UNIVERSIDAD DE QUINTANA ROO

División de Ciencias e Ingeniería

Diseño e implementación de un sistema PLC (Power Line Comunication)

Tesis para obtener el grado de

Ingeniero en Redes

PRESENTA

Kenia Nuñez Gualberto

Director de Tesis

Dr. Freddy Ignacio Chan Puc

Asesores

Dr. Jaime Silverio Ortegón Aguilar

MC. Emmanuel Torres Montalvo

Chetumal, Quintana Roo, México, Noviembre de 2010.

UNIVERSIDAD DE QUINTANA ROO

División de Ciencias e Ingeniería

Tesis elaborada bajo supervisión del Comité de Asesoría y aprobada como requisito parcial para obtener el grado de:

INGENIERO EN REDES

Comité deTesis

Director: _____

Dr. Freddy Ignacio Chan Puc

Asesor: _____

Dr. Jaime Silverio Ortegón Aguilar

Asesor: _____

MC. Emmanuel Torres Montalvo

Chetumal, Quintana Roo, México, Noviembre de 2010.

AGRADECIMIENTOS

Agradezco a Dios por la vida, por el conocimiento que me ha otorgado y por dotarme de los medios necesarios para culminar esta etapa.

Gracias a mi familia que me ha apoyado en todo.

Gracias a Martín por que ha estado conmigo a lo largo de toda la carrera y sobre todo en los momentos que sufrí y que disfruté haciendo este trabajo de tesis.

Gracias al comité de tesis, en especial al profe Freddy por ser mi director, porque tuvo paciencia, dedicó tiempo y me apoyó en todo lo que pudo a lo largo del desarrollo de este trabajo de tesis.

Gracias a la academia de redes, porque cada uno de ellos colaboró para mi formación como profesionista. En especial al profe Vladi por brindarme su amistad y al profe Jaime por ser una gran persona.

Gracias a la División de Ciencias e Ingeniería porque este trabajo fue financiado en la Convocatoria 2010 “Apoyo a la titulación”.

DEDICATORIA

A mi papá por que trabaja arduamente cada día, por que se que es un gran sueño para él que todas sus hijas terminen sus estudios. Y ahora soy la tercera en hacerle ese sueño realidad.

A mi madre por todos sus esfuerzos, por que ha valido la pena sus madrugadas y sus noches sin dormir.

A mis hermanas por el poyo que me han dado, en especial a Leti y a Dora.

RESUMEN

La domótica controla y automatiza la gestión inteligente de la vivienda. Aporta confort, comunicación y seguridad, además de gestionar eficientemente el uso de la energía, favoreciendo el ahorro de agua, electricidad y combustibles.[1]

La mayoría de los hogares cuentan con aparatos que se conectan a la línea eléctrica para poder realizar sus funciones, algunos se controlan manualmente y otros con control remoto; el uso de dichos aparatos genera un consumo energético. La Comisión Federal de Electricidad (CFE) en su página de Internet propone algunos consejos para el ahorro de energía, principalmente para disminuir el costo en el recibo de luz y preservar los recursos naturales.[2]

Las lámparas y los focos son las cargas eléctricas más comunes en una casa, por lo que un sistema que permita controlar el encendido y apagado de los mismos, proporcionaría un ahorro energético, confort y comodidad.

Es posible implementar un sistema de control y automatización del encendido y apagado de luminarias basado en microcontroladores utilizando como medio de comunicación la línea eléctrica y así reduciendo gastos de instalación.

La comodidad y el confort son aspectos muy valorados en un hogar. Pero el vivir cómodo y confortable tiene un precio al cual no todos tienen acceso.

El diseño de un sistema de comunicación por la línea eléctrica permite utilizar la instalación eléctrica ya existente, dando como resultado que se pueda controlar y automatizar los aparatos que se conectan a la línea eléctrica, por ejemplo el encendido y apagado de lámparas; dando como resultado comodidad y confort en el hogar a bajo costo, y ahorro energético.

---

[1] Basado en [1].

[2] Basado en [2]

Por ello se considera factible interconectar dispositivos a través de la red eléctrica, utilizando la misma como un canal de comunicación y control.

El objetivo de este trabajo es diseñar e implementar un sistema PLC (Power Line Comunication / Comunicación sobre la línea eléctrica) basado en microcontrolador para la automatización del encendido y apagado de lámparas a bajo costo.

En el capítulo uno se aborda la investigación del funcionamiento de un sistema PLC (Power Line Comunication /Comunicación sobre la línea eléctrica).

En el capítulo dos se explica el diseño y funcionamiento de un circuito transmisor mediante la línea eléctrica.

En el tercer capítulo se detallan los algoritmos y la programación del envío de datos por la línea eléctrica utilizando un microcontrolador.

Finalmente, se integró el sistema de control basado en microcontrolador empleando la red eléctrica, los resultados y conclusiones se describen en los dos últimos capítulos.

Contenido

Indice de figuras

Indice de tablas

Símbolos y abreviaturas

µ	Indica un factor de 10^{-6}.
ASM	Lenguaje ensamblador
CA	Corriente alterna
CMOS	complementary metal-oxide-semiconductor
CPU	central processing unit / unidad central de procesamiento
DIP	Dual in-line package /
E/S	Entrada /salida
GND	Ground /tierra
HEX	Hexadecimal
Hz	Hertz
K	Indica un factor de 10^{3}
M	Indica un factor de 10^{6}
m	Indica un factor de 10^{-3}.
PDIP	Plastic Dual in-line package /
PIC	Peripheral Interface Controler / Controlador de interfaz periférico.
RISC	Reduced Instruction Set Computer /
s	Segundo
SOIC	small-outline integrated circuit
TTL	Transistor Transistor Logic / Lógica transistor a transistor
Vcc	Fuente de voltaje en corriente continua
Ω	Ohmio es la unidad de resistencia eléctrica.

CAPITULO 1

CONCEPTOS BASICOS

En este capítulo se describen los conceptos que se utilizan en el desarrollo de esta tesis. Inicia explicando el término domótica y los diferentes sistemas que existen. También se explica el protocolo X-10 y su funcionamiento. Los microcontroladores, es otro tema que se aborda, así como, la importancia de elegir entre la gran variedad que existe.[3]

1.1. DOMOTICA

La palabra domótica proviene de la unión de la palabra “domo” y el sufijo “tica”. La palabra “domo” etimológicamente proviene del latín domus que significa casa, y el sufijo “tica” proviene de la palabra automática, otros autores les dan significados como informática o robótica; incluso hay quienes dividen el sufijo “tica” en dos partes:”tic” (tecnologías de información) y “a” (automatización).

La Asociación Española de Domótica (CEDOM) define la domótica como “la incorporación al equipamiento de nuestras viviendas y edificios de una sencilla tecnología que permita gestionar de forma energéticamente eficiente, segura y confortable para el usuario los distintos aparatos e instalaciones domésticas tradicionales que conforman una vivienda”

El origen de la domótica se sitúa en Europa en los años 70 con el surgimiento del protocolo X-10, a partir de ahí, comenzó su evolución y comenzaron a surgir los distintos estándares e infraestructuras.

En la actualidad existen diversos sistemas de comunicación domóticos, los cuales se clasifican dependiendo del canal utilizado para dicha transmisión:

Sistemas tradicionales donde emisor y receptor están unidos físicamente: estas soluciones requieren infraestructura previa, debiendo ser previstos con anterioridad a la construcción del edificio, de forma que sea posible la introducción del cableado necesario.

[3] Capitulo basado en [3], [4], [5], [6], [7] y [8]

Sistemas por radio frecuencia: estos métodos presentan como ventaja el hecho de que no necesitan cableado. Sin embargo, tienen como inconveniente la limitación en el alcance, debido a su elevada susceptibilidad a las perturbaciones del medio, o simplemente causadas por la propia distribución de las paredes del edificio.

Sistemas basados en corrientes portadoras: permiten comunicar los diversos elementos utilizando las líneas eléctricas existentes como soporte de información.

1.2. PROTOCOLO X10

1.2.1. HISTORIA DEL PROTOCOLO X10

X-10 es uno de los protocolos más antiguos que se están usando en aplicaciones domóticas. Fue diseñado en Escocia entre los años 1976 y 1978, por la empresa PICO electronics y la empresa de sistemas de audio BSR, dentro de un conjunto de proyectos que denominaron X, de los cuales el que más éxito y repercusión tuvo fue el 10. El objetivo era transmitir datos por las líneas de baja tensión a bajo costo, aunque la velocidad fue muy baja (60 bps (bits por segundo) en EEUU y 50 bps (bits por segundo) en Europa). Fue introducido por primera vez en 1978 por Sears Home Control System y Radio Shack Plug´n Power System.

El formato de codificación X-10 es un estándar de transmisión por corrientes portadoras (Power Line Comunication) asegurando que todos los equipos que lo utilizan sean compatibles entre sí. El elemento básico y fundamental de la técnica de corrientes portadoras es el aprovechamiento doble de la instalación eléctrica ya existente como conductor de energía y de información. Con los componentes X-10 la red, además del suministro de corriente, se encarga también de la transmisión de señales de mando para los diversos aparatos eléctricos. Con ello se puede enviar señales de corriente

portadoras a cualquier punto de la instalación que se desee y, a su vez, se puede solicitar de dicho punto las informaciones pertinentes.

El sistema permite el accionamiento a distancia y control remoto de diversos receptores eléctricos, desde uno o desde varios puntos, y trabaja tanto en redes de corriente alterna monofásica como trifásica.

FUNCIONAMIENTO DEL PROTOCOLO X-10

La transmisión de señales se sincroniza con el punto de cruce en cero de la corriente eléctrica (de 50Hz a 60Hz). Se intenta transmitir lo más cerca del punto cero como sea posible, pero se acepta una variación de 200 µs desde el punto cero. Un “1” binario del mensaje se representa por un pulso de 120 KHz durante 1 ms, en el paso por cero de la señal de red, y el “0” binario del mensaje se representa por la ausencia de ese pulso de 120 KHz como se observa en la Fig.1.1.

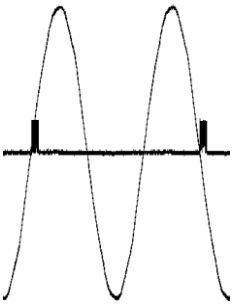

Figura 1.1. Envío de un “1” y un “0” binario

Cada pulso deberá ser enviado tres veces si se tratara de un sistema trifásico, esto con la intención de sincronizar el cruce por cero de las tres fases como se observa en la Fig. 1.2.

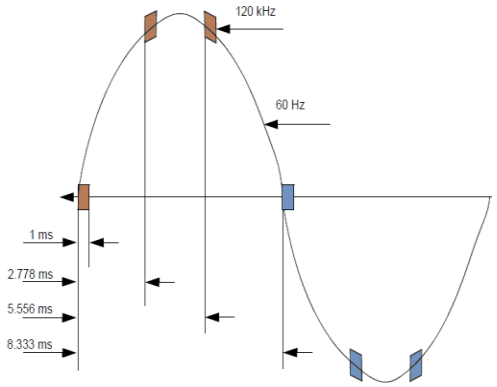

Figura 1.2. Envío de tres pulsos para un sistema trifásico.

TRAMA X-10

Un mensaje completo en X-10 está compuesto por el código de inicio 1110, seguido por un código de casa y un código de control. El código de control puede ser una dirección de unidad o un comando. En la Tabla 1.1 se muestra la dirección y el código de casa.

Tabla 1.1. Dirección y códigos de casa.

Dirección de casa	Códigos de casa			
	H1	**H2**	**H4**	**H8**
A	0	1	1	0
B	1	1	1	0
C	0	0	1	0
D	1	0	1	0
E	0	0	0	1
F	1	0	0	1
G	0	1	0	1
H	1	1	0	1
I	0	1	1	1
J	1	1	1	1
K	0	0	1	1
L	1	0	1	1
M	0	0	0	0
N	1	0	0	0
O	0	1	0	0
P	1	1	0	0

El Protocolo X-10 puede utilizar hasta 16 unidades por cada dirección de casa lo cual nos daría un total de 256 unidades. En la Tabla 1.2 se pueden apreciar las direcciones de unidad con sus respectivos códigos de control.

Tabla 1.2. Direcciones de unidad y códigos control.

Dirección de unidad	Códigos de control				
	D1	**D2**	**D4**	**D8**	**D16**
1	0	1	1	0	0
2	1	1	1	0	0
3	0	0	1	0	0
4	1	0	1	0	0
5	0	0	0	1	0
7	0	1	0	1	0
8	1	1	0	1	0
9	0	1	1	1	0
10	1	1	1	1	0
11	0	0	1	1	0
12	1	0	1	1	0
13	0	0	0	0	0
14	1	0	0	0	0
15	0	1	0	0	0
16	1	1	0	0	0

Las 16 unidades que maneja el protocolo X-10 se pueden controlar con los comandos que se muestran en la Tabla 1.3.

Tabla1.3. Comandos y códigos de control.

Comandos	**Códigos de control**				
	D1	D2	D3	D4	D5
Encender todas las unidades	0	0	0	0	1
Apagar todas las unidades	0	0	0	1	1
Encender	0	0	1	0	1
Apagar	0	0	1	1	1
Atenuar la intensidad	0	1	0	0	1
Aumentar la intensidad	0	1	0	1	1
Apagar todas las luces	0	1	1	0	1
Código extendido	0	1	1	1	1
Petición de saludo	1	0	0	0	1
Aceptación de saludo	1	0	0	1	1
Atenuación preestablecida	1	0	1	X	1
Datos extendidos	1	1	0	0	1
Estado = On	1	1	0	1	1
Estado =Off	1	1	1	0	1
Petición de Estado	1	1	1	1	1

Para construir una red de dispositivos X-10, a cada dispositivo se le asigna una identificación consistente de un código de 9 bits, donde los primeros 4 bits corresponden al “código de casa” y los otros 5 bits al “código de dispositivo”.

El “código de inicio” de la trama X-10 siempre es representada por el número binario 1110, y puesto que cada dígito binario ocupa medio ciclo de la corriente eléctrica se requieren 2 ciclos completos para enviar el “código de inicio”

Los bits del “código de casa” y del “código de control”, se transmiten en forma de complemento en los ciclos de la corriente alterna; es decir, para enviar el 1 se envía un 1 en el primer medio ciclo y un 0 en el segundo medio ciclo; para enviar el 0 se envía un 0 en el primer medio ciclo y un 1 en el segundo medio ciclo. El código de inicio es el único que no se envía en complemento, ver Tabla 1.4.

Tabla 1.4. Complemento del código X-10.

	Código de inicio	**Código de casa**				**Código de control**				
Código X-10	1110	**0**	**1**	**1**	**0**	**0**	**1**	**1**	**0**	**0**
Complemento	1110	**01**	**10**	**10**	**01**	**01**	**10**	**10**	**01**	**01**

La trama X-10 se divide en dos paquetes de datos, los cuales deben enviarse como mínimo dos veces para garantizar que los datos sean correctos. Antes de enviar el segundo paquete es necesario esperar tres ciclos de corriente alterna sin transmisión[4], correspondientes a seis cruces por cero.

La diferencia entre los dos paquetes es que en el primer paquete el código de control se refiere a una dirección de unidad y en el segundo bloque se refiere a un comando tal como se observa en la Fig. 1.3.

