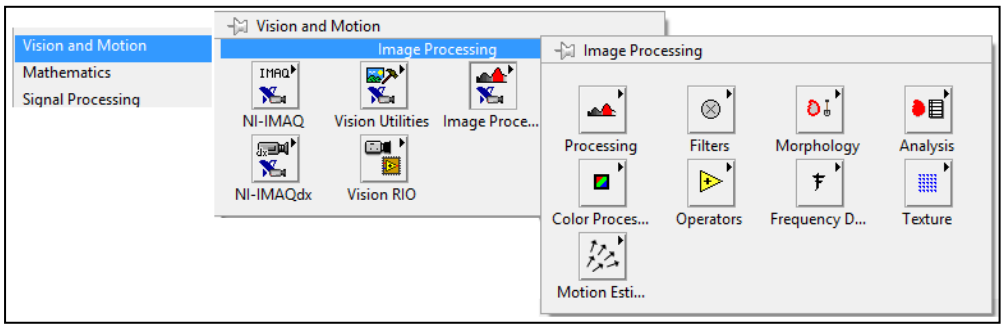


Fuente: elaboración propia, utilizando *software* Labview.

3.3.4. Procesamiento de imágenes

Estas funciones en el módulo de *vision and motion* se utilizan para analizar, filtrar y procesar imágenes. Se pretende enseñar algunas formas de procesamiento por medio de Labview y las que se utilizan para la extracción de datos característicos de medidores eléctricos de energía; las funciones se pueden encontrar en Image processing del módulo *Vision and Motion*.

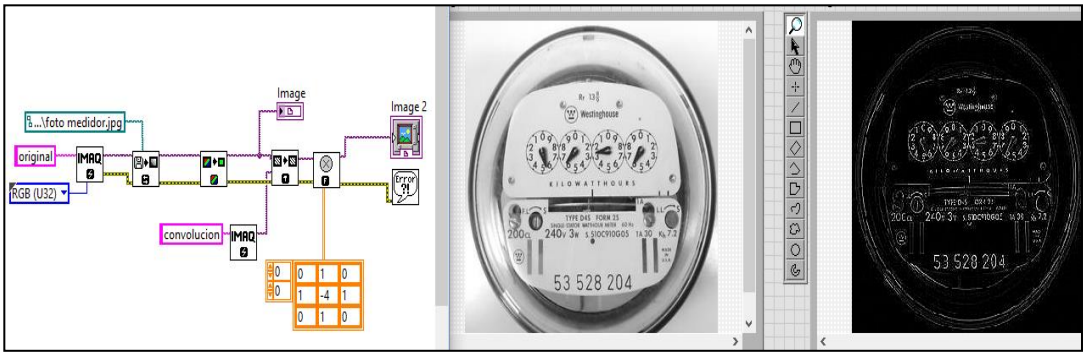
Figura 77. **Funciones Image Processing**



Fuente: elaboración propia, utilizando *software* Labview.

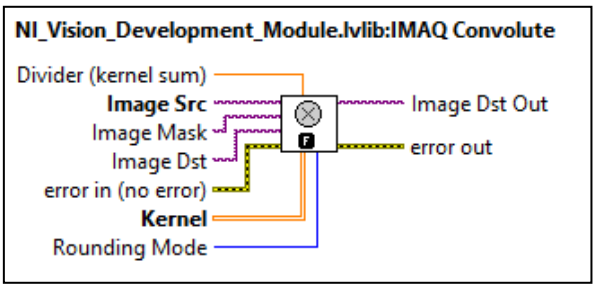
Los filtros son muy útiles para resaltar una característica en la imagen, la matriz de convolución o kernel se diseña según lo buscado en la imagen para nuestro caso es el realce de bordes, detectar bordes, entre otros. Se utilizará el kernel de 3X3 y la función que permite realizar la convolución se llama IMAQ, las entradas más importantes es el kernel, a la derecha de la figura 78 está el resultado utilizando el kerner que se muestra a la izquierda de la figura 78.

Figura 78. **Programa utilizando IMAQ *convolute* y *kernel***



Fuente: elaboración propia, utilizando *software* Labview.

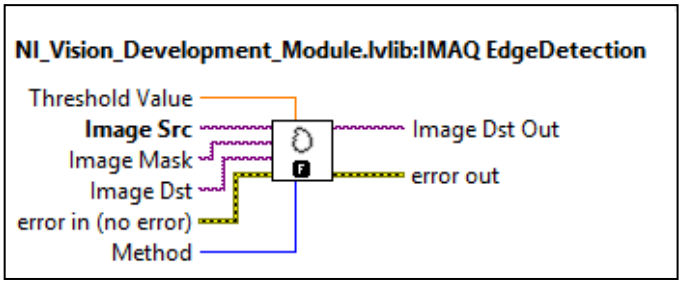
Figura 79. **Función IMAQ *convolute***



Fuente: elaboración propia, utilizando *software* Labview.

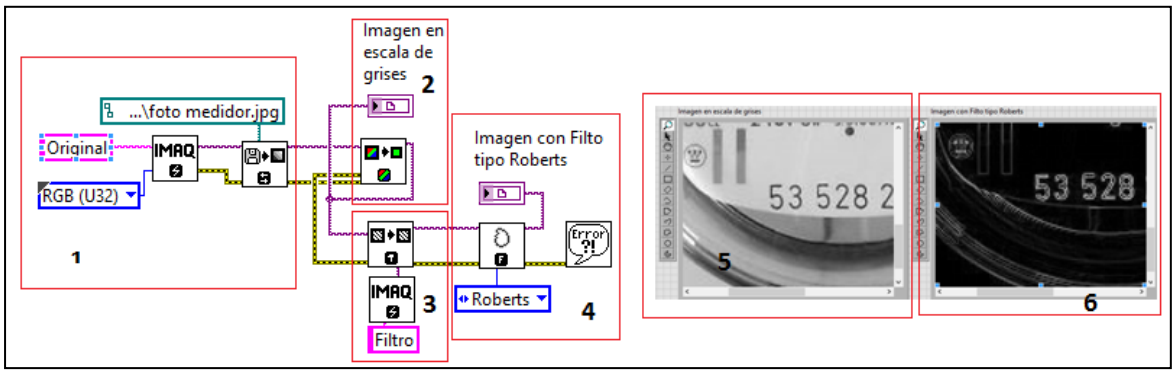
Utilizando filtros también se pueden detectar bordes que permite segmentar y reconocer cada uno de los objetos que están en la imagen; la función que permite detectar los bordes es IMAQ *EdgeDetection* y la entrada es una imagen en escala de grises y el operador puede ser tipo: *Differentiation*, Gradiente, Prewit, Roberts, Sigma, Sobel.

Figura 80. **IMAQ *EdgeDetection***



Fuente: elaboración propia, utilizando *software* Labview.

Figura 81. **VI para detectar bordes**



Fuente: elaboración propia, utilizando *software* Labview.

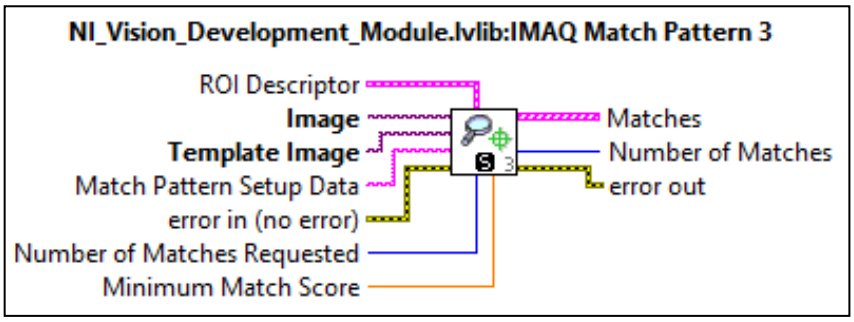
En la figura 81 se muestra un programa donde se aplica un filtro para detectar bordes; en este caso se utiliza la función IMAQ *edgeDetection* y se está utilizando el método Roberts, se ha divido el programa en 6 bloques; en el primer bloque, se crea un espacio de memoria para que se cargué la imagen previamente almacenada que tiene la característica RGB32; en el segundo bloque, la imagen se convierte a escala de grises; en el tercer bloque, se crea un espacio nuevo de memoria para cargar una copia de la imagen que ya está en escala de grises; en el cuarto bloque, se aplica la función para detección de bordes IMAQ *EdgeDetection* utilizando el método Roberts; el quinto bloque muestra la imagen en escala de grises y el sexto es la imagen con el filtro para detección de bordes.