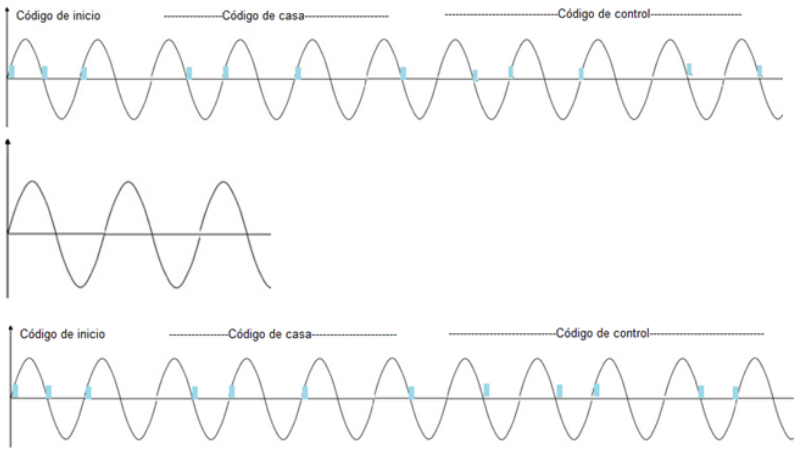

Figura 1.3. Envío de paquetes X-10.

[4]En el caso de comandos para aumentar o disminuir la iluminación, los bloques son enviados consecutivamente, sin los 3 ciclos entre ellos.

1.3. MICROCONTROLADORES

Un microcontrolador es un circuito integrado que ofrece las posibilidades de una computadora. En su interior se localiza un procesador, memoria, y varios periféricos. El secreto de los microcontroladores lo encontramos en su tamaño, su precio y su diversidad. Se emplea para controlar el funcionamiento de una tarea determinada y, debido a su reducido tamaño, suele ir incorporando en el propio dispositivo al que gobierna. Esta última característica es el que confiere la denominación “controlador embebido”. En su memoria solo reside un programa destinado a gobernar una aplicación determinada; sus líneas de entrada/salida soportan el conexionado de los sensores y actuadores del dispositivo a controlar, y todos los recursos complementarios disponibles tienen como única finalidad atender sus requerimientos. Una vez programado y configurado el microcontrolador solamente sirve para gobernar la tarea asignada.

La industria informática acapara gran parte de los microcontroladores que se fabrican, casi todos los periféricos de la computadora son regulados por el programa de un microcontrolador. Los electrodomésticos de línea blanca y línea marrón incorporan numerosos microcontroladores.

1.3.1. DIFERENCIA ENTRE MICROPROCESADOR Y MICROCONTROLADOR

El microprocesador es un circuito integrado que contiene la Unidad Central de Proceso, también llamada procesador, de una computadora. La CPU está formada por la Unidad de control, que interpreta las instrucciones, y el camino de datos (bus), que las ejecuta.

Los pines de un microprocesador comunican al exterior las líneas de los buses de direcciones, datos y control, para permitir conectarle con la memoria y los módulos de E/S y configurar una computadora implementada por varios circuitos integrados. Se dice que un microprocesador es un sistema abierto porque su configuración es variable de acuerdo con la aplicación a la que se destine, ver Fig. 1.4.

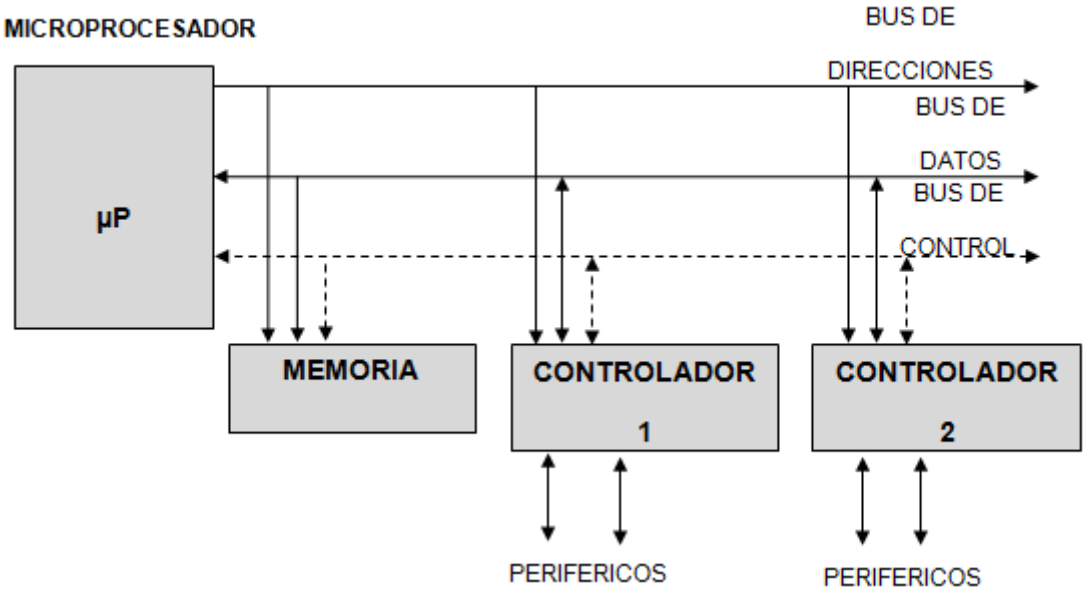

Figura 1.4. Estructura de un sistema abierto basado en microprocesador.

Un microcontrolador posee todos los componentes de una computadora, pero con características fijas que no pueden alterarse. Las partes principales son:

1. Procesador
2. Memoria no volátil para contener el programa
3. Memoria de lectura/escritura para guardar los datos
4. Líneas de E/S para los controladores periféricos:
    a. Comunicación paralelo
    b. Comunicación serie
    c. Diversas puertas de comunicación
5. Recursos auxiliares:
    a. Circuito de reloj
    b. Temporizadores
    c. Perro Guardián (Watch Dog)
    d. Conversores AD y DA
    e. Comparadores analógicos
    f. Protección ante fallos de alimentación
    g. Estado de reposo o bajo consumo

En la Fig. 1.5 se observa la estructura de un microcontrolador.

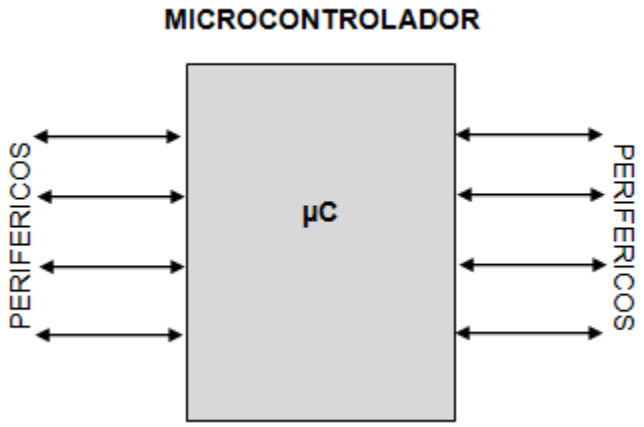

Figura 1.5. Estructura de un sistema cerrado basado en microcontrolador.

1.3.2.MICROCONTROLADORES PIC

Los PIC son una familia de microcontroladores tipo RISC fabricados por Microchip Technology Inc., originalmente desarrollado por la división de microelectrónica de General Instrument. Los microcontroladores PIC se caracterizan por que la arquitectura del procesador sigue el modelo Harvard. En esta arquitectura, la CPU se conecta de forma independiente y con buses distintos con la memoria de instrucciones y con la de datos. Es posible realizar operaciones de acceso (lectura o escritura) simultáneamente en ambas memorias, ésta es la estructura para los PIC's. La arquitectura Harvard permite a la CPU acceder simultáneamente a las dos memorias, como se observa en laFig.1.6.

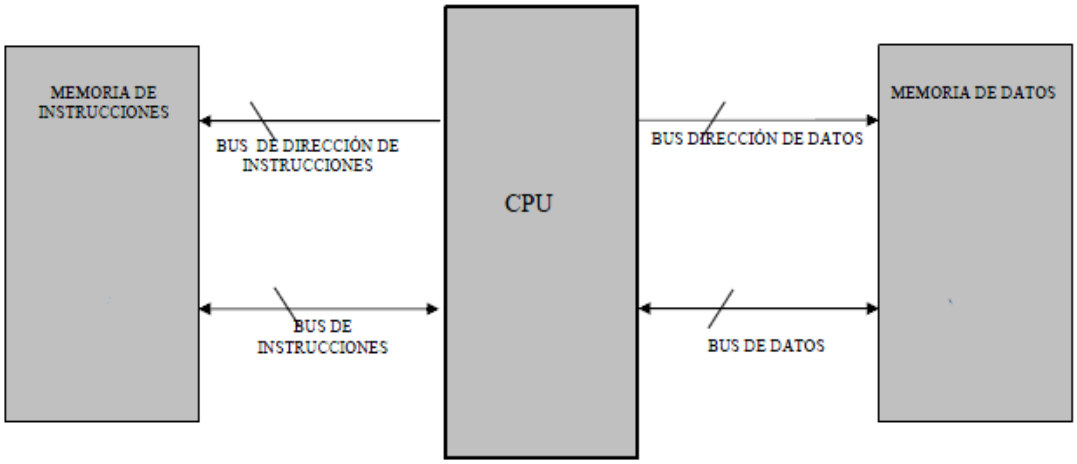

Figura 1.6. Arquitectura Harvard.

Genéricamente los microcontroladores se clasifican según el tamaño de los datos que maneja el repertorio de instrucciones y existen 4 grandes grupos: de 4, de 8, de 16 y de 32 bits. Los microcontroladores PIC de 8 bits se clasifican en tres grandes gamas: Base, Media y Mejorada, con un total de unos 300 modelos diferentes que contienen distintas capacidades de memoria, periféricos y distintos tipos de encapsulados. La diversidad de los modelos de PIC tiene una finalidad: poder seleccionar el más adecuado para cada aplicación. En la Fig. 1.7 se puede observar la distribución de los microcontroladores PIC.

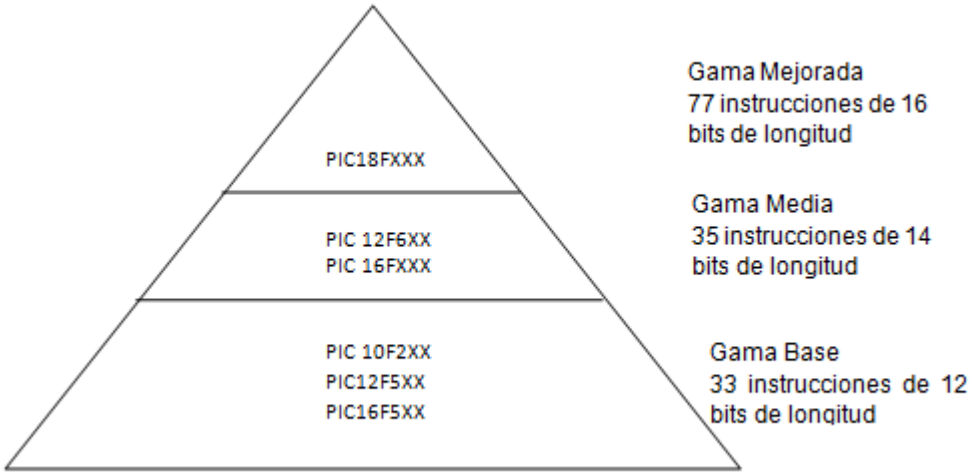

Figura 1.7. Gamas de la familia PIC.

MICROCONTROLADOR PIC 16F84A

El Microcontrolador PIC16F84A pertenece a la familia de la gama media de Microchip y dentro de ella es uno de los más pequeños; además es el que dispone de menos recursos. Está fabricado en tecnología CMOS, consume baja potencia, y es completamente estático (si el reloj se detiene, los datos de la memoria no se pierden), posee memoria FLASH, esto hace que tenga menor consumo de energía y mayor capacidad de almacenamiento.

El encapsulado más común para este microcontrolador es el DIP de 18 pines, y utiliza un reloj de 4 MHz (cristal de cuarzo). Sin embargo, hay otros tipos de encapsulado, por ejemplo, el encapsulado tipo montaje superficial es mucho más pequeño. En la Fig.1.8 se puede observar los pines con sus respectivos nombres del PIC16F84A.

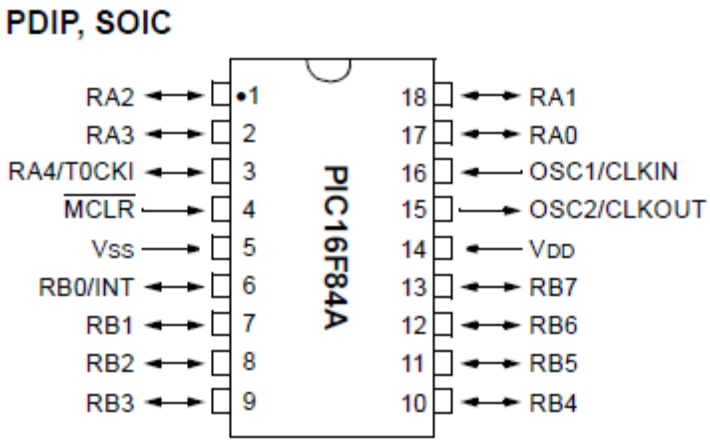

Figura 1.8.-Descripcion de los pines del PIC 16F84A.

PUERTO A

El puerto A corresponde a 5 líneas bidireccionales de E/S (definidas por programación) ubicadas en los pines 1, 2, 3,17 y 18. Es capaz de entregar niveles TTL cuando la alimentación aplicada en VDD es de 5V ± 5%.El pin 3 también puede funcionar como contador de sucesos o como temporizador.

PUERTO B

El puerto B corresponde a ocho líneas bidireccionales de E/S (definidas por programación) ubicadas en los pines6, 7, 8, 9, 10, 11, 12 y 13. Pueden manejar niveles TTL cuando la tensión de alimentación aplicada en VDD es de 5V ± 5%.RB0 puede programarse además como entrada de interrupciones externas INT. Los pines RB4 a RB7 pueden programarse para responder a interrupciones por cambio de estado. Las patas RB6 y RB7 se corresponden con las líneas de entrada de reloj y entrada de datos respectivamente, cuando está en modo programación del integrado.

PINES 4, 5, 14, 15 y 16

El pin 4permite que al encender el sistema el microcontrolador quede en estado de reset por un tiempo mientras se estabilizan todas las señales del circuito ose puede implementar un circuito para reiniciar el programa del PIC.

Los pines 5 y 14 corresponden a tierra y alimentación. La tensión de alimentación de un PIC está comprendida entre 2V y 6V aunque se recomienda no sobrepasar los 5.5V.

Los pines 15 y 16 corresponden a los pines de la entrada externa de reloj y salida de oscilador a cristal respectivamente.

EL OSCILADOR EXTERNO

Es un circuito externo que le indica al microcontrolador la velocidad a la que debe trabajar. Este circuito, que se conoce como oscilador o reloj, es muy simple pero de vital importancia para el buen funcionamiento del sistema. El P1C16C84/F84 puede utilizar cuatro tipos de reloj diferentes. Estos tipos son:

- ✓ **RC.** Oscilador con resistencia y condensador.
- ✓ **XT.** Cristal.
- ✓ **HS.** Cristal de alta velocidad.
- ✓ **LP.** Cristal para baja frecuencia y bajo consumo de potencia.

En el momento de programar el microcontrolador se debe especificar que tipo de oscilador se usa.

ORGANIZACION DE MEMORIA DEL PIC 16F84A

El PIC 16F84A se encuentra dividido en dos bloques de memoria estos son: la memoria de datos y la memoria de programa. Cada uno de los bloques posee su propio bus por lo que pueden ser accesados durante el mismo ciclo de oscilador.

ORGANIZACION DE LA MEMORIA DE PROGRAMA

La memoria de programa se divide en páginas de 2,048 posiciones. El PIC16F84A sólo tiene implementadas 1K posiciones es decir de 0000h a 03FFh y el resto no está implementado, ver Fig. 1.9. Cuando ocurre un Reset, el contador de programa (PC) apunta a la dirección 0000h, y el microcontrolador se inicia nuevamente. Por esta razón, en la primera dirección del programa se debe escribir todo lo relacionado con la iniciación del mismo (*por ejemplo, la configuración de los puertos...*)

Si ocurre una interrupción el contador de programa apunta a la dirección 0004h, entonces ahí se escribirá la programación necesaria para tender dicha interrupción. Algo que se debe tener en cuenta es la pila o stack, que consta de 8 posiciones o niveles.

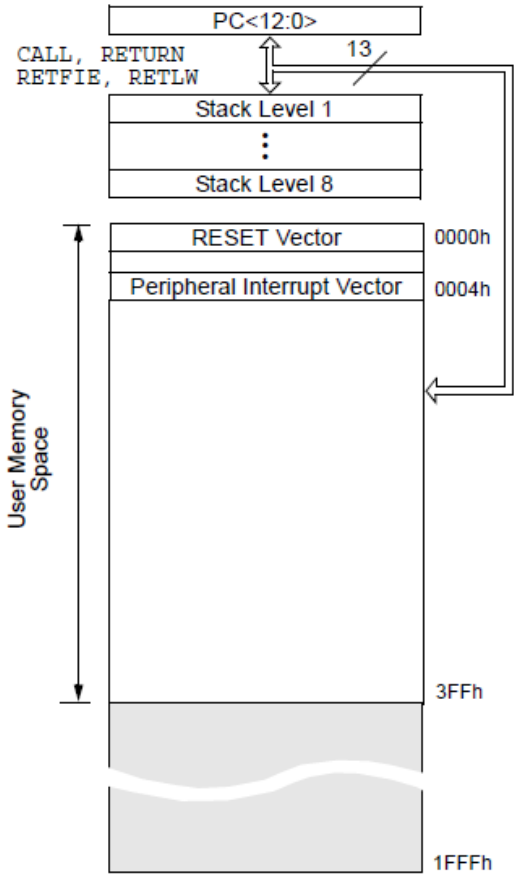

Figura 1.9. Organización de la memoria del programa.

ORGANIZACIÓN DE LA MEMORIA DE DATOS

La Memoria de datos se puede desglosar en dos tipos de registros: los de propósitos específicos (SFR) y los de propósito general (GPR).La RAM del PIC16F84A se halla dividida en dos bancos (banco 0 y banco 1) de 128 bytes cada uno.

Los registros de propósito específicos se localizan en las primeras 12 posiciones, algunos registros se hallan repetidos en la misma dirección de los dos bancos para simplificar su acceso; los registros de propósito especifico son los encargados del control del procesador y sus recursos, ver Fig. 1.10.

Los 36 registros restantes de cada banco se destinan a registros de propósito general y en realidad solo son operativos los 36 del banco 0 por que los del banco 1 se mapean sobre el banco 0, es decir cuando se apunta a un registro general del banco 1, se accede al mismo del banco 0.

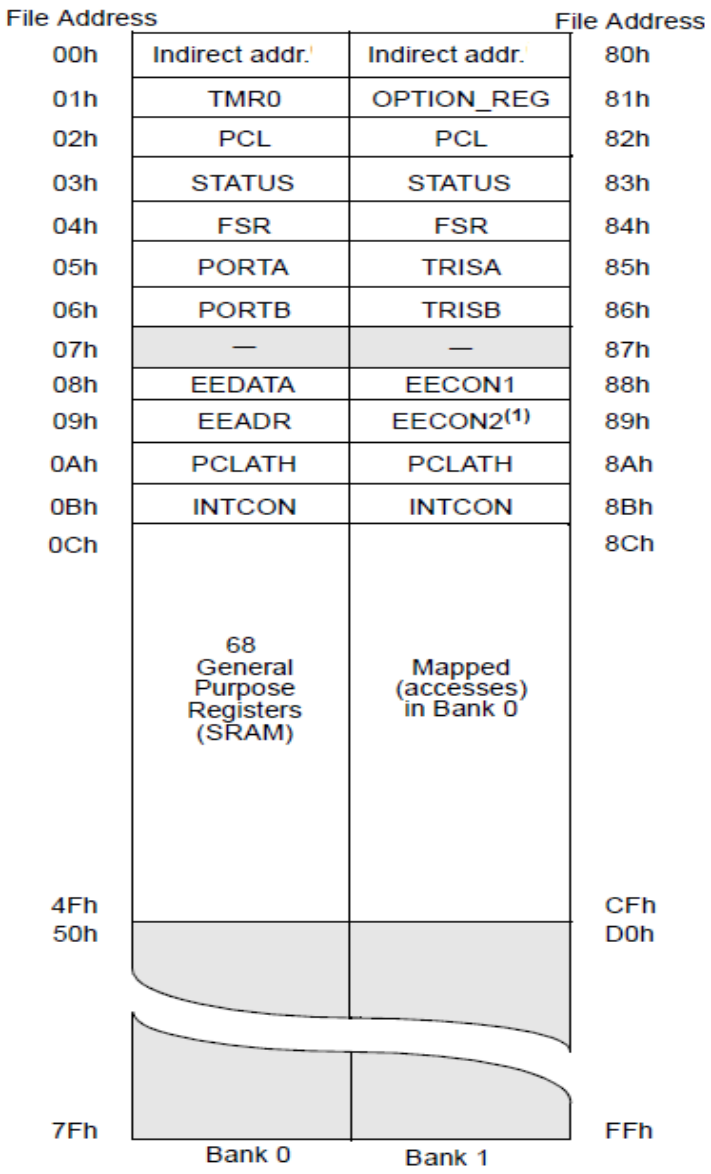

Figura 1.10. Organización de la memoria de datos.

La Fig.1.11 presenta la estructura de los registros.

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on RESET	Details on page
Bank 0											
00h	INDF	Uses contents of FSR to address Data Memory (not a physical register)								---- ----	11
01h	TMR0	8-bit Real-Time Clock/Counter								xxxx xxxx	20
02h	PCL	Low Order 8 bits of the Program Counter (PC)								0000 0000	11
03h	STATUS(2)	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	8
04h	FSR	Indirect Data Memory Address Pointer 0								xxxx xxxx	11
05h	PORTA(4)	—	—	—	RA4/T0CKI	RA3	RA2	RA1	RA0	---x xxxx	16
06h	PORTB(5)	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0/INT	xxxx xxxx	18
07h	—	Unimplemented location, read as '0'								—	—
08h	EEDATA	EEPROM Data Register								xxxx xxxx	13,14
09h	EEADR	EEPROM Address Register								xxxx xxxx	13,14
0Ah	PCLATH	—	—	—	Write Buffer for upper 5 bits of the PC(1)					---0 0000	11
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	10
Bank 1											
80h	INDF	Uses Contents of FSR to address Data Memory (not a physical register)								---- ----	11
81h	OPTION_REG	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	9
82h	PCL	Low order 8 bits of Program Counter (PC)								0000 0000	11
83h	STATUS (2)	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	8
84h	FSR	Indirect data memory address pointer 0								xxxx xxxx	11
85h	TRISA	—	—	—	PORTA Data Direction Register					---1 1111	16
86h	TRISB	PORTB Data Direction Register								1111 1111	18
87h	—	Unimplemented location, read as '0'								—	—
88h	EECON1	—	—	—	EEIF	WRERR	WREN	WR	RD	---0 x000	13
89h	EECON2	EEPROM Control Register 2 (not a physical register)								---- ----	14
0Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of the PC(1)					---0 0000	11
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	10

Figura 1.11. Estructura de los registros SFR de los dos bancos de la memoria de datos.

CAPITULO2

DISEÑO DEL CIRCUITO TRANSMISOR X-10

En este capítulo se explica cómo construir un transmisor del protocolo X-10. Se desarrolla el diseño del circuito y se explica el funcionamiento de cada una de sus partes. Por ejemplo, la importancia del detector de cruce por cero para el envío de datos, la frecuencia de 120 KHz, entre otros.

2.1. DISEÑO DEL TRANSMISOR X-10

Los microcontroladores PIC pueden ser utilizados para la automatización de un hogar; el protocolo X-10 en combinación con algún PIC logra como resultado diversas aplicaciones domóticas. En este caso se utilizará para el control de luminarias, el PIC a utilizar es el 16F84A.

El hardware para la funcionalidad del circuito transmisor está dividido en los bloques que muestra la Fig.2.1.

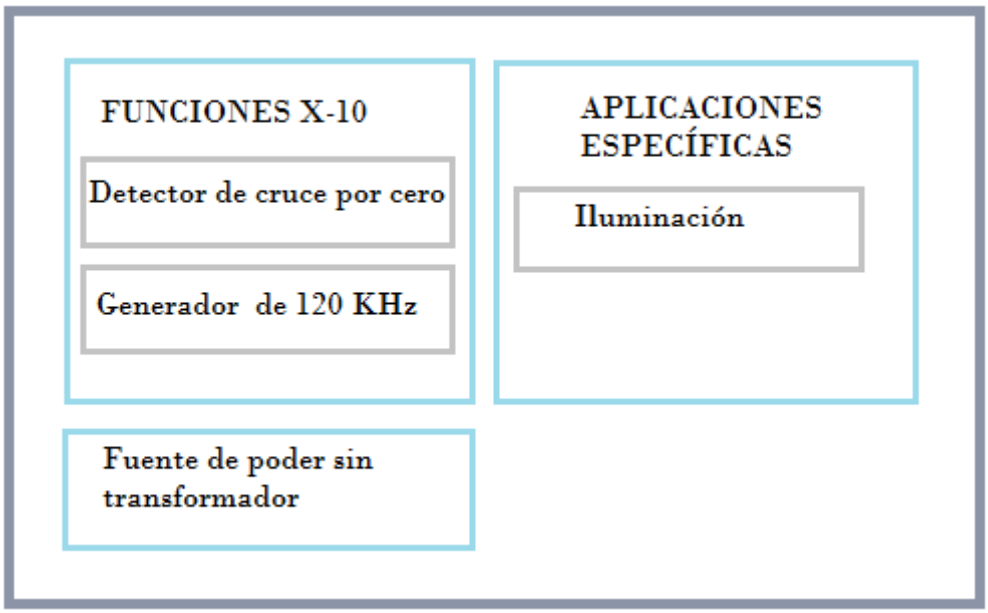

Figura 2.1. Diagrama a bloques del circuito transmisor X-10.

2.1.1.DETECTOR DE CRUCE POR CERO

En el protocolo X-10 la información que se transmite es sincronizada con los cruces por cero de la línea de corriente alterna. Un detector de cruce por cero es fácil de crear utilizando el pin RB0 el cual es capaz de detectar alguna interrupción externa al PIC, en este caso se logra utilizando los flancos de subida o de bajada[5].

El voltaje de la línea eléctrica es de 120 V, si aplicamos ese voltaje directamente a un pin del PIC este saldría severamente dañado. El PIC tiene un circuito de protección en las terminales de entrada/salida, este circuito está formado por diodos que protegen al PIC de cualquier valor mayor que VDD y menor que VSS como se puede observar en la Fig. 2.2. Ellos pueden tomar varios miliamperios de corriente sin ningún daño en el chip.

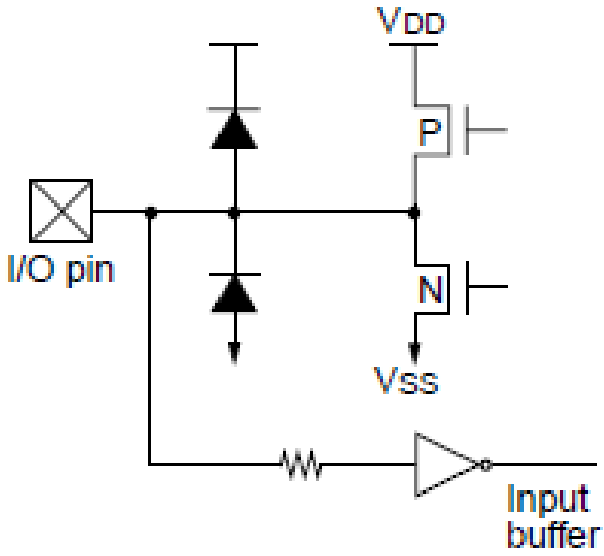

Figura 2.2. Circuito de protección.

Los altos voltajes pueden aplicarse a las entradas del PIC siempre y cuando la corriente sea limitada, esto se puede lograr conectando una resistencia en serie a la entrada del PIC. Para calcular la corriente utilizamos la ley de ohm:

[5] Este apartado se basa en [9]

$$i = \frac{v}{r} \quad i = \frac{120}{5 * 10^6} \quad i = 24\mu A$$

La corriente máxima que un PIC puede soportar cuando está recibiendo sobre voltaje en una de sus terminales es de $\pm 500\mu A$ y el resultado de la ecuación anterior es de 24µA, por lo tantola resistencia elegida es de 5MΩ. En la Fig. 2.3 se observa el circuito detector de cruce por cero en el cual se puede apreciar cómo se conecta la resistencia de 5MΩ.

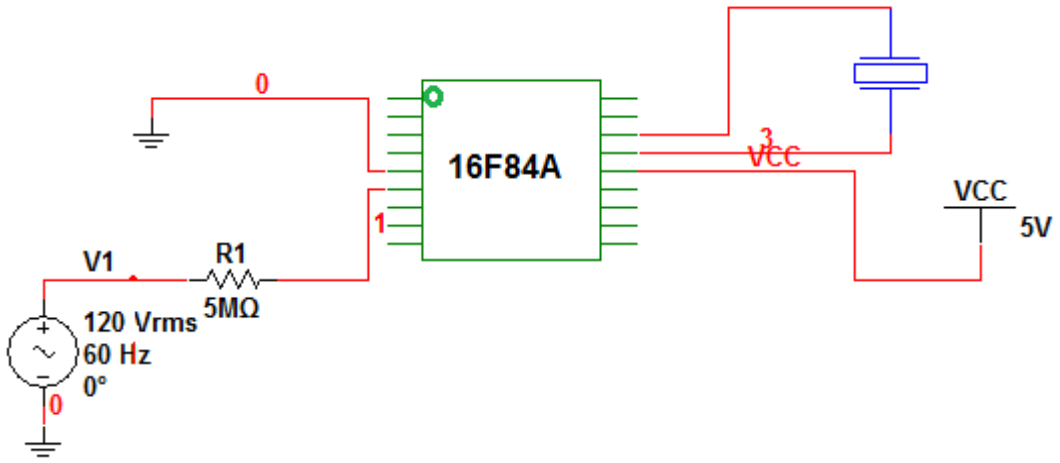

Figura 2.3. Circuito detector de cruce por cero.