Una de las herramientas muy utilizadas en el procesamiento de imágenes es la búsqueda de patrones; las funciones que permiten realizar la búsqueda basada en una plantilla se llama IMAQ *Match Pattern*, las entradas más importantes son:

- Imagen donde se busca
- Plantilla

La salida muestra los *matches* que despliegan la información como coordenadas de las coincidencias.

Figura 82. **IMAQ Match Pattern**



Fuente: elaboración propia, utilizando *software* Labview.

4. DISEÑO DEL AUTÓMATA PARA LA CAPTURA DE IMÁGENES

4.1. Justificación para el diseño del autómata

Hoy en día las empresas a nivel mundial están migrando de un proceso manual a un proceso automatizado con el objetivo de ser más eficientes y hacer un buen uso de sus activos. En muchas industrias, la labor humana puede ser reemplazada por máquinas autómatas que se encarguen de procesos peligrosos que ponen en riesgo la vida humana o que ayuden a mejorar los indicadores de producción siendo más competitivos a nivel mundial.

El uso de la visión artificial permite aprovechar de mejor forma el recurso humano y tecnológico; lo certero y eficaz de estos sistemas permite reemplazar el recurso humano y utilizar este recurso en actividades que aportan más valor a la empresa.

Actualmente, una persona se encarga de registrar el ingreso de los medidores de energía eléctrica; tomar la lectura de energía eléctrica en kWh y registrar los datos característicos de los medidores; es de mucho impacto el no registrar esta información a tiempo ya que se ven afectadas varios unidades de trabajo dentro de la empresa de electricidad.

Este proceso inicia cuando los medidores de energía eléctrica son retirados de los domicilios y son llevados a las bodegas de la *utility* para registrar el ingreso. Por día ingresan 300 medidores de energía eléctrica que son colocados en estructuras metálicas llamados *rack*; cada estructura metálica

tiene la capacidad de contener 28 medidores que pasan por el proceso antes mencionado; si cada estructura puede contener 28 medidores entonces se necesitan casi 11 estructuras para colocar los 300 medidores. Para tomar los datos de los 28 medidores se necesitan aproximadamente 45 minutos que es el equivalente a 0,75 hora; a continuación, el análisis del tiempo empleado en este proceso y el costo asociado a este.

Tabla V. **Costo horas-hombre**

Horas hombre	Costo horas - hombre	Cantidad de Rack	Total horas hombre	Total costo por día	Costo Mensual
0.75 horas	Q 19,27	11	8,25	Q 158,98	Q 4 769,32

Fuente: elaboración propia.

Si este proceso se automatiza y se emplea el recurso humano en otras actividades, se tendrá un ahorro aproximado de Q 4 500,00 más el valor de las otras actividades que realice la persona, con un ahorro cercano a los Q 7 000,00 mensuales, y decir, en un año el ahorro sería de Q 84 000,00.

4.2. Especificaciones de los dispositivos de hardware y *software* a utilizar

Es muy importante elegir los dispositivos y *software* adecuados que se emplearán en la construcción del autómata; se van a describir los más importantes y las características más relevantes:

- Motor stepper: ofrece alta torsión, precisión y fácil conectividad a drivers; es fácil diseñar un sistema de control de movimiento y precisión utilizando motores stepper pero hay que considerar las características

eléctricas y mecánicas; la tabla siguiente muestra las características mecánicas y eléctricas del motor stepper NEMA 17 que puede ser empleado para el diseño del autómata.

Tabla VI. **Características motor stepper NEMA 17**

Electrical Specifications

Model	Step angel (°)	Motor Length L(mm)	Rate Voltage (V)	Rate Current (A)	Phase Resistance (Ω)	Phase Inductance (mH)	Holding Torque (g.cm)	Lead Wire (NO.)	Detent Torque (g.cm)	Motor Weight (kg)
42BYGHW205	1.8	34	12	0.3	40	18	1200	6	200	0.2
42BYGHW208	1.8	34	12	0.4	30	37	2600	4	200	0.2
42BYGHW213	1.8	34	12	0.6	20	15	2000	6	200	0.2
42BYGHW602	1.8	40	12	0.4	30	21	2500	6	220	0.26
42BYGHW603	1.8	40	12	0.4	30	40	3800	4	220	0.26
42BYGHW607	1.8	40	3.6	1.2	3	3.2	2600	6	220	0.26
42BYGHW609	1.8	40	3.4	1.7	2	3	4000	4	220	0.26
42BYGHW801	1.8	48	12	0.4	30	25	3200	6	280	0.36
42BYGHW804	1.8	48	3.6	1.2	3	5	4500	4	280	0.36
42BYGHW811	1.8	48	3.1	2.3	1.25	1.8	4800	4	280	0.36

Fuente: *Blog de PatagoniaTec Electronica*. http://saber.patagoniatec.com/ nema-17-42byghw208-arduino-argentina-ptec/1-0x0/. Consulta: 7 de noviembre de 2016.

Las características más importantes y las que definen el tipo de motor a utilizar son las de holding torque y amperios.

- Microcontrolador: el microcontrolador PIC16F877a es de 8 bits aunque los de 16 y 32 bits son superiores al de 8 bits; el de microcontrolador de 8 bits es el más común en el mercado y es preferido para casi cualquier aplicación; son de bajo costo, confiables, de fácil programación y pueden comprarse en el mercado local e internacional. En nuestro país un microcontrolador de 8 bit el cual puede ser el PIC 16F877a puede costar alrededor de Q 60,00, aproximadamente USD 8,00. Al escoger el microcontrolador hay que tomar en consideración varios aspectos

importantes con base en la aplicación, por ejemplo, si se desea analizar información y con base en el resultado tomar una decisión se debe considerar como característica crítica el procesamiento de datos.

Para este autómata, las características importantes son: comunicación serial RS232, entradas y salidas digitales y controlador de motores stepper. Cualquier microcontrolador que reúna estas características puede ser empleado para en el autómata.

Módulo de iluminación: el éxito de un buen reconocimiento de imagen depende de una buena difusión de luz; con una buena imagen se garantiza la extracción de toda la información que para nuestro caso son los datos característicos de los medidores de energía eléctrica; el módulo de iluminación que se va a utilizar es el módulo de iluminación RL1424 de Advance Ilumination.

Figura 83. **Características módulo RL1424**

Electrical Specifications	Color	24v Current	All Other Controls
	880, 660, 625	0.06 A	0.06 A Max
	395, 470, 520, WHI	0.04 A	0.04 A Max
Normal Operating Temperature	0 - 60°C		
Weight (g)	226.8 g (8.0 oz.)		
Standard Cable Information	Up to 2 meters (80") long - 105°C rated PVC jacket, foil shield with drain.		
Photobiological Risk Factor IEC 62471	**Exempt Applicable Wavelengths:** 880 **Group 1 (Low-Risk) Applicable Wavelengths:** 470, 520, 625, 660, WHI **Group 2 (Medium-Risk) Applicable Wavelengths:** 395		
Compliance	CE RoHS IEC 62471		
IP Rating	IP 40		
Lumen Maintenance	L70 = 50,000 hours		

Fuente: *Módulo de iluminación RL1424 de avance iluminación*. http://www.avanceytec.com.mx/productos/iluminacion/leds-para-iluminacion/. Consulta: 7 de noviembre de 2016.