2.1.2.GENERADOR DE 120KHZ

El protocolo X-10 envía pulsos de un milisegundo a una frecuencia de 120 KHz sobre la línea de 60 Hz. Se utilizó el PIC16F84A para generar los pulsos mientras que la frecuencia de 120 KHz se introdujo mediante un generador de señales[6].

Para que el pulso de 1ms se envíe a una frecuencia de 120 KHz se utiliza una compuerta lógica AND formada por dos compuertas lógicas NAND de tecnología CMOS;la salida de la compuerta se conecta a un filtro pasa altas formado por una resistencia y un capacitor como se observa en la Fig. 2.4.

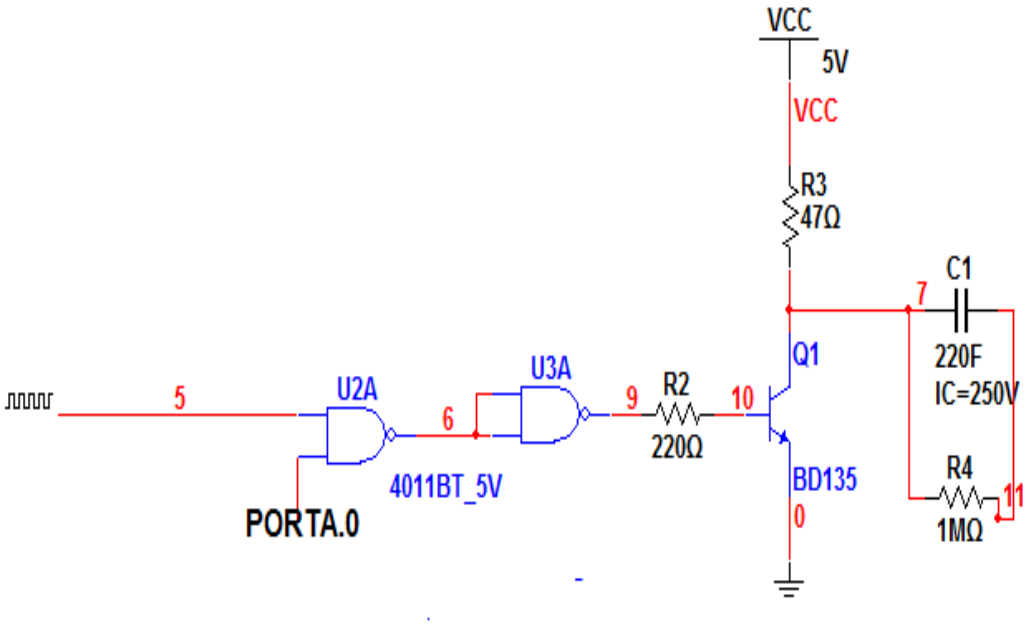

Figura 2.4. Circuito acoplador de 120KHz.

[6] Este apartado se basa en [10]

2.1.3.CIRCUITO TRANSMISOR X-10[7]

En la Fig. 2.5 se puede observar el diseño completo del transmisor X-10 y en la Fig. 2.6 se observa el prototipo en el momento en que se mide el voltaje de la línea eléctrica.

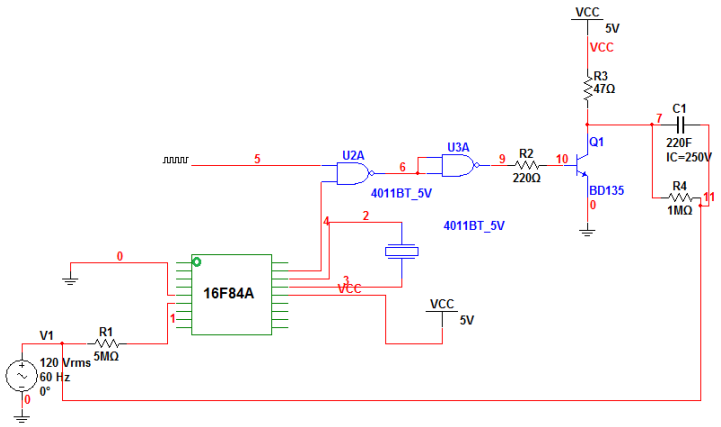

Figura 2.5. Diseño del transmisor X-10.

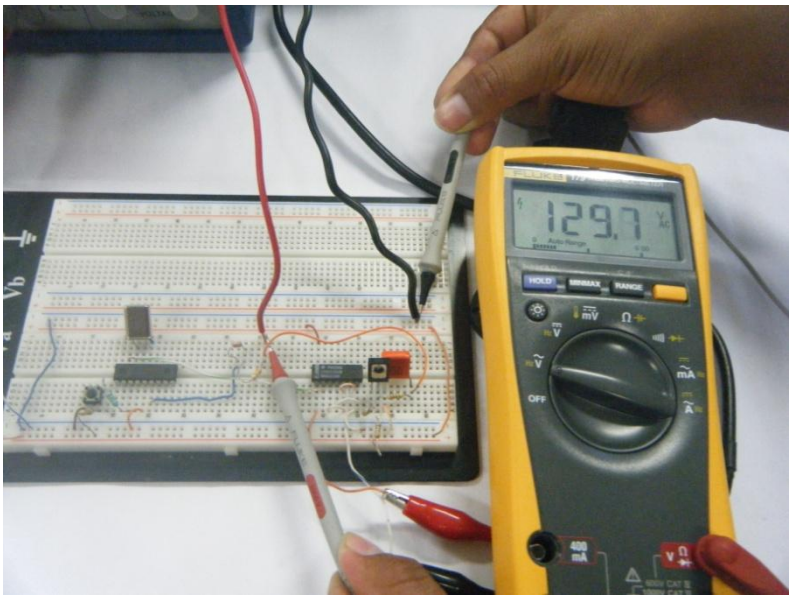

Figura 2.6. Circuito transmisor X-10.

[7] Este apartado se basa en [10]

CAPITULO 3

PROGRAMACION X-10

En este capítulo se desarrolla la programación del microcontrolador. Es importante que la programación del PIC funcione correctamente para que se puedan transmitir los datos en la línea eléctrica.

Primeramente, se explica el diagrama de flujo principal para entender el funcionamiento del programa, después cada subrutina del diagrama de flujo principal es explicada por separado. Finalmente, se lleva a cabo la programación en el lenguaje ensamblador.

3.1. DIAGRAMAS DE FLUJO

El diagrama de flujo representa gráficamente el algoritmo que se ha programado. En este capítulo se explican los diagramas de flujo y la programación utilizada.

3.1.1.DIAGRAMA DE FLUJO PRINCIPAL

Las figuras de este apartado muestran el diagrama de flujo principal del programa que se ha desarrollado, está dividido por cuestión de espacio. Al iniciar el programa lo primero es la configuración de puertos, es en este apartado donde se configura si un puerto es de entrada o de salida, a que posición de memoria pertenece, etc.

Primero se configuran los puertos, después se habilita la detección del flanco ascendente para detectar el primer cruce por cero, y seguidamente se configura el TIMER para que funcione como un contador a 1ms tal como lo ilustra la Fig. 3.1.

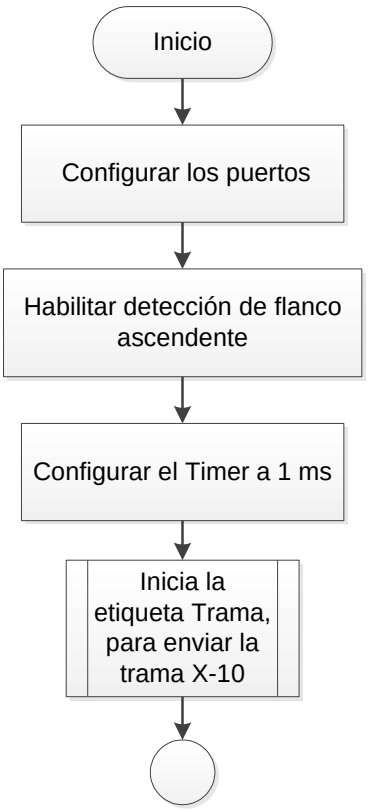

Figura 3.1. Configuraciones en el diagrama de flujo principal.

A continuación se envía el primer paquete de datos que está integrado por el código de inicio, el código de casa y el código de unidad, el cual se envía dos veces, como se observa en la Fig. 3.2.

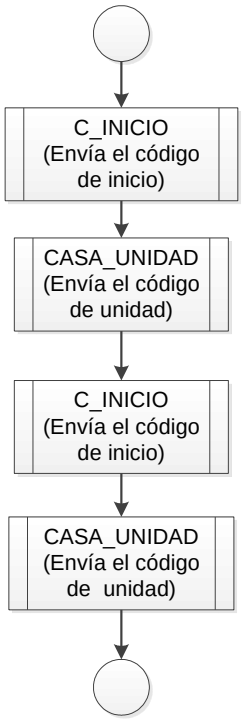

Figura 3.2. Envío del primer paquete X-10 en el diagrama de flujo principal.

Después del primer paquete de datos transcurren seis cruces por cero para volver a transmitir el segundo paquete, ver Fig.3.3.

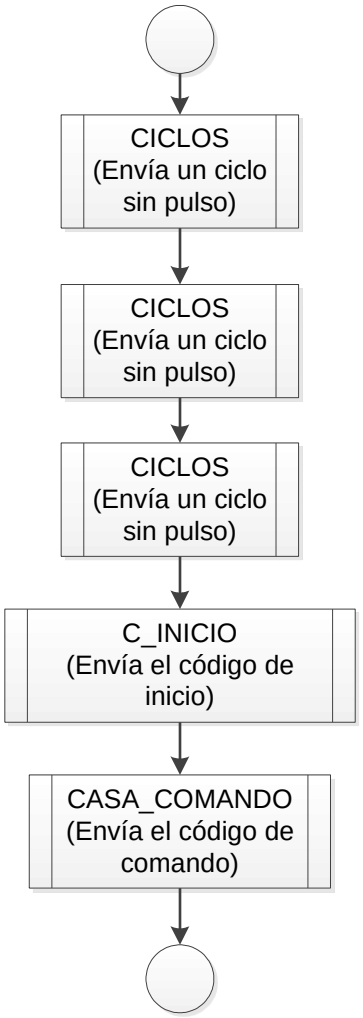

Figura 3.3. Ciclos de espera en el diagrama de flujo principal.

El segundo paquete está conformado por el código de inicio, el código de casa y el código de comando. Al terminar de enviar el segundo paquete transcurren cuatro ciclos de espera y finaliza el programa como lo ilustra la Fig.3.4.

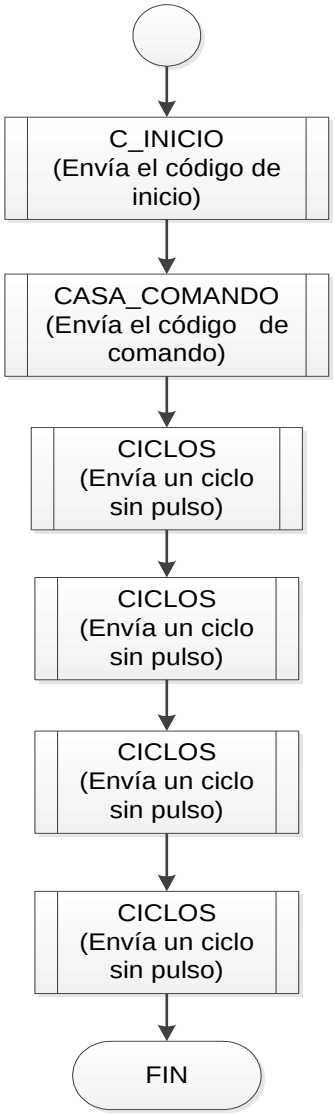

Figura 3.4. Envío del segundo paquete X-10 en el diagrama de flujo principal.

3.1.2. SUBRUTINAS

En el diagrama de flujo anterior se puede observar que el envío de la trama X-10 comienza en la etiqueta Trama, se realizan llamadas a diferentes subrutinas las cuales se explicarán detalladamente en este apartado.

SUBRUTINA C_INICIO

La función principal de la subrutina C_INICIO es enviar el código de Inicio 1110, ver Fig. 3.5.

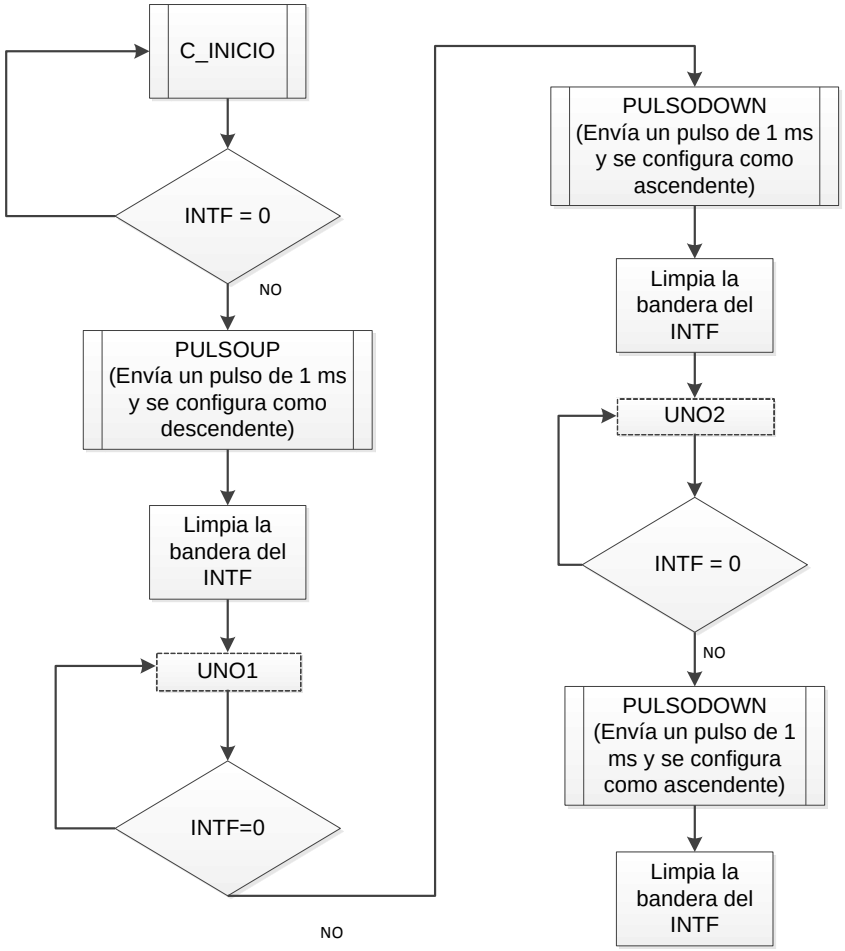

Figura 3.5. Subrutina C_INICIO.

Al llamar a la subrutina C_INICIO lo primero que se realiza es verificar la bandera INTF; la bandera INTF se localiza en el bit uno del registro ITCON. Si la bandera es igual a cero el programa regresa a la etiqueta C_INICIO de lo contrario envía un pulso con duración de 1ms y configura el PIC para detectar el flanco descendente; seguidamente limpia la bandera INTF para que se pueda detectar otro cruce por cero. Los siguientes pasos se repiten para enviar los pulsos faltantes, la configuración de flancos se van alternando.

SUBRUTINAS PULSOUP Y PULSODOWN

Las subrutinas PULSOUP y PULSODOWN tienen como objetivo enviar un pulso de 1 ms, llamando a dos subrutinas mas, PULSE, CONFIGDOWN y CONFIGUP respectivamente, ver Fig. 3.6.

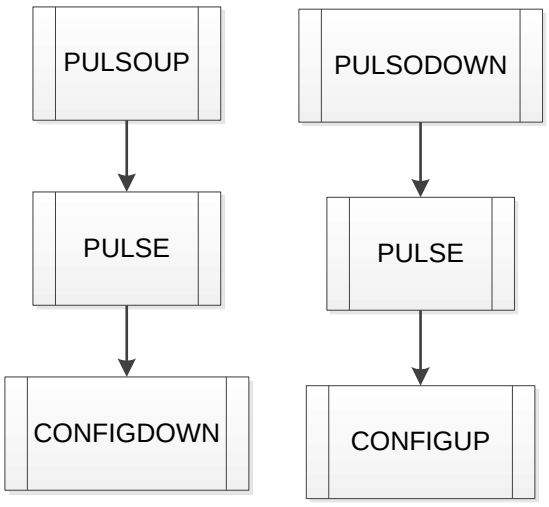

Figura 3.6. Subrutinas PULSOUP y PULSODOWN.

SUBRUTINA PULSE

La subrutina PULSE, prende el bit0 del puerto A, seguidamente llama a la subrutina PAUSA, después apaga el puerto A0. Como resultado se obtiene un pulso de 1 ms como se observa en la Fig. 3.7.

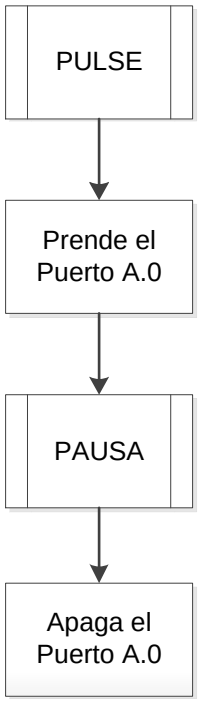

Figura 3.7. Subrutina PULSE.

SUBRUTINA PAUSA

Es en esta subrutina donde se utiliza el Timer para que funcione como temporizador. El primer paso es reiniciar el Timer, seguidamente inicia la etiqueta REVISAR y comienza una operación para detectar cuando el temporizador ha llegado al límite de 1ms, ver Fig. 3.8

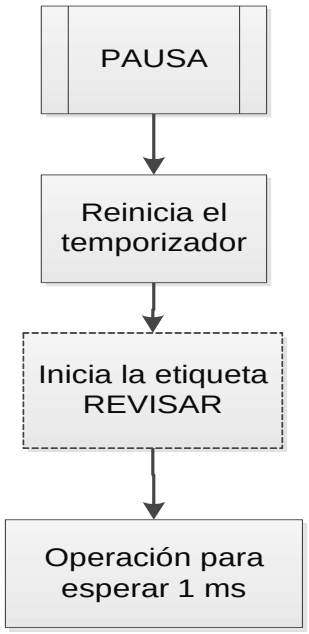

Figura 3.8. Subrutina PAUSA.

EL REGISTRO OPTION

La misión principal de este registro es controlar TMR0 y el divisor de frecuencia. Ocupa la posición 81H de la memoria de datos que equivale a la dirección 1 del banco 1. EL bit TOCS (Timer0 Clock Edge Select) selecciona la procedencia de los impulsos de reloj, que pueden ser del oscilador interno (Fosc/4) o los que se aplican desde el exterior del pin TOCKI.

El bit PSA del registro OPTION asigna el divisor de frecuencia al Timer (PSA=0) o al WDT (PSA=1). Los 3 bits de menos peso de OPTION seleccionan el rango por el que divide, el divisor de frecuencia, los impulsos que se le aplican en su entrada. El bit 6INTDEG sirve para determinar el flanco activo que provocará una interrupción externa al aplicarse ala RB0/INT. Un 1 si es ascendente y un 0 si es descendente. En la Fig. 3.9 se resumen los valores da cada bit.

RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit 7							bit 0

bit 7 **$\overline{\text{RBPU}}$:** PORTB Pull-up Enable bit
1 = PORTB pull-ups are disabled
0 = PORTB pull-ups are enabled by individual port latch values

bit 6 **INTEDG**: Interrupt Edge Select bit
1 = Interrupt on rising edge of RB0/INT pin
0 = Interrupt on falling edge of RB0/INT pin

bit 5 **T0CS**: TMR0 Clock Source Select bit
1 = Transition on RA4/T0CKI pin
0 = Internal instruction cycle clock (CLKOUT)

bit 4 **T0SE**: TMR0 Source Edge Select bit
1 = Increment on high-to-low transition on RA4/T0CKI pin
0 = Increment on low-to-high transition on RA4/T0CKI pin

bit 3 **PSA**: Prescaler Assignment bit
1 = Prescaler is assigned to the WDT
0 = Prescaler is assigned to the Timer0 module

bit 2-0 **PS2:PS0**: Prescaler Rate Select bits

Bit Value	TMR0 Rate	WDT Rate
000	1 : 2	1 : 1
001	1 : 4	1 : 2
010	1 : 8	1 : 4
011	1 : 16	1 : 8
100	1 : 32	1 : 16
101	1 : 64	1 : 32
110	1 : 128	1 : 64
111	1 : 256	1 : 128

Figura 3.9. Registro OPTION.

Para calcular los tiempos a controlar con TMR0 se utilizan las siguientes fórmulas prácticas.

$$\boldsymbol{Temporización = 4 * TOSC * Valor\ cargado\ en\ el\ TMR0 * Rango\ del\ divisor}$$

$$\boldsymbol{Valor\ a\ cargar\ en\ TMR0 = \frac{Temporización}{4} * TOSC * Rango\ del\ divisor}$$

Para calcular el valor del TMR0 para 1ms se utilizan los datos de la tabla 3.1.

Tabla 3.1. Valores asignados al registro OPTION.

RBPU	INTDEG	TOCS	TOSE	PSA	PSA2	PSA1	PSA0
1	**1**	**0**	**1**	**0**	**0**	**0**	**1**

Los tres bits de menos peso corresponden aun divisor de frecuencia 1:4, entonces el rango del divisor es 4 y el valor de TOSC se calcula con un oscilador de 4MHz. Sustituyendo en la fórmula queda de la siguiente manera:

$$\boldsymbol{valor\ a\ cargaren\ TMR0 = \frac{0.001}{4 * \frac{1}{4000000} * 4}}$$

$$\boldsymbol{Valor\ a\ cargaren\ TMR0 = 250}$$

El valor a cargar en TMR0 debe ser menor a 256, como el resultado cumple con el requisito, entonces se convierte el número 250 decimal a su valor en hexadecimal lo cual equivale a FA.

El valor binario 11010001 corresponde a los bits del registro OPTION, serán asignados al momento de programar, una vez completada esta acción, se puede leer el valor que contiene el TMR0 sin detener su contaje; para realizarlo se le resta el valor hexadecimal FA.

SUBRUTINAS CONFIGDOWN y CONFIGUP

EL protocolo X-10 se sincroniza con el cruce por cero de la línea eléctrica; en un ciclo de la onda senoidal existen dos cruces por cero, los cuales son detectados cuando la senoidal pasa de negativo a positivo (flanco ascendente) y de positivo a negativo (flanco descendente). La Fig. 3.10 ilustra las subrutinas CONFIGDOWN y CONFIGUP, las cuales configuran la detección de dichos flancos.

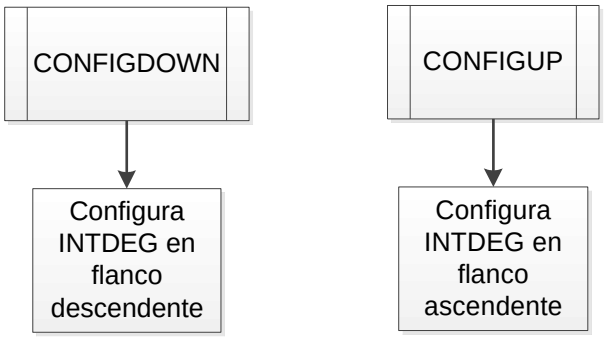

Figura 3.10. Subrutinas CONFIGDOWN y CONFIGUP.

EL REGISTRO INTCON

La mayor parte de los señalizadores y bits de permiso de las fuentes de interrupción en los PIC 16X8X están implementados sobre los bits del registro INTCON, que ocupa la dirección 0B H del banco cero , hallándose duplicado en el banco uno.

Al configurar este registro se asignan todos los bits en 0, así se deshabilita el permiso global de interrupciones (GIE) pero permite que las banderas se activen. La bandera que se necesita es la que ocupa el bit 1(INTF), esta al ponerse en uno indica que el pin RB0/INT se ha activado. Al activarse el bit INTF significa que el cruce por cero de la onda senoidal ha sucedido e inmediatamente

se enviara un pulso con la subrutina correspondiente dependiendo si es un flanco ascendente o descendente. La Fig. 3.11 resume los bits del registro INTCON.

GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF
bit 7							bit 0

bit 7 **GIE:** Global Interrupt Enable bit
1 = Enables all unmasked interrupts
0 = Disables all interrupts

bit 6 **EEIE**: EE Write Complete Interrupt Enable bit
1 = Enables the EE Write Complete interrupts
0 = Disables the EE Write Complete interrupt

bit 5 **T0IE**: TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt

bit 4 **INTE**: RB0/INT External Interrupt Enable bit
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt

bit 3 **RBIE**: RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **T0IF**: TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INTF**: RB0/INT External Interrupt Flag bit
1 = The RB0/INT external interrupt occurred (must be cleared in software)
0 = The RB0/INT external interrupt did not occur

bit 0 **RBIF**: RB Port Change Interrupt Flag bit
1 = At least one of the RB7:RB4 pins changed state (must be cleared in software)
0 = None of the RB7:RB4 pins have changed state

Figura 3.11. Registro INTCON.

SUBRUTINA CASA_UNIDAD

La subrutina CASA_UNIDAD se encarga de enviar los pulsos de 1ms necesarios para el código de la casa y la dirección de unidad. En este caso la casa es M y el número binario que le corresponde es 0000, la unidad es la número uno y el número binario que le corresponde es 01100, ver Fig. 3.12.

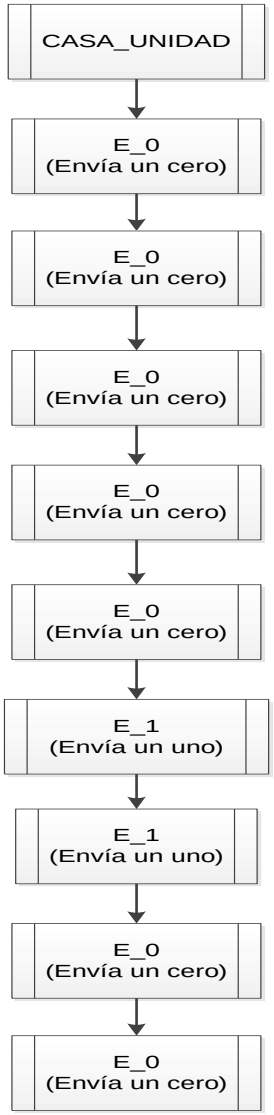

Figura 3.12. Subrutina CASA_UNIDAD.

SUBRUTINA E_0

La subrutina E_0 se ilustra en la Fig. 3.13, esta subrutina se encarga de enviar un 0 binario, como el protocolo X-10 utiliza complementos y solo envía datos en los cruces por cero en realidad se necesitan utilizar los dos flancos, el de subida que no envía datos y el de bajada que envía un pulso de 1ms. El resultado es el envío de 01.

EL funcionamiento consiste en detectar que la bandera INTF esté encendida, después configurar el flanco para que sea descendente, limpiar la bandera para que vuelva a detectar otro flanco y cuando la bandera se active entonces enviar el pulso de 1ms al igual que reconfigurar el flanco, por último se desactiva la bandera.

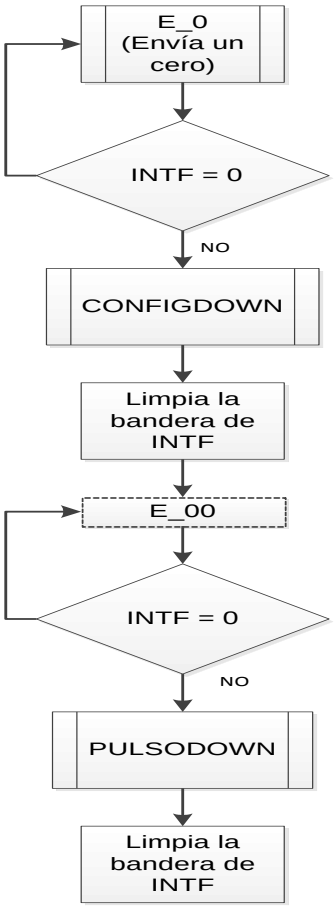

Figura 3.13 Subrutina E_0.

SUBRUTINA E_1

La subrutina E_1 funciona similar a la subrutina E_0, la diferencia es que ésta envía un 1 binario y su complemento es 10; es decir sólo en el flanco de subida envía el pulso, ver Fig. 3.14.

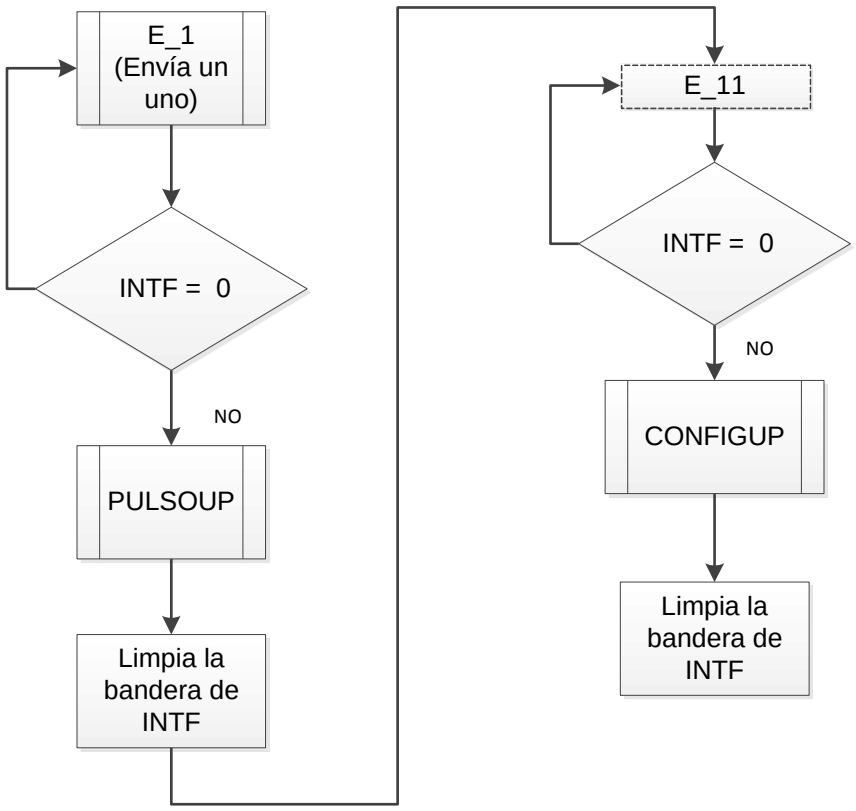

Figura 3.14. Subrutina E_1.