- Cámara de visión artificial: este sensor es una de las partes más fundamentales del autómata ya que por medio de este sensor se captura las imágenes; puede ser del tipo ethernet o seriales, depende mucho de la aplicación y la velocidad de captura. La cámara que se va a utilizar es la acA640-90; a continuación, se muestran las características más importantes.

Figura 84. **Características cámara acA640-90**

Basler ace Cameras	Resolution (H x V pixels)	Sensor	Sensor Technology	Optical Sensor Size (in.)	Pixel Size (m)	Frame Rate	Pow Cons (PoE
acA640-90gm/gc	659 x 494	Sony ICX424	Progressive Scan CCD	1/3	7.4 x 7.4	90	3.1 V
acA640-120gm/gc	659 x 494	Sony ICX618	Progressive Scan CCD	1/4	5.6 x 5.6	100	2.3 V
acA645-100gm/gc	659 x 494	Sony ICX414	Progressive Scan CCD	1/2	9.9 x 9.9	100	3.6 V
acA750-30gm/gc	752 x 580	Sony ICX409	Interlaced Scan CCD	1/3	6.5 x 6.25	30	2.5 V
acA780-75gm/gc	782 x 582	Sony ICX415	Progressive Scan CCD	1/2	8.3 x 8.3	75	3.6 V
acA1300-30gm/gc	1296 x 966	Sony ICX445	Progressive Scan CCD	1/3	3.75 x 3.75	30	2.5 V
acA1300-60gm/gc	1280 x 1024	e2v EV76C560	CMOS, Global Shutter	1/1.8	5.3 x 5.3	60	<3.0
acA1600-20gm/gc	1628 x 1236	Sony ICX274	Progressive Scan CCD	1/1.8	4.4 x 4.4	20	3.4 V
acA1600-60gm/gc	1600 x 1200	e2v EV76C570	CMOS, Global Shutter	1/1.8	4.5 x 4.5	60	2.5 V
acA2000-50gm/gc	2048 x 1088	CMOSIS CMV2000	CMOS, Global Shutter	2/3	5.5 x 5.5	50	3.4 V
acA2040-25gm/gc	2048 x 2048	CMOSIS CMV4000	CMOS, Global Shutter	1	5.5 x 5.5	25	3.4 V
acA2500-14gm/gc	2592 x 1944	Aptina MT9P	CMOS, Rolling shutter	1/2.5	2.2 x 2.2	14	2.5 V

Fuente: *Basler ace cameras*. http://www.ni.com/datasheet/pdf/en/ds-414. Consulta: 7 de noviembre de 2016.

Software de desarrollo Labview: se prefiere utilizar como *software* de desarrollo Labview por su capacidad y velocidad de procesamiento, integración con otros dispositivos, escalabilidad del *hardware*, facilidad de uso del *software*, entre otros. A continuación, se presenta la comparación del desempeño de la velocidad del algoritmo comparado con otros *software* del mercado en este caso el líder del mercado.

Tabla VII. **Comparación de velocidad de desempeño del algoritmo**

	Velocidad del *software* NI Vision (ms)	Velocidad del *software* Visión Líder (ms)	Incremento de velocidad de NI
Histograma	0,91	2,03	2,2X
Transformada Geométrica	3,1	10,3	3,3X
Morfología	1,8	5,9	3,3X
OCR	3,3	5,9	1,8X
Igualación Geométrica	93,0	149,8	1,6X
Clasificación de Objetos	7,5	–	–

Fuente: *Notas técnicas.* http://www.ni.com/white-paper/5603/es/. Consulta: 7 de noviembre de 2016.

Se considera la licencia profesional que permite implementar aplicaciones profesionales; incluye la habilidad de desarrollar e implementar ejecutables e instaladores; ofrece herramientas de pruebas unitarias y de análisis de código e integración de control de versiones para desarrollos en equipo. *National Instrument* en su página web tiene disponible las características de las licencias: Base *Edition, Full edition y profesional edition*.

En la figura 85 se observa una pequeña comparación de estas tres opciones del *software* de desarrollo Labview; para más información puede consultarse en su página de internet http://www.ni.com/labview/buy/esa/.

Figura 85. **Comparación de las ediciones de Labview**

Característica	LabVIEW Base	LabVIEW Completo	LabVIEW Profesional	Software Suites de NI
Soporte para SO	Windows	Windows, Mac, Linux		Windows
Idiomas	Inglés, Francés, Alemán, Chino Simplificado, Japonés, Coreano			
Programa de Servicio Estándar				
Soporte técnico	Un año incluido			
Formación/capacitación en línea 24/7	LabVIEW Core 1, 2	LabVIEW Core 1, 2, 3*		Toda la formación/capacitación de NI
Actualizaciones de software	Un año incluido			

Fuente: *Compre LabVIEW.* http://www.ni.com/labview/buy/esa/. Consulta: 7 de noviembre de 2016.

4.3. Dimensionamiento y movimiento de la estructura metálica

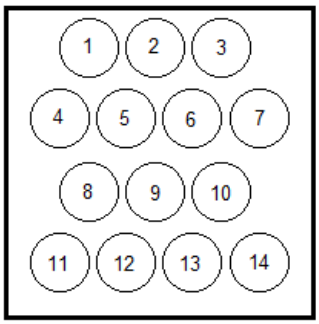
La estructura metálica comúnmente llamada *rack* es donde se colocan los medidores de energía eléctrica y son capaces de contener 28 medidores, 14 por cada lado, las dimensiones físicas son las siguientes: 0,55 m x 0,84 m x 1,35 m.

Figura 86. **Estructura metálica o *rack***

Fuente: Empresa eléctrica de Guatemala.

La estructura del autómata debe ser capaz de acoplarse a la estructura del *rack* para capturar las imágenes de una forma rápida y eficiente. Debido a que cada lado de la estructura contiene 14 medidores, esto implica tomar 14 fotografías una por cada medidor de energía eléctrica y 14 por cada lado; la estructura del autómata se diseña como la estructura de un plotter, pero con una matriz de posiciónes más sencilla que es 4 columnas y 4 filas, y como se muestra en la siguiente figura.

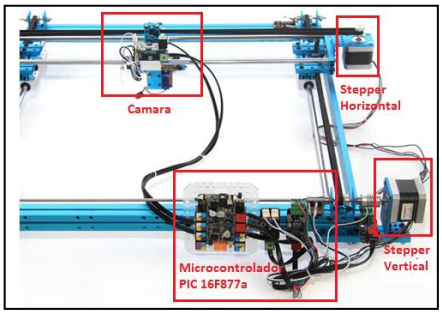
Figura 87. **Matriz de posiciones 4X4**



Fuente: elaboración propia.

Adicional, es necesario el control automatizado de un sistema de movimiento que se encargue de posicionar la cámara fotográfica en cada posición donde están ubicados los medidores para la posterior captura de una imagen de cada medidor; el sistema de control está conformado por el microcontrolador que es el encargado de controlar el movimiento de los motores *stepper* quien recibe el comando de posición por el *software* de desarrollo a través del puerto serial. Los motores *stepper* están encargados del movimiento vertical y horizontal, consiguiendo de esta forma dos grados de libertad.

Figura 88. **Estructura metálica**

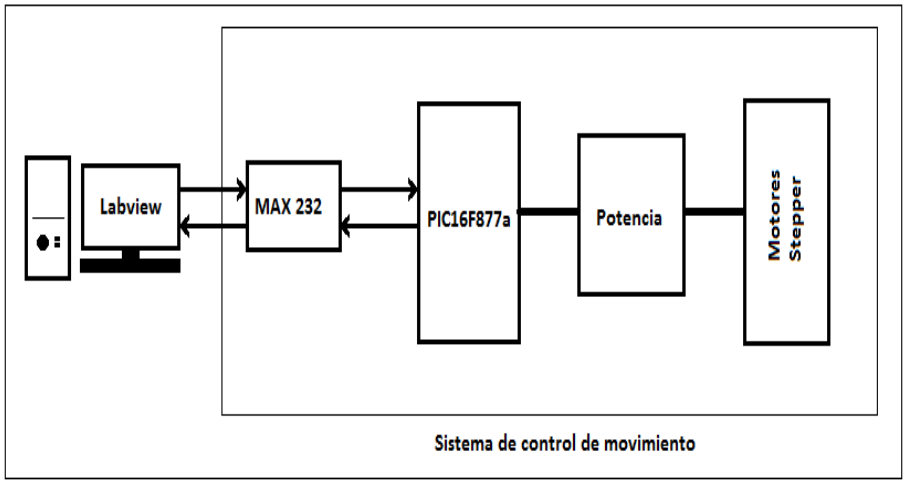


Fuente: elaboración propia, utilizando *software* Labview.

4.4. Diseño del algoritmo para el microcontrolador PIC16F877A

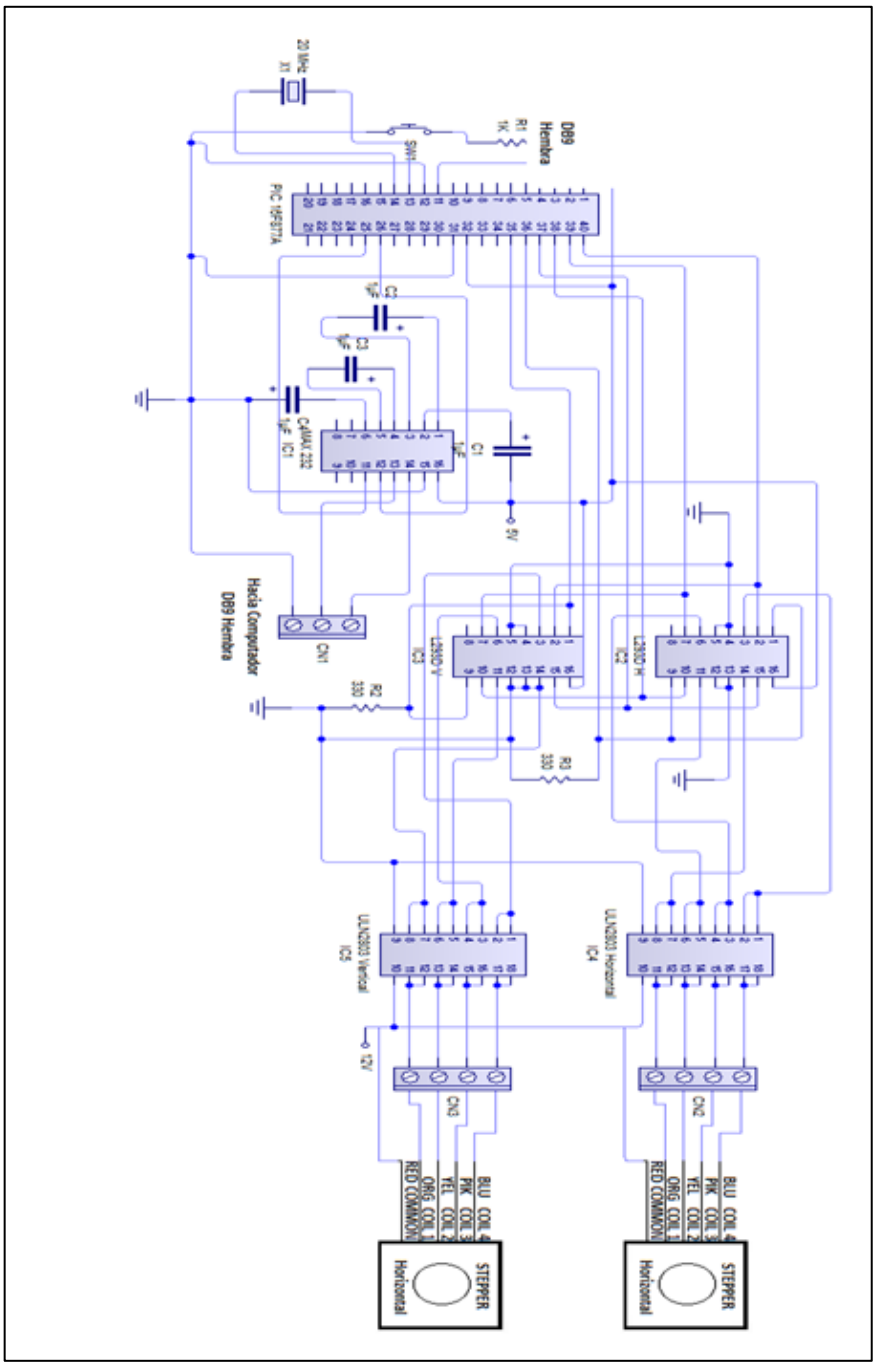
Como se ha mencionado anteriormente, el microcontrolador es el encargado del movimiento de los motores paso a paso horizontal y vertical, el microcontrolador recibe el comando de posicionamiento que es enviado por el programa en Labview a través del puerto serial RS232, en el siguiente diagrama de bloques se muestra gráficamente quienes interactúan en este proceso.

Figura 89. **Diagrama de bloques control de movimiento**



Fuente: elaboración propia.

Figura 90. **Diagrama eléctrico control de movimiento**

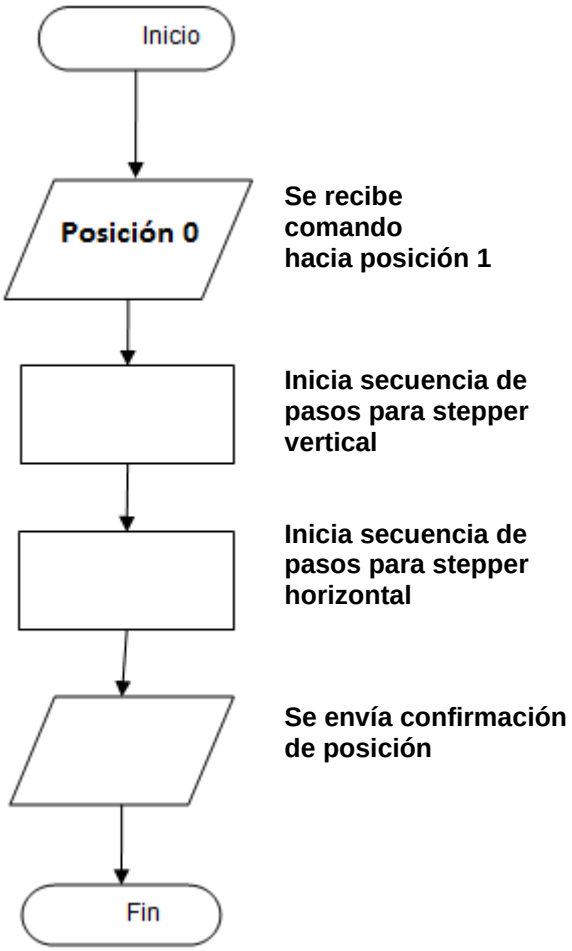


Fuente: elaboración propia, utilizando *software* Labview.

El algoritmo descrito es la ruta a seguir en la programación y se realizará con un diagrama de flujo que es un conjunto prescrito de instrucciones ordenadas y bien definidas que permitirá llevar a cabo el posicionamiento de la cámara para capturar la imagen.