SUBRUTINA CICLOS

Esta subrutina es utilizada para que durante un ciclo de la senoidal no envíe datos, es muy útil después de enviar el primer paquete de datos por que el protocolo X-10 requiere seis cruces por cero antes de enviar el segundo paquete; También es utilizado antes de volver a enviar otra trama porque se necesitan por lo menos cuatro cruces por cero para volver a transmitir. La subrutina CICLOS se ilustra en la Fig. 3.15.

Funciona de la siguiente manera: detecta el flanco de subida y lo reconfigura, detecta el flanco de bajada y lo reconfigura, es importante limpiar la bandera después de cada reconfiguración de lo contrario no detectará el siguiente flanco.

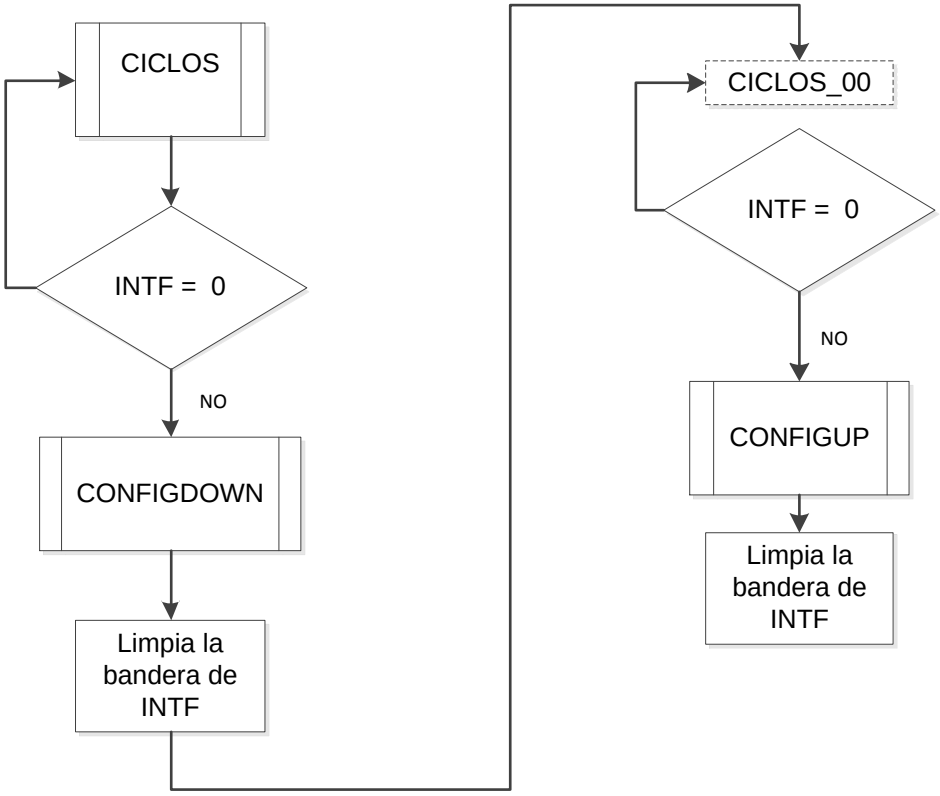

Figura 3.15. Subrutina CICLOS.

SUBRUTINA CASA_COMANDO

La subrutina CASA_COMANDO es similar a la subrutina CASA_UNIDAD, la diferencia es que se envía el comando en vez de la dirección de unidad. El programa envía el código de casa M que equivale al número binario 0000 y el comando encender que equivale al número binario 00101, como se observa en la Fig. 3.16.

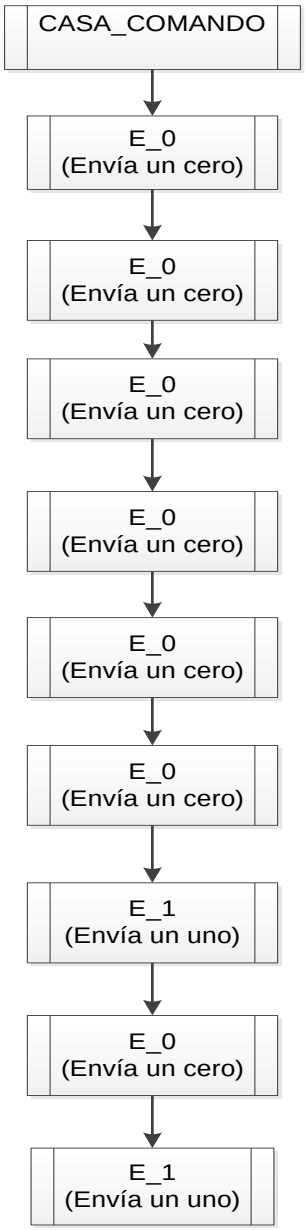

Figura 3.16. Subrutina CASA_COMANDO.

3.2. PROGRAMACION X-10

El lenguaje de programación utilizado fue Ensamblador, se eligió por que ofrece las siguientes ventajas:

- ✓ Flexibilidad en la configuración de los registros del PIC.
- ✓ Menor tiempo de respuesta del PIC.

Después de realizar la programación se utilizó la aplicación MPASM, con MPASM se puede producir un archivo HEX a partir de un archivo ASM. El archivo HEX es el que se graba dentro del PIC para que funcione. En la Fig. 3.17 se observa la interfaz de la aplicación MPASM.

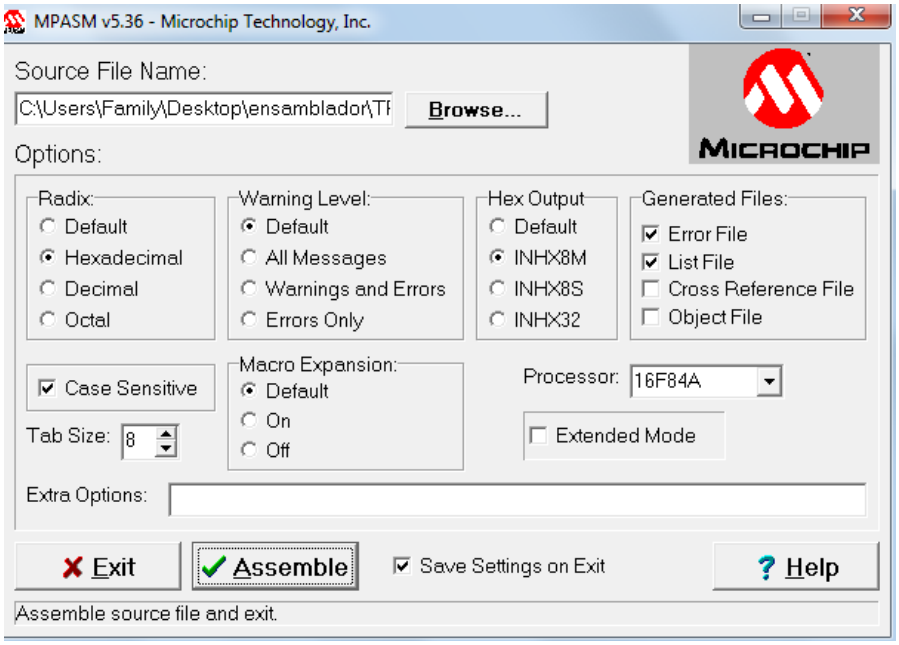

Figura 3.17. Interfaz de la aplicación MPASM.

Una vez generado el archivo HEX se utiliza el programa PIC-600 US BURN para grabarlo en el PIC, ver Fig. 3.18.

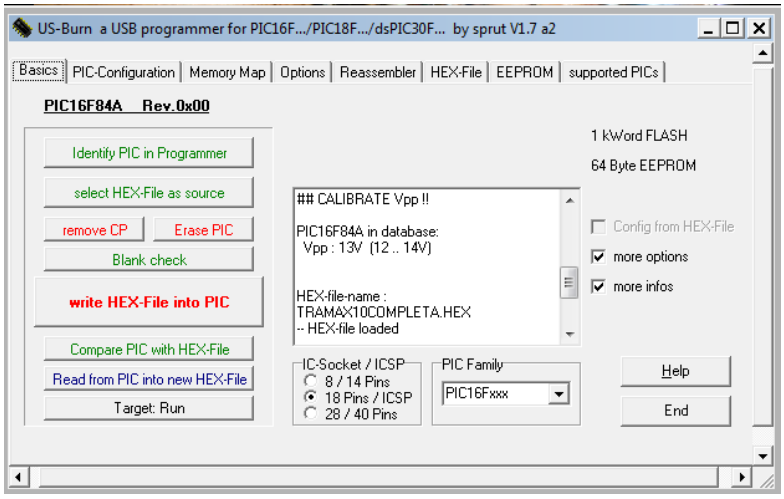

Figura 3.18.- Interfaz de la aplicación PIC 600US BURN.

Los diagramas de flujo comentados con anterioridad se ven reflejados en el código ensamblador, dicho código se encuentra documentado en los anexos de este documento.

CAPITULO 4

RESULTADOS

En esta sección se muestra los resultados obtenidos de las pruebas realizadas con el protocolo X-10. El protocolo fue programado en lenguaje ensamblador pero anteriormente se utilizó PIC Basic Pro para el envío de la trama, por lo tanto se incluyen formas de onda obtenidas con ambos lenguajes de programación.

4.1. X-10CON PIC BASIC

En PIC Basic Pro posee dos comandos para el protocolo X-10 uno es para transmitir (XOUT) y el otro para recibir (XIN); el comando utilizado fue el XOUT, el cual tiene la siguiente sintaxis:

XOUT DataPin, ZeroPin, [HouseCode\KeyCode {\Repeat}]

Envía un código de casa *HouseCode* seguido por un código de control *KeyCode*, repetidos un número *Repeat* de veces en formato X-10. Si no se usa *Repeat* se asume 2 veces como mínimo. XOUT se usa para enviar información de control a dispositivos X-10.

DataPin es automáticamente convertido en salida para enviar datos a la interfaz X-10. ZeroPin es automáticamente convertido en entrada para recibir el tiempo de cruce por cero de la interface X-10[8].

PIC Basic incluye una tabla de códigos de comando con valores binarios totalmente diferente a la que se utiliza según la teoría de la transmisión X-10. Los códigos de comando terminan en cero cuando deberían terminar en uno.

Se programó lo siguiente.

XOUT PORTA.1,PORTA.0,[house\unit,house\unitOn]

Donde “house” corresponde al código de la casa M, “unit” corresponde al código de unidad y “unitOn” es el comando encender.

[8] Tomado de la sección de ayuda del programa PIC Basic Compiler Pro

La trama X-10 generada por el comando XOUT no pudo ser captada completamente. La Fig. 4.1 muestra el código de inicio que corresponde al número binario 1110, cada ovalo representa un uno, dentro de cada ovalo se pueden apreciar tres pulsos, los cuales equivalen al mismo, esto sucede para poder enviarlo en un posible sistema trifásico. Se puede observar que los pulsos no corresponden a la ubicación señalada; es decir existen retrasos muy grandes.

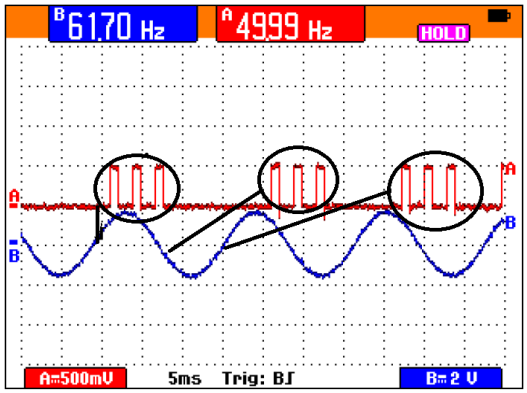

Figura 4.1. Código de inicio.

En la Fig. 4.2 se muestra un acercamiento con ambas señales sobrepuestas. Es posible observar que existe la sincronía con cero y que el tiempo entre los pulsos para un posible sistema trifásico corresponden a los de la teoría de la transmisión explicada en el capítulo de conceptos básicos.

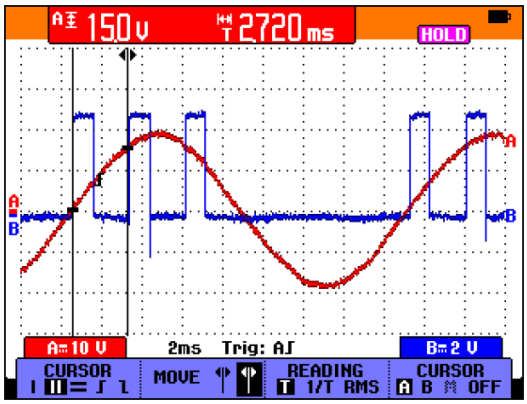

Figura 4.2. Zoom del código de inicio.

En la Fig. 4.3 los pulsos tienen una frecuencia de 120 KHz, existe la sincronía con cero y el segundo pulso ha sido enviado en el flanco correspondiente a diferencia del tercer pulso.

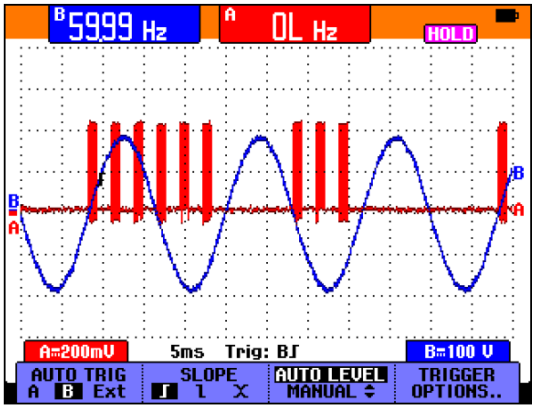

Figura 4.3. Código de inicio con frecuencia de 120 KHz.

La duración de los pulsos debe ser de 1 ms, el resultado obtenido con el comando XOUT se muestra en la Fig. 4.4.

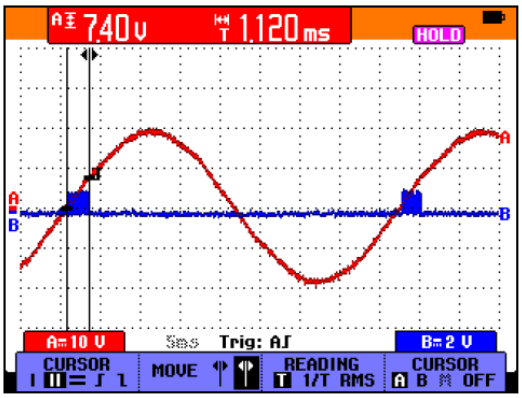

Figura 4.4. Duración del pulso.

Los resultados obtenidos con PIC Basic Pro utilizando el comando XOUT no fueron los deseados, porque el código de casa y el código de comando que recibía el receptor eran diferentes a lo programado en el transmisor; cada vez que se recibía la trama dichos códigos cambiaban.