En el siguiente diagrama de flujo se puede observar que inicia cuando el programa Labview envía el comando de posición "1" al microcontrolador por medio del puerto serial RS232; a partir de la posición "0", que es la de reposo, el microcontrolador se encargara del movimiento hasta llegar a la posición "1" para luego retroalimentar al programa desarrollado en Labview que se encuentra en la posición deseada "1" que captura la imagen del medidor de energía eléctrica; este proceso sigue hasta capturar la imagen de todas las posiciones.

Figura 91. **Diagrama de flujo control de movimiento**



Fuente: elaboración propia.

A continuación, se detalla el código de programación o algoritmo del control de movimiento para el microcontrolador PIC 16F877A, el código de programación fue realizado en el programa de PIC SIMULATOR IDE V7.4 modo evaluación.

```
Dim i As Byte  'inicio de rutina
```

```
AllDigital
Define STEP_A_REG = PORTB  'stepper horizontal
Define STEP_A_BIT = 7  'stepper horizontal
Define STEP_B_REG = PORTB  'stepper horizontal
Define STEP_B_BIT = 6  'stepper horizontal
Define STEP_C_REG = PORTB  'stepper horizontal
Define STEP_C_BIT = 5  'stepper horizontal
Define STEP_D_REG = PORTB  'stepper horizontal
Define STEP_D_BIT = 4  'stepper horizontal
Define STEP_MODE = 2
TRISB.3 = 0
TRISB.2 = 0
StepHold
inicio:
Serin PORTC.7, 9600, i
If i = "1" Then  '1 es para inicio de posición 1
Serout PORTC.6, 9600, "captura", CrLf  'Primera captura
Goto inicio
Endif
If i = "2" Then  'posición 2
PORTB.3 = 1
StepCW 20, 300  '20 pasos para la posición 2
Serout PORTC.6, 9600, "captura", CrLf  'captura para labview
PORTB.3 = 0
Goto inicio
Endif
If i = "3" Then  'posición 3
PORTB.3 = 1
StepCW 20, 300  '20 pasos para la posición 3
```

```
Serout PORTC.6, 9600, "captura", CrLf  'captura para labview
PORTB.3 = 0
Goto inicio
Endif
If i = "4" Then  'posición 4
PORTB.3 = 1
StepCCW 40, 300  '40 pasos para regresar a la posición 1
PORTB.3 = 0
PORTB.2 = 1
StepCW 20, 300  '20 pasos para bajar a la segunda fila
Serout PORTC.6, 9600, "captura", CrLf  'captura para labview
PORTB.2 = 0
Goto inicio
Endif
If i = "5" Then  'posición 5
PORTB.3 = 1
StepCW 20, 300  '20 pasos para la posición 5
Serout PORTC.6, 9600, "captura", CrLf  'captura para labview
PORTB.3 = 0
Goto inicio
Endif
If i = "6" Then  'posición 6
PORTB.3 = 1
StepCW 20, 300  '20 pasos para la posición 6
Serout PORTC.6, 9600, "captura", CrLf  'captura para labview
PORTB.3 = 0
Goto inicio
Endif
If i = "7" Then  'posición 7
```

```
PORTB.3 = 1
StepCW 20, 300  '20 pasos para la posición 7
Serout PORTC.6, 9600, "captura", CrLf  'captura para labview
PORTB.3 = 0
Goto inicio
Endif
If i = "8" Then  'posición 8
PORTB.3 = 1
StepCCW 60, 600  '60 pasos para regresar a la posición 4
PORTB.3 = 0
PORTB.2 = 1
StepCW 20, 300  '20 pasos para bajar a la tercera fila
Serout PORTC.6, 9600, "captura", CrLf  'captura para labview
PORTB.2 = 0
Goto inicio
Endif
If i = "9" Then  'posición 9
PORTB.3 = 1
StepCW 20, 300  '20 pasos para la posición 9
Serout PORTC.6, 9600, "captura", CrLf  'captura para labview
PORTB.3 = 0
Goto inicio
Endif
If i = "10" Then  'posición 10
PORTB.3 = 1
StepCW 20, 300  '20 pasos para la posición 10
Serout PORTC.6, 9600, "captura", CrLf  'captura para labview
PORTB.3 = 0
Goto inicio
```

```
Endif
If i = "11" Then  'posición 1
PORTB.3 = 1
StepCCW 40, 600  '40 pasos para regresar a la posición 8
PORTB.3 = 0
PORTB.2 = 1
StepCW 20, 300  '20 pasos para bajar a la cuarta fila
Serout PORTC.6, 9600, "captura", CrLf  'captura para labview
PORTB.2 = 0
Goto inicio
Endif
If i = "12" Then  'posición 12
PORTB.3 = 1
StepCW 20, 300  '20 pasos para la posición 12
Serout PORTC.6, 9600, "captura", CrLf  'captura para labview
PORTB.3 = 0
Goto inicio
Endif
If i = "13" Then  'posición 13
PORTB.3 = 1
StepCW 20, 300  '20 pasos para la posición 13
Serout PORTC.6, 9600, "captura", CrLf  'captura para labview
PORTB.3 = 0
Goto inicio
Endif
If i = "14" Then  'posición 114
PORTB.3 = 1
StepCW 20, 300  '20 pasos para la posición 14
Serout PORTC.6, 9600, "captura", CrLf  'captura para labview
```

```
WaitMs 1000  'tiempo en lo que labview captura la imagen
WaitMs 2000  'tiempo en lo que labview captura la imagen
StepCCW 60, 300  'regresando a la posición 1
PORTB.3 = 0  'regresando a la posición 1
PORTB.2 = 1  'regresando a la posición 1
StepCCW 60, 300  'regresando a la posición 1
PORTB.2 = 0  'regresando a la posición 1
Goto inicio  'regresando a la posición 1
Endif  'regresando a la posición 1
Goto inicio
```

4.5. Diseño del algoritmo para la captura y extracción de datos característicos

El programa central debe ser capaz de interactuar con el microcontrolador encargado de posicionar la cámara para posteriormente capturar la imagen de los medidores de energía eléctrica.

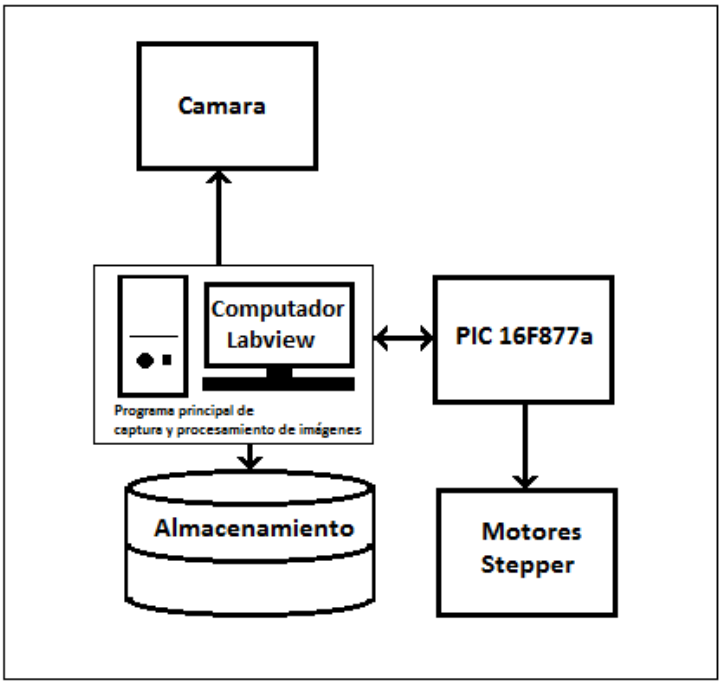
Cuando la imagen es capturada; inicia el procesamiento para la extracción de los datos característicos de los medidores: número de serie de fábrica, número de serie de la utility y lectura en kWh.

Se puede extraer más información aprovechando que se tiene la imagen del medidor; esta información adicional puede ser voltaje de operación, forma, kH, entre otros, como también un análisis de las partes mecánicas frontales que los componen.

En el siguiente diagrama de bloques se pueden encontrar los dispositivos que interactúan para la captura de imágenes; como parte principal del autómata

se encuentra el computador con el *software* de desarrollo Labview donde se ejecutará el programa central encargado de la captura, procesamiento, almacenamiento de imágenes y comandos para el posicionamiento de la cámara.

Figura 92. **Diagrama de bloques de captura y procesamiento de imágenes**

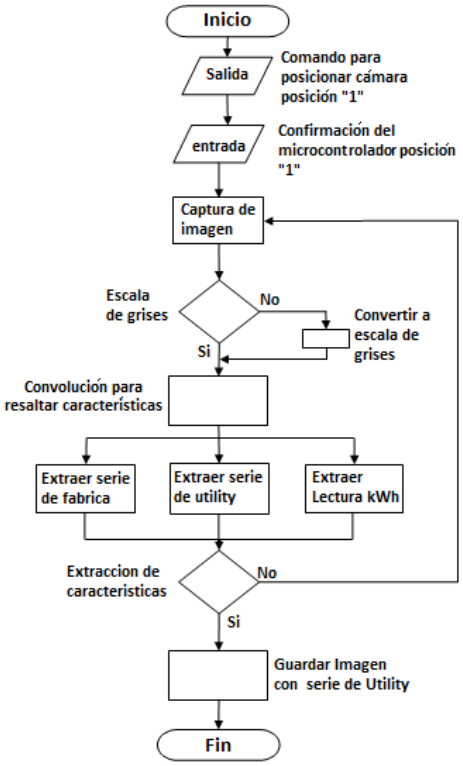


Fuente: elaboración propia.

El diagrama de flujo de la figura 91, muestra la ruta a seguir en la programación, como se puede observar hay datos de salida y entrada; estos son para el microcontrolador y antes de iniciar con el procesamiento de imágenes, se captura la imagen para luego seguir con los procesos involucrados en el procesamiento de imágenes para obtener el número de serie de fábrica, número de serie de la utility y lectura en kWh.

Como se ha mencionado, es una ruta en la programación y en el *software* de desarrollo Labview se tienen otros retos y pruebas que son necesarios para llegar al resultado deseado.

Figura 93. **Diagrama de flujo para la captura y el procesamiento de imágenes**

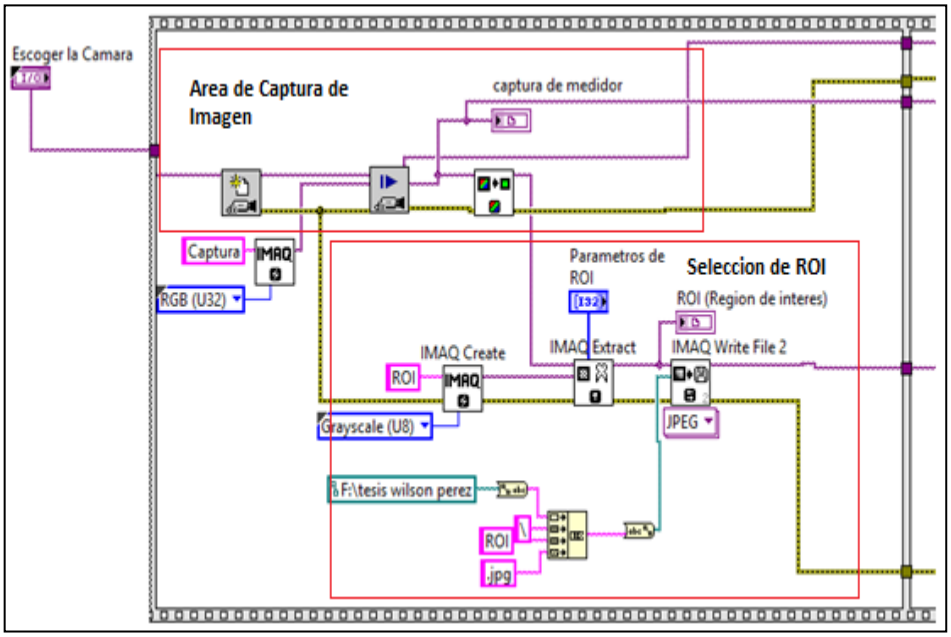


Fuente: elaboración propia.

El algoritmo para la extracción de datos característicos consta de tres partes importantes, la primera parte del algoritmo se muestra en la figura 94. Esta primera parte a su vez está compuesta de dos partes: la primera parte es la captura de la imagen del medidor y la segunda parte es la región de interés ROI; el ROI es un array de 4 variables (izquierda, derecha, arriba, fondo y

rotación) que indica la ubicación del área de interés y esta ubicación será extraída de la imagen y se guardará la imagen con el nombre ROI.

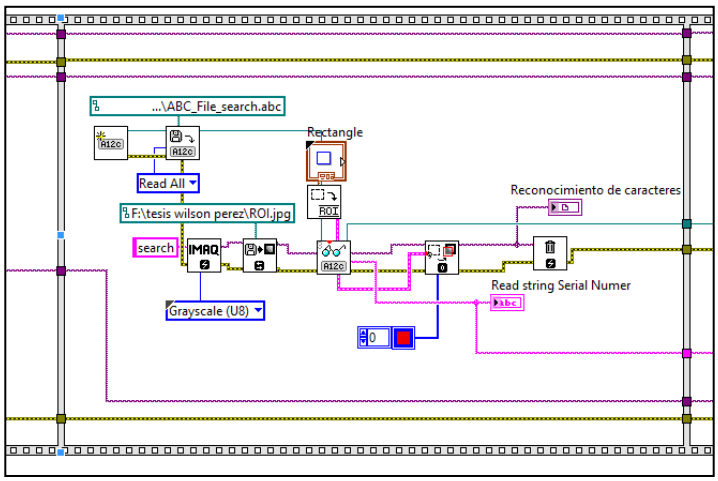
Figura 94. **Captura de imagen del medidor y ROI**



Fuente: elaboración propia, utilizando *software* Labview.

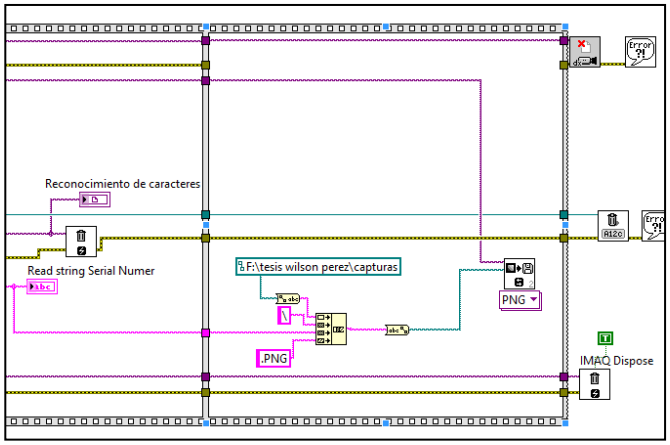
En la segunda parte importante, figura 95, se trata únicamente el ROI extraído en la primera parte; esta parte del algoritmo es la encargada del reconocimiento óptico de caracteres OCR; cuando los caracteres coinciden con el patrón, la función de Labview OCR devuelve el *string* que reconoce en la figura del ROI. La tercera parte del algoritmo se encarga de guardar la imagen con el nombre del *string* extraído por el OCR este código se muestra en la figura 96.

Figura 95. **Reconocimiento óptico de caracteres OCR**



Fuente: elaboración propia, utilizando *software* Labview.