4.2. X-10 CONLENGUAJE ENSAMBLADOR

Los resultados obtenidos en el lenguaje ensamblador fueron totalmente diferentes; en esta ocasión se logró capturar mayor parte de la trama. A continuación se describen los resultados obtenidos.

4.2.1. X-10 SIN FRECUENCIADE 120 KHZ

En la Fig. 4.5 se observa el código de inicio1110. Es importante recordar que el código de inicio es el único que no se envía en complemento.

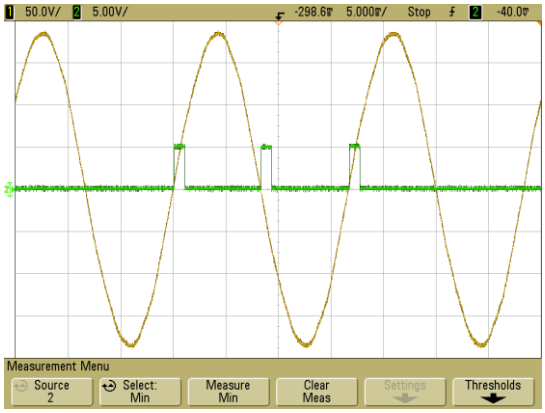

Figura 4.5. Código de inicio.

La trama X-10 es enviada en dos paquetes, en la Fig. 4.6 se observa el primero; el cual está compuesto por el código de inicio (1110), código de comando (0000) y código de unidad (01100).El último pulso se encuentra subrayado en azul porque es con el que se identifica si es un código de unidad o de comando. Este paquete es enviado a la casa M unidad 1, ver tablas 1.1 y 1.2. Los pulsos son enviados en el cruce por cero, en esta toma los pulsos no cuentan con la frecuencia de 120 KHz por lo tanto el receptor no detectaría la señal.

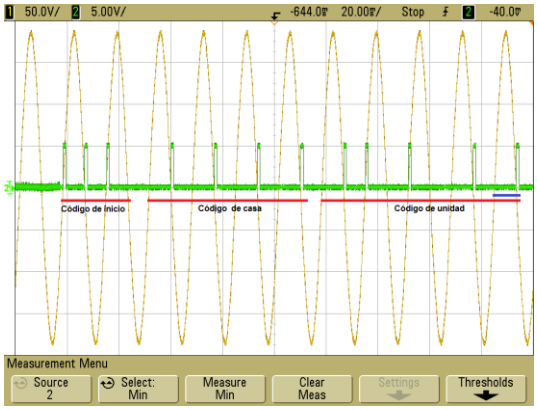

Figura 4.6. Primer paquete de la trama X-10.

Antes de enviar el segundo paquete, es necesario esperar seis cruces equivalentes a 3 ciclos de la onda senoidal, ver Fig. 4.7.

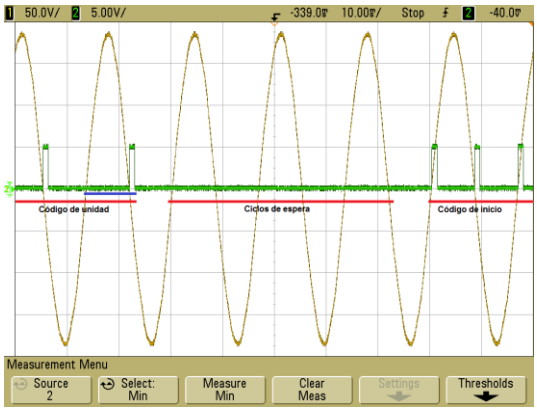

Figura 4.7. Ciclos de espera.

La Fig. 4.8 ilustra el segundo paquete de la trama, este se envía a la casa M con el comando encender. El último pulso indica que es un código de comando.

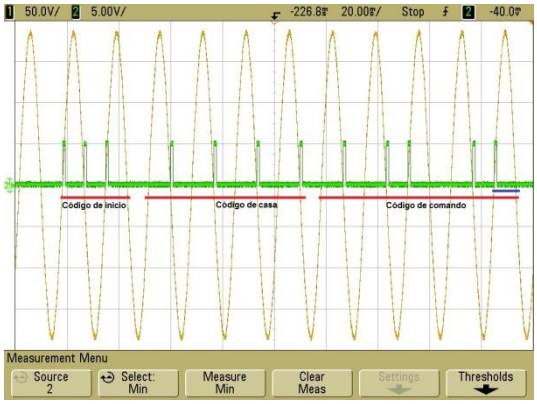

Figura 4.8. Segundo paquete de la trama X-10.

En la Fig. 4.9 se observa la trama completa, se aprecian los dos paquetes que la conforman así como los ciclos de espera, cada paquete es enviado dos veces.

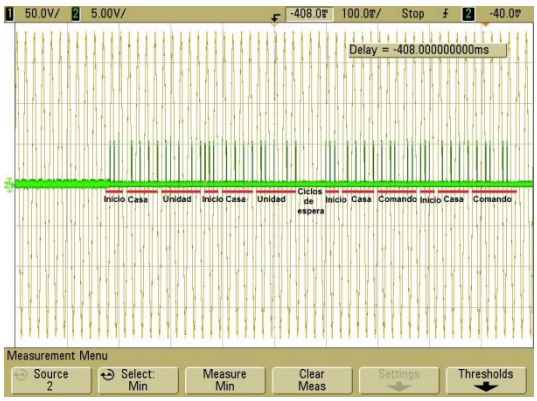

Figura 4.9. Trama completa.

El retraso en el flanco de subida que se obtuvo al programar el protocolo X-10 fue de 60 µs y el retraso en el flanco de bajada fue de19µs, ver Fig. 4.10 y 4.11.

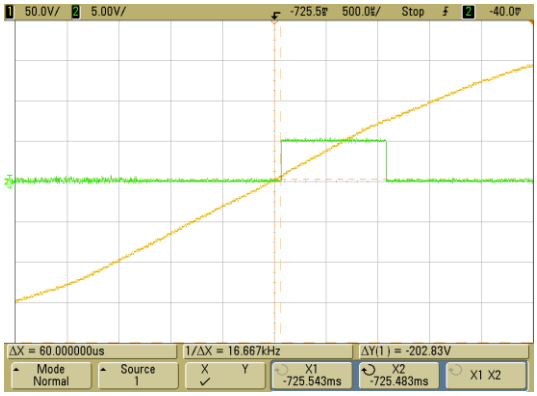

Figura 4.10. Retraso en el flanco de subida.

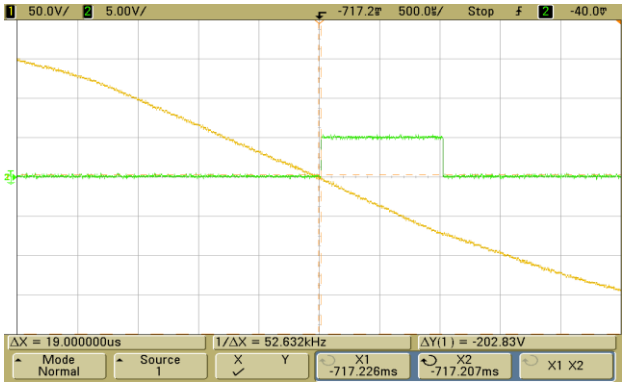

Figura 4.11. Retraso en el flanco de bajada.

Todos los pulsos enviados duraban 1 ms como se muestra en la Fig.4.12.

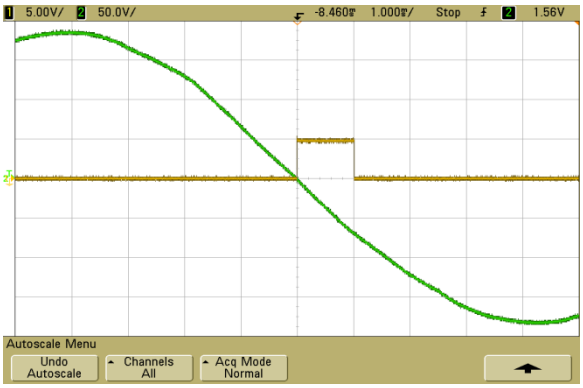

Figura 4.12. Duración del pulso.

4.2.2. X-10 CON FRECUENCIADE 120 KHZ

Las formas de onda que se muestran en las siguientes páginas corresponden a partes de la trama X-10 introduciéndoles una frecuencia de 120 KHz, frecuencia necesaria para transmitir la tramaX-10.

En la Fig. 4.13 se observa que la duración de los pulsos que se envían es de 1 ms.

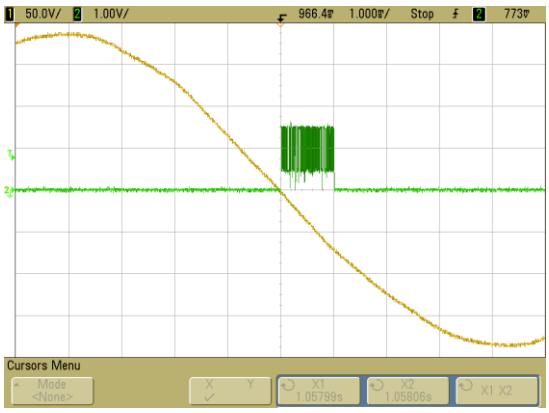

Figura 4.13. Duración del pulso.

La Fig. 4.14 muestra un acercamiento del código de inicio.

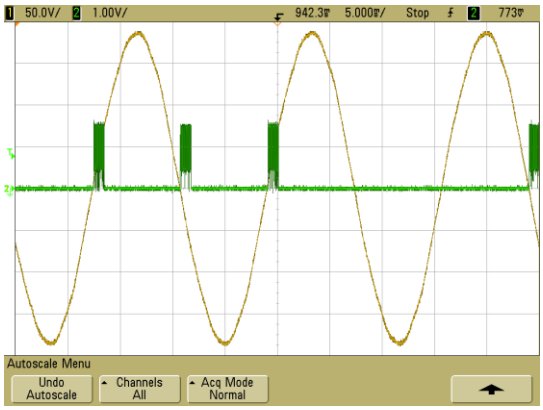

Figura 4.14. Código de inicio

En la Fig. 4.15 se observa la terminación del primer paquete X-10, los ciclos de espera y el código de inicio del segundo paquete X-10.

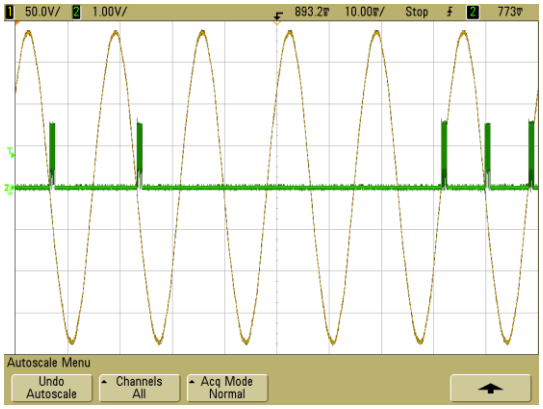

Figura 4.15. Ciclos de espera.

Las formas de onda anteriores demuestran que es posible transmitir señales de control a través de la línea eléctrica. El resultado obtenido de esta transmisión fue el controlar el encendido de un foco de 60 Watts, ver Fig. 4.16.

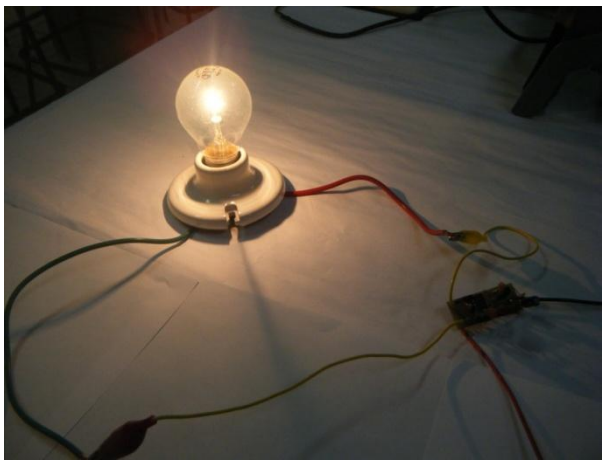

Figura 4.16. Resultado obtenido del envío de la trama X-10.

El circuito transmisor se conecta a la corriente eléctrica en cualquier contacto de la casa y el circuito receptor se conecta a la carga eléctrica. El circuito receptor utilizado es comercial, posee dos cables que se conectan a la línea eléctrica y dos cables que se conectan al foco o a la carga eléctrica que se vaya a controlar.

Los datos enviados desde el transmisor llegan al receptor y dependiendo del comando enviado es la acción que realiza. La Fig. 4.17 muestra el transmisor y el receptor en funcionamiento.

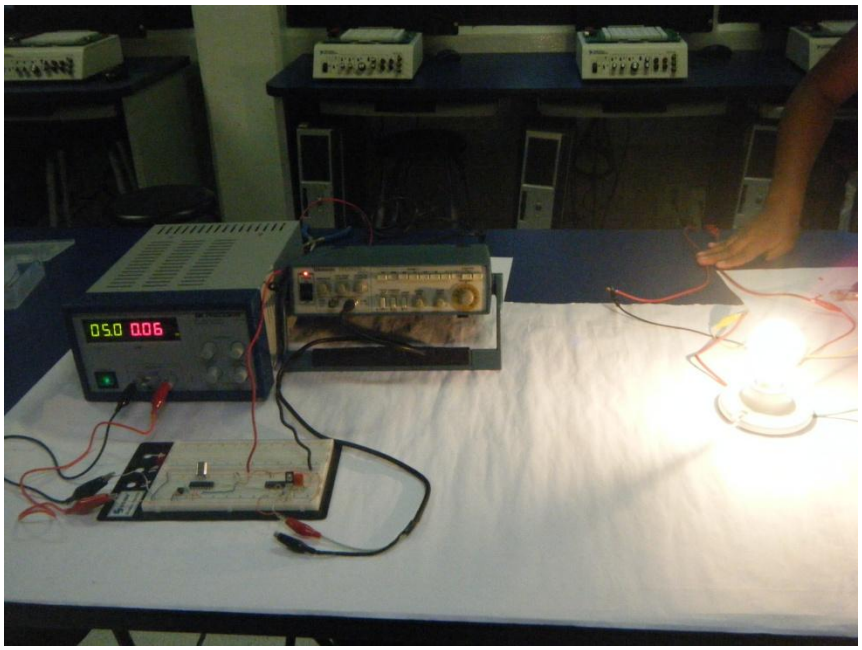

Figura 4.17. Transmisor y receptor X-10 en funcionamiento.

CONCLUSIONES

La implementación de un sistema domótico conlleva una aplicación de diversos conocimientos adquiridos durante la formación como ingeniero en Redes. Se involucran áreas de comunicaciones, electrónica, electricidad, programación entre otras. El objetivo de esta tesis fue el estudio, diseño e implementación de un sistema de comunicación y control empleando la línea de CA.

Durante la realización de este trabajo, surgen una gran cantidad de aspectos a considerar, principalmente durante la implementación. Aspectos como el ruido, perturbaciones, retardos y demás, nos enseñan que el diseño de un sistema conlleva una complejidad más allá de lo aprendido en el aula. Este trabajo puede servir de base para sistemas más complejos.