Figura 96. **Guardar imagen**



Fuente: elaboración propia, utilizando *software* Labview.

4.6. Comunicación entre el microcontrolador y *software* de desarrollo Labview

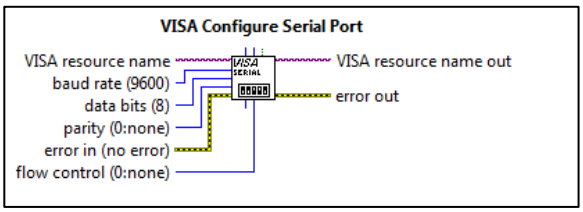
Las comunicaciones entre el *software* de desarrollo Labview y el microcontrolador son parte fundamental de todo el sistema; es de vital importancia que Labview envíe los comandos de control para el posicionamiento de la cámara y luego se capture la imagen del medidor de energía eléctrica; todo este proceso gobernado por Labview y el microcontrolador. Los recursos necesarios en este proceso son los siguientes:

- MAX232: convierte las señales de un puerto RS232 a señales compatibles con niveles lógicos TTL; sirve como interfaz de transmisión y recepción para las señales RX, TX, CTS y RTS.

- PIC16F877a: El microcontrolador PIC 16F877a tiene integrado el módulo de comunicación básico llamado UART que permite una comunicación serial implementando el protocolo RS232, esta función del microcontrolador facilita la comunicación entre dos dispositivos que utilicen el mismo protocolo, en este caso el computador y microcontrolador; el microcontrolador utiliza los pines RC7 para transmisión y el RC6 para la recepción de datos y a su vez estos dos pines conectados al MAX232 que se utiliza para convertir los niveles lógicos TTL a niveles de transmisión RS232.

Las funciones de Labview que permiten la comunicación con otro dispositivo o a través del puerto serial de la computadora se llaman VISA Configure Serial Port, VISA write y VISA read que se encargan de inicializar el puerto, especificar las configuraciones del puerto serial, envío y recepción de información.

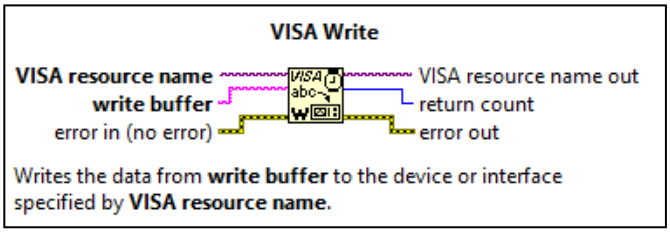
Figura 97. **VISA *configure Serial Port***



Fuente: elaboración propia, utilizando *software* Labview.

- VISA *resource name*: puerto físico del computador o controlador.
- *Buad rate*: la tasa de transmisión y por defecto está configurado a 9600.
- *Data bits*: número de bits de data entrante.
- Parity: especifica la paridad utilizada por cada trama de información transmitida o recibida.
- *Flow control*: especifica el tipo de control utilizado como mecanismo de transferencia.
- *Stop bits*: especifica la cantidad de bits utilizados para indicar fin de la trama.
- VISA *resource name out*: especifica el recurso a ser utilizado y también controla y especifica la sesión y clase.

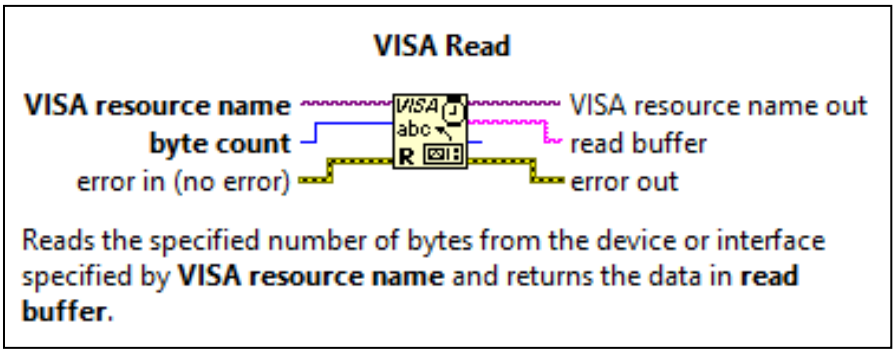
Figura 98. **VISA *write***



Fuente: elaboración propia, utilizando *software* Labview.

- *Write buffer*: información que se envía tipo string
- VISA *resource name*: referencia de la sesión del puerto por el cual se enviará información.

Figura 99. **VISA *read***



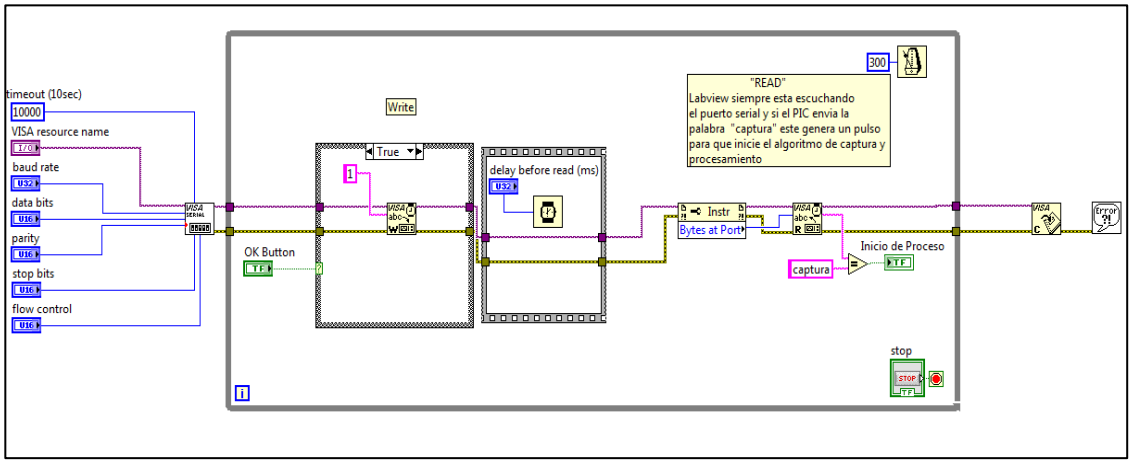
Fuente: elaboración propia, utilizando *software* Labview.

- VISA *resource name*: referencia de la sesión del puerto por el cual se enviará información.
- *Read buffer*: lee el número específico de bytes desde el dispositivo o interfaz que va hacia la entrada VISA resource name.
- *Byte count*: tamaño del buffer que se escribió en el puerto.

En la figura 100 se observa el instrumento virtual para la comunicación entre el microcontrolador PIC 16F877A y la máquina con Labview, se configura la función VISA configure serial port, esto crea una sesión que se conecta a las funciones de VISA Read y Write, VISA Read siempre está en modo escucha por medio del ciclo while; cuando el microcontrolador envía la palabra captura por el puerto serial, este activa un booleano que ejecuta todo el algoritmo de captura y procesamiento de imágenes y VISA Write depende de un botón OK;

cuando es presionado, este envía un carácter 1 al microcontrolador para que inicie todo el proceso.

Figura 100. **VI comunicación entre PIC y Labview**



Fuente: elaboración propia, utilizando *software* Labview.

4.7. Análisis de costos

Se tiene como objetivo analizar la viabilidad de este proyecto y a continuación se describe y se indican los costos asociados en el desarrollo e implementación del autómata. Es necesario considerar el impacto de la tasa de cambio de dólares americanos a quetzales ya que la compra de materiales y adquisición del *software* es en moneda extranjera siendo dólares americanos. Los costos asociados se encuentran enlistados a continuación.