El diseño de un sistema de comunicación por la línea eléctrica permitió utilizar la instalación eléctrica ya existente, dando como resultado que se pueda controlar y automatizar los aparatos que se conectan a la línea eléctrica.

A continuación se listan las principales conclusiones del trabajo

- ✓ El protocolo X-10 es susceptible al ruido externo, de hecho en ambientes con alto grado de contaminación electrónica, es prácticamente imposible que funcione el sistema. Tener cerca fuentes de ruido (taladros, licuadoras, maquinas de soldar) contaminan la red de tal manera que los datos se pierden.
- ✓ El manejo de tiempos en el protocolo, (200 µs) es crucial para poder enviar datos, por ello la selección del lenguaje de programación de bajo nivel, es un aspecto relevante durante la etapa del diseño
- ✓ Los esquemas electrónicos son sencillos pero se debe tener cuidado en el manejo de señales (amplificación y filtrado) para lograr una comunicación.
- ✓ Este esquema no es recomendable para cargas críticas, dada su susceptibilidad de fallo.

BIBLIOGRAFIA

[1] CEDOM. (s.f.). Como ahorrar energía instalando domótica en su vivienda. Gane en confort y seguridad. España.

[2] Programa de Ahorro de Energía del Sector Eléctrico. (s.f.). *Comisión federal de elctricidad*. Recuperado el 3 de Febrero de 2010, de http://www.cfe.gob.mx/sustentabilidad/ahorroenergia/Paginas/ahorrodeenergia.aspx

[3] Cristóbal Romero Morales, Francisco Vázquez Serrano, Carlos de Castro Lozano, (2007). Domótica e Inmótica. Viviendas y Edificios Inteligentes (Segunda edición). Madrid, España: RA-MA

[4] de Palencia, L. F., Angulo, J. M., & Romero, S. (Febreo de 2001). Domótica: Comunicaciones por red eléctrica. *Revista española de electronica* , 2.

[5] X10 Corporation. (s.f.). Recuperado el 18 de Febrero de 2010, de X10 Technology Transmission Theory: www.x10.com

[6] Angulo Usategui, J. M., & Angulo Martínez, I. (2003). *Microcontroladores PIC diseño práctico de aplicaciones primera parte* (Tercera edición ed.). Madrid: Mc-Graw Hill.

[7] Angulo Usategui, J. M., Romero Yesa, S., & Angulo Martinez, I. (2006). *Microcontroladores PIC diseño práctico de aplicaciones segunda parte* (Segunda edición ed.). Madrid: Mc Graw Hill.

[8] Microchip. (s.f.). PIC 16F84A Data Sheet.

[9] Cox, D. (s.f.). *Microchip.*Recuperado el 1 de Mayo de 2010, de http://ww1.microchip.com/downloads/en/AppNotes/00521c.pdf

[10] Burroughs, J. (s.f.). *Microchip.*Recuperado el 1 de Mayo de 2010, de http://ww1.microchip.com/downloads/en/AppNotes/00236B.pdf

ANEXOS

CODIGO PARA EL TRANSMISOR X-10

```
;TRAMAX10COMPLETA:ASM
;programa que envía una trama X-10
        LISTP=16F84A            ;Comando que indica el PIC usado
        RADIXHEX                ;Los valores se representan en hexadecimal
;*****************************************************************************
ETIQUETAS
;*****************************************************************************
W       EQU     0X00    ;define la posición W
F       EQU     0X01    ;define la posición F
;----- Register Files----------------------------------------------------
TMR0            EQU     0X01    ;define la posición TMR0
STATUS          EQU     0X03    ;define la posición STATUS
PORTA           EQU     0X05    ;define la posición PORTA
PORTB           EQU     0X06    ;define la posición PORTB
INTCON          EQU     0X0B    ;define la posición INTCON
OPTION_REG      EQU     0X81    ;define la posición OPTION_REG
TRISA           EQU     0X85    ;define la posición TRISA
TRISB           EQU     0X86    ;define la posición TRISB

;----- INTCON Bits --------------------------------------------------------
INTF            EQU     0x01    ;define la posición INTF
;----- OPTION Bits --------------------------------------------------------
INTEDG                  EQU     0x06    ;define la posición INTEDG
;----- STATUS Bits --------------------------------------------------------
RP0             EQU     0x05    ;define la posición RP0
Z               EQU     0x02    ;define la posición Z
;------------- variable auxiliar ----------------------------------------
CONTA           EQU     0X0C    ;define la posición CONTA
;*****************************************************************************
MACROS
;*****************************************************************************
;;macro para configurar un puerto
CONFPORT MACRO PUERTO, VALl1    ;inicia el macro
        BSF     STATUS,RP0              ;entra modo configuración (banco 1)
        MOVLW           VAL             ;W <-- val1val1 se carga a W para configurar el Puerto
como E/S
        MOVWF           PUERTO          ;W --> Puerto el valor de w se carga al puerto
        BCF     STATUS,RP0              ; sale del modo de configuración (banco 0)
        ENDM                            ;termina el macro
;*****************************************************************************
        ORG     0x00
        GOTO    inicio
;*****************************************************************************
;CONFIGURACIÓN DE PUERTOS
;*****************************************************************************
```

```
inicio
        CONFPORT     TRISA,  0x00    ;configura puerto A como salida
        CONFPORT     TRISB,   0x01   ;configura puerto B.0 como entrada
        CONFPORT     OPTION_REG,0XD1     ;1101 0001 activa flanco ascendente y elTMR0a 1
ms
        CONFPORT     INTCON,         0X00    ; habilita la bandera INTF para activarse por
interrupciones por RB0
         GOTO        TRAMA                   ; ir a la etiqueta TRAMA
;-------------------------------------------------------------------------
TRAMA
        CALL    C_INICIO            ;llama a la subrutina C_INICIO
        CALL    CASA_UNIDAD         ;llama a la subrutina CASA_UNIDA
        CALL    C_INICIO            ;llama a la subrutina C_INICIO
        CALL    CASA_UNIDAD         ;llama a la subrutina CASA_UNIDAD
        CALL    CICLOS              ;llama a la subrutina CICLOS
        CALL    CICLOS              ;llama a la subrutina CICLOS
        CALL    CICLOS              ;llama a la subrutina CICLOS
        CALL    C_INICIO            ; llama a la subrutina C_INICIO
        CALL    CASA_COMANDO        ; llama a la subrutina CASA_COMANDO
        CALL    C_INICIO            ; llama a la subrutina C_INICIO
        CALL    CASA_COMANDO        ; llama a la subrutina CASA_COMANDO
        CALL    CICLOS              ;llama a la subrutina CICLOS
        CALL    CICLOS              ;llama a la subrutina CICLOS
        CALL    CICLOS              ;llama a la subrutina CICLOS
        CALL    CICLOS              ;llama a la subrutina CICLOS
        GOTO    END                 ;ir al final del programa
;;=========================================================================
;                       SUBRUTINAS
;=========================================================================
;**********************************C_INICIO*************************************
;subrutina que envía el código de inicio1110, el código de inicio es el único que no se envía en
complemento
;y utiliza cuatro cruces por cero para ser enviado.
;FA( flanco ascendente), FD(flanco descendente)
;la razón por la que al final se configura como FA es porque después de enviar
; el ultimo 1 el siguiente cruce por cero no se envía nada.
C_INICIO

        BTFSS INTCON,INTF       ;testea el estado de la bandera INTF del registro INTCON
        GOTO C_INICIO           ;si INTF =0vuelve a C_inicio
        CALL PULSOUP            ;si es 1 manda el pulso en el FA y se configura como FD
        BCF INTCON,INTF         ;limpia la bandera INTF
UNO1
        BTFSS INTCON,INTF       ;testea el estado de la bandera INTF del registro INTCON
        GOTO UNO1               ;si es 0 vuelve a UNO1
        CALL PULSODOWN          ; si es 1 manda el pulso en FD y lo configura como FA
        BCF INTCON,INTF         ;limpia la bandera INTF
```

```
UNO2
        BTFSS INTCON,INTF       ;testea el estado de la bandera INTF del registro INTCON
        GOTO UNO2               ;si INTF =0 vuelve a UNO2
        CALL PULSODOWN          ; si INTF=1 manda el puso en FA y se configura como FA
        BCF INTCON,INTF         ;limpia la bandera INTF
        RETURN                  ; regresa a la rutina principal

;*****************************CASA_UNIDAD*************************************
*******************************************************************************
; Subrutina que envía el código de casa y el código de unidad: 0000 01100
CASA_UNIDAD
        ; comienza el código de casa
        CALL    E_0             ; llama a la subrutina E_0
        CALL    E_0             ; llama a la subrutina E_0
        CALL    E_0             ; llama a la subrutina E_0
        CALL    E_0             ; llama a la subrutina E_0
        ; comienza el código de unidad
        CALL    E_0             ; llama a la subrutina E_0
        CALL    E_1             ; llama a la subrutina E_1
        CALL    E_1             ; llama a la subrutina E_1
        CALL    E_0             ; llama a la subrutina E_0
        CALL    E_0             ; llama a la subrutina E_0
        RETURN                  ; regresa a la rutina principal

;*****************************CASA_COMANDO************************************
*******************************************************************************
; subrutina que envía el código de casa y el código de comando (encender) : 0000 01100

CASA_COMANDO
        ; comienza el código de casa
        CALL    E_0             ; llama a la subrutina E_0
        CALL    E_0             ; llama a la subrutina E_0
        CALL    E_0             ; llama a la subrutina E_0
        CALL    E_0             ; llama a la subrutina E_0
        ;comienza el código de comando
        CALL    E_0             ; llama a la subrutina E_0
        CALL    E_0             ; llama a la subrutina E_0
        CALL    E_1             ; llama a la subrutina E_1
        CALL    E_0             ; llama a la subrutina E_0
        CALL    E_1             ; llama a la subrutina E_1
        RETURN                  ;regresa a la rutina principal
;*********************************E_0***************************************
*******************************************************************************
; la subrutina E_0 envía un 0 binario en complemento, utiliza dos cruces por cero
; solo en el segundo cruce por cero envía un pulso de 1ms
```

```
E_0
        BTFSS   INTCON,INTF         ;testea el estado de la bandera INTF del registro INTCON
        GOTO    E_0                 ;si INTF =0 vuelve a E_0
        CALL    CONFIGDOWN          ;si INTF=1 ; llama a la subrutina CONFIGDOWN
        BCF     INTCON,INTF         ;limpia la bandera INTF
E_00
        BTFSS   INTCON,INTF         ;testea el estado de la bandera INTF del registro INTCON
        GOTO    E_00                ;si es 0 vuelve aE_00
        CALL    PULSODOWN           ; si es 1 llama a la subrutina PULSODOWN
        BCF     INTCON,INTF         ;limpia la bandera INTF
        RETURN                      ; regresa a la subrutina de donde fue llamada

;**********************************E_1*****************************************
*******************************************************************************
; la subrutina E_1 envía un 1 binario en complemento, utiliza dos cruces por cero
;solo en el primer cruce envía un pulso de 1ms

E_1
        BTFSS   INTCON,INTF         ;testea el estado de la bandera INTF del registro INTCON
        GOTO    E_1                 ;si es 0 vuelve a E_1
        CALL    PULSOUP             ; si es 1 llama a la subrutina PULSOUP
        BCF     INTCON,INTF         ;limpia la bandera INTF
E_11
        BTFSS   INTCON,INTF         ;testea el estado de la bandera INTF del registro INTCON
        GOTO    E_11                ;si INTF=0 vuelve a E_11
        CALL    CONFIGUP            ;si INTF=1 llama a la subrutina CONFIGUP
        BCF     INTCON,INTF         ;limpia la bandera INTF
        RETURN                      ;regresa a la subrutina de donde fue llamada
;**************************************CICLOS***********************************
*******************************************************************************
; Subrutina que deja pasar un ciclo sin envío de datos
CICLOS
        BTFSS   INTCON,INTF         ;testea el estado de la bandera INTF del registro INTCON
        GOTO    CICLOS              ;si INTF =0 vuelve a E_0
        CALL    CONFIGDOWN          ;si INTF=1 llama a CONFIGDOWN
        BCF     INTCON,INTF         ;limpia la bandera INTF
CICLOS_00
        BTFSS   INTCON,INTF         ;testea el estado de la bandera INTF del registro INTCON
        GOTO    CICLOS_00           ;si INTF =0 vuelve a CERO2
        CALL    CONFIGUP            ;si INTF=1 llama a CONFIGUP
        BCF     INTCON,INTF         ;limpia la bandera INTF
        RETURN                      ;regresa a la rutina principal

;**********************************PULSODOWN************************************
*******************************************************************************
;envía un pulso en el flanco descendente y configura el flanco de detección como flanco ascendente
```

```
PULSODOWN
        CALL PULSE              ;llama a la subrutina PULSE
        CALL CONFIGUP           ;llama a la subrutina CONFIGUP
        RETURN                  ;regresa a la subrutina E_0

;*************************************PULSOUP*************************************
;*********************************************************************************
;envía un pulso en el flanco ascendetente y configura el flanco de detección como flanco
descendete

PULSOUP
        CALL PULSE              ;genera un pulso de 1ms por el PORTA,0
        CALL CONFIGDOWN         ;activa flanco descendente, pone en 0 a INTF y pone a
GIE=1
        RETURN                  ;regresa a la subrutina E_1

;*************************************PULSE***************************************
;*********************************************************************************
;generaun pulso de 1ms

PULSE
        BSF PORTA,0             ;enciende el puerto A.0
        CALL PAUSA              ;llama a PAUSA
        BCF PORTA,0             ;apaga puerto A.0
        RETURN                  ;regresa a la subrutina de donde fue llamada
;*************************************PAUSA***************************************
;*********************************************************************************
;utiliza el TIMER como un temporizador con duracion de 1ms

PAUSA
        CLRF TMR0               ;reinicia el TMR0
REVISAR
        MOVF   TMR0,W           ;carga el valor del timer a W
        SUBLW0XFA               ;restaw- 0xFA
        BTFSS   STATUS,Z        ;verifica si se llega al límite del timer testeando el bit Z
        GOTO   REVISAR          ; si z=0 regresa a la eiqute REVISAR
        RETURN                  ; si z=1 regresa a la subrutina PULSE

;*************************************CONFIGUP************************************
;*********************************************************************************
;configura el bit INTDEG del registro OPTION_REG para que el flanco activo de interrupción externa
sea el flanco ascendente

CONFIGUP
        BSF STATUS,RP0          ;entra al modo de configuración
        BSF OPTION_REG, INTEDG  ;activa detección de flanco ascendente
        BCF STATUS,RP0          ;sale del modo de configuración
```

```
        RETURN                          ; regresa a la subrutina de la donde fue llamada
;*************************CONFIGDOWN*************************************
**************************************************
;configura el bit INTDEG del registro OPTION_REG para que el flanco activo de interrupción externa
sea el flanco descendente

CONGFIGDOWN

        BSF STATUS,RP0                  ;se entra modo configuración
        BCF OPTION_REG, INTEDG          ;activa flanco descendente
        BCF STATUS,RP0                  ;se sale modo configuración
        RETURN                          ; regresa a la subrutina de la donde fue llamada

        END                             ;termina el programa
```