Tabla VIII. **Costos de licencias y materiales**

Núm.	Cantidad	Descripción	Costo Unitario	Subtotal
1	1	Licencia *software* de desarrollo Labview completo	$ 5 500,00	$ 5 500,00
2	1	Módulo *Vision and Motion*	$ 4 110,00	$ 4 110,00
3	1	Módulo *Report Toolkit*	$ 570,00	$ 570,00
4	1	Cámara de visión artificial	$ 615,00	$ 615,00
5	2	Motores y *driver*	$ 180,00	$ 360,00
6	1	Estructura metálica	$ 250,00	$ 250,00
7	1	Microcontrolador 16F877A	$ 13,00	$ 13,00
9	1	Iluminación para visión artificial	$ 375,00	$ 375,00
10	1	Máquina de escritorio	$ 1 000,00	$ 1 000,00
			Total	**$ 12 793,00**

Fuente: elaboración propia.

También, es necesario considerar un pago anual por concepto de soporte y renovación de licencia que es del 20 % del costo total de licencia y módulos adicionales tales como: *Report Toolkit*, *Vision and Motion*.

Tabla IX. **Soporte y renovación anual de licencia base**

No.	Cantidad	Descripción	Costo Unitario	SubTotal
1	1	Licencia *software* de desarrollo Labview completo	$ 5 500,00	$ 5 500,00
2	1	Módulo *Vision and Motion*	$ 4 110,00	$ 4 110,00
3	1	Módulo *Report Toolkit*	$ 570,00	$ 570,00
		Total		**$ 10 180,00**
		Renovación anual (20 %)		**$ 2 036,00**

Fuente: elaboración propia.

Muchas veces no se considera las horas-hombre en el desarrollo e implementación de los autómatas y es muy importante considerarlos debido al desarrollo del algoritmo y pruebas con el *software*, equipos o materiales. Los dispositivos y *software* por si solos no cobran importancia hasta que todos trabajan con perfecta sincronía previamente configurados por el ingeniero de desarrollo; en la siguiente tabla solo se consideran el diseño del algoritmo con Labview y las pruebas de campo.

Tabla X. **Horas hombre ingeniero de desarrollo**

No.	Descripción	Horas
1	Pruebas de iluminación	4
2	Prueba de capturas de imágenes	8
4	Comunicación RS232 con el microcontrolador PIC16F877A	8
5	Diseño del Algoritmo de Posicionamiento	80
6	Diseño del algoritmo para la extracción de datos característicos	160
7	Implementación de los algoritmos	240
8	Pruebas de integración con todos los dispositivos	120
9	Prueba finales y producción	120
		740
	Total de horas y costo asociado	Q39 000,00

Fuente: elaboración propia.

CONCLUSIONES

1. Se determinó que con el microcontrolador PIC 16F877A pueden controlarse motores paso a paso *stepper* y su módulo de comunicación serial RS232 lo hace ideal para la conexión con otros dispositivos.

2. Se comprobó que con Labview se pueden integrar dispositivos como el microcontrolador PIC16F877A por medio del puerto serial RS232.

3. Se determinó que Labview como *software* de programación es una buena opción para el desarrollo de algoritmos que permitan captura de imágenes, control de dispositivos a través del puerto serial y procesamiento de imágenes.

4. Aplicando técnicas de procesamiento digital de imágenes con el *software* Labview pueden resaltarse características importantes en las imágenes que permitan la extracción de datos: tamaño de los objetos, textos, valores numéricos, entre otros.

RECOMENDACIONES

1. Sustituir al computador por un equipo dedicado a la ejecución del programa de captura y procesamiento de imágenes; este equipo puede ser hardware de National Instrument como el CompactRIO con Windows o Linux embebido; con esto se logra un menor tiempo en el procesamiento de imágenes, seguridad y ejecución del código más confiable.

2. La integración de Labview con el microcontrolador y motores paso a paso puede funcionar muy bien, pero se puede mejorar si se utiliza hardware y *software* dedicado para este trabajo el cual puede ser el módulo de funciones Motion de Labview y el controlador de potencia de motores paso a paso, con el fin de conseguir micropasos, alto torque y confiabilidad en el proceso.

BIBLIOGRAFÍA

1. BODINGTON ESTEVA, Christian. *Basic para microcontroladores PIC*. Canada: Spanish Edition, 2006. 362 p.

2. *Comparativa de microcontroladores actuales*. [en línea]. <http:// server-die.alc.upv.es/asignaturas/lsed/2002 03/Micros/ downloads/ trabajo.pdf>. [Consulta: 21 de noviembre de 2016].

3. GORDILLO ERAZO, Lenin Edwin; YÁNEZ ROCA, Jorge Luis. *Aplicación de visión con LabView para la detección de frascos con turbiedades*. Trabajo de graduación de Ing. en electrónica y Telecomunicaciones, Facultad de Ingeniería en Electricidad y Computación, Escuela Superior Politécnica Del Litoral, 2009, 389 p.

4. *Motores de paso o steppers motors*. [en línea]. <http://galia.fc.uaslp.mx/~cantocar/microcontroladores/SLIDES_8051_PDF/21_MOTOR.PDF>. [Consulta: 21 de noviembre de 2016].

5. *Motor paso a paso*. [en línea]. <http://grupovirtus.org/moodle/pluginfile.php/4511/mod_resource/content/1/SEMANA_8/material_1.pdf>. [Consulta: 21 de noviembre de 2016].

6. National Instruments. *IMAQ Vision Concepts Manual*. National Instruments corporation: Edition Part Number 322916A-01, 2000. 313 p.

7. _____. *IMAQ Vision for LabVIEW*. National Instruments corporation: Edition Part Number 371007A-01, 2004. 141 p.

8. _____. *LabVIEW user manual*. National Instruments corporation: Edition Part Number 320999E-0, 2003. 349 p.

9. *PIC16F887A Data Sheet*. [en línea]. <http://ww1.microchip.com/downloads/en/DeviceDoc/39582C.pdf>. [Consulta: 21 de marzo de 2017].

10. *PIC16 SIMULATOR IDE*. [en línea]. <http://www.oshonsoft.com/pic16.html>. [Consulta: 10 de noviembre de 2016].

11. *Procesamiento digital de imágenes*. [en línea]. <http://verona.fi-p.unam.mx/boris/teachingnotes/Introduccion.pdf>. [Consulta: 21 de noviembre de 2016].

12. _____. [en línea]. <http://catarina.udlap.mx/u_dl_a/tales/documentos/mel/gonzalez_g_ra/capitulo2.pdf>. [Consulta: 21 de noviembre de 2016].

13. *Sección de motores paso a paso*. [en línea]. <http://catarina.udlap.mx/u_dl_a/tales/documentos/lep/hernandez_b_ii/capitulo3.pdf>. [Consulta: 21 de noviembre de 2016].

14. TIUL VALENZUELA, Hugo Leonel. *Diseño de un sistema de control de temperatura y detección de humo con el microcontrolador pic 16f887, utilizando tecnología inalámbrica bluetooth para la sala de servidores del centro de cálculo e investigación educativa de la*

Facultad de Ingeniería, USAC. Trabajo de graduación de Ing. Electrónica. Universidad de San Carlos de Guatemala, Facultad de Ingeniería, 2014. 98 p.

15. VAQUIER MARTÍNEZ, Ricardo y JUAN AGUILAR, José Oswaldo. *Sistema de visión artificial para la detección de cuerpos sólidos en botellas a través del procesamiento de imágenes con LABVIEW*. Trabajo de graduación de Ing. Electrónico y Telecomunicaciones, Universidad Veracruzana, 2014. 110 p.

16. *10 considerations when choosing vision software*. [en línea]. <http://sine.ni.com/np/app/main/p/ap/vision/lang/en/pg/1/sn/n17:vision,n21:11601/fmid/3053/>. [Consulta: 21 de noviembre de 2016].