SISTEMA DE LECTURA PARA CONTADOR DE ENERGIA ELECTRICA A TRAVES DE LA LINEA TELEFONICA

OBERT PATERNINA AGUIRRE

MOISES TAMAYO JIMENEZ

ORLANDO VILLADIEGO ORELLANO

CORPORACION UNIVERSITARIA TECNOLOGICA DE BOLIVAR

FACULTA DE INGENIERIA ELECTRICA Y ELECTRONICA

CARTAGENA DE INDIAS D. T. Y C

1999

SISTEMA DE LECTURA PARA CONTADOR DE ENERGIA ELECTRICA

A TRAVES DE LA LINEA TELEFONICA

OBERT PATERNINA AGUIRRE

MOISES TAMAYO JIMENEZ

ORLANDO VILLADIEGO ORELLANO

Trabajo de grado presentado como requisito parcial para optar al titulo de ingeniero electricista

Director
NOLBERT RUMBO AVILA
Ingeniero Electricista

CORPORACION UNIVERSITARIA TECNOLOGICA DE BOLIVAR

FACULTA DE INGENIERIA ELECTRICA Y ELECTRONICA

CARTAGENA DE INDIA D.T Y C

1999

Cartagena, Octubre 13 de 1998

Señores

COMITÉ DE PROYECTO DE GRADO

CORPORACION UNIVERSITARIA TECNOLOGICA DE BOLIVAR

L.C.

Distinguidos señores:

Como director de tesis de los señores OBERT PATERNINA AGUIRRE, MOISES TAMAYO JIMENEZ Y ORLANDO VILLADIEGO ORELLANO estudiantes matriculados en el programa de ingeniería eléctrica, me permito presentar a consideración para su estudio y aprobación el trabajo de grado titulado " SISTEMA DE LECTURA PARA CONTADOR DE ENERGIA ELECTRICA A TRAVES DE LA LINEA TELEFONICA " Con el fin que dichos estudiantes obtengan él titulo de ingeniero electricista.

Atentamente

_____
Ing. Electricista
NOLBER RUMBO AVILA

Cartagena, octubre 13 de 1998

Señores

COMITÉ DE PROYECTO DE GRADO

CORPORACION UNIVERSITARIA TECNOLOGICA DE BOLIVAR

L.C.

Distinguidos señores:

La presente tiene como objetivo presentar a consideración para su estudio y aprobación el proyecto de grado titulado " SISTEMA DE LECTURA PARA CONTADOR DE ENERGIA ELECTRICA A TRAVES DE LA LINEA TELEFONICA " Para optar él titulo de ingeniero electricista.

Agradeciendo de antemano la atención prestada.

Cordialmente:

_____

OBERT PATERNINA AGUIRRE
C.C 73,159.906 de Cartagena

Cartagena, octubre 13 de 1998

Señores

COMITÉ DE PROYECTO DE GRADO

CORPORACION UNIVERSITARIA TECNOLOGICA DE BOLIVAR

L.C.

Distinguidos señores:

La presente tiene como objetivo presentar a consideración para su estudio y aprobación el proyecto de grado titulado " SISTEMA DE LECTURA PARA CONTADOR DE ENERGIA ELECTRICA A TRAVES DE LA LINEA TELEFONICA " Para optar él titulo de ingeniero electricista.

Agradeciendo de antemano la atención prestada.

Cordialmente:

_____

MOISES TAMAYO JIMENEZ

C.C 73,148.792 de Cartagena

CONTENIDO

LISTA DE FIGURAS

LISTA DE CUADROS

Nota de aceptación

_____

_____

_____

_____
Presidente del jurado

_____
Jurado

_____
Jurado

Cartagena, 1999.

Articulo 105. La corporación se reserva el derecho de propiedad intelectual de todos los trabajos de grado aprobados, los cuales no pueden ser explotados comercialmente sin su aceptación.

RESUMEN

El objetivo de esta investigación fue desarrollar un sistema basado en microcontroladores capaz de digitalizar la información de un contador electromecánico y enviarla a una unidad de consulta a través de la línea telefónica. Este proyecto se realizo de manera descriptiva para facilitar la comprensión del texto. Se tiene como resultado y a la ves se concluye que este tipo de investigación es realizable y aplicable a las necesidades de conversión y transmisión de información localizada en lugares de poco o difícil acceso. Además denotamos la importancia que tienen los microcontroladores al ser aplicables a sistemas que realicen funciones especificas, reduciendo el hardware necesario y los costos; facilitando la programación del sistema y brindando velocidades de proceso suficientes para los trabajos a realizar.

1. UNIDAD DE LECTURA

1.1 CIRCUITO TRANSDUCTOR

Es el encargado de generar un pulso definido por cada media revolución del disco del contador electromecánico. Esto lo logramos haciendo dos perforaciones al disco giratorio, situadas 180 grados una respecto la otra. De esta forma un optoacoplador situado estratégicamente sensa cada media revolución dada por el disco. Ver figura 1.

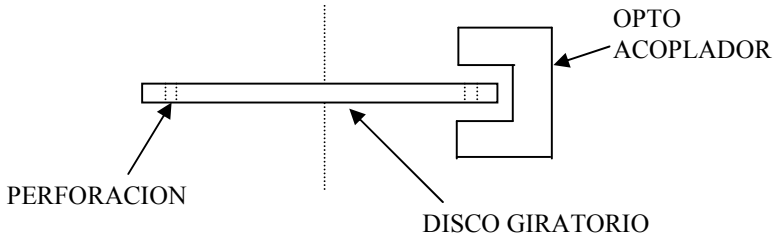


Figura 1. Sensor de revoluciones del contador de energía.

La razón de utilizar dos perforaciones, es debido a que el registro RTCC del microcontrolador se incrementa en uno por cada dos pulsos entregados por el circuito transductor al pin TOCK, por tanto se obtiene un incremento en el registro RTCC por cada revolucion del disco. En la figura 2 observamos el circuito transductor completo, el cual está conformado por un optoacoplador de referencia

MST 8 y una sección del flipflop data CD4013.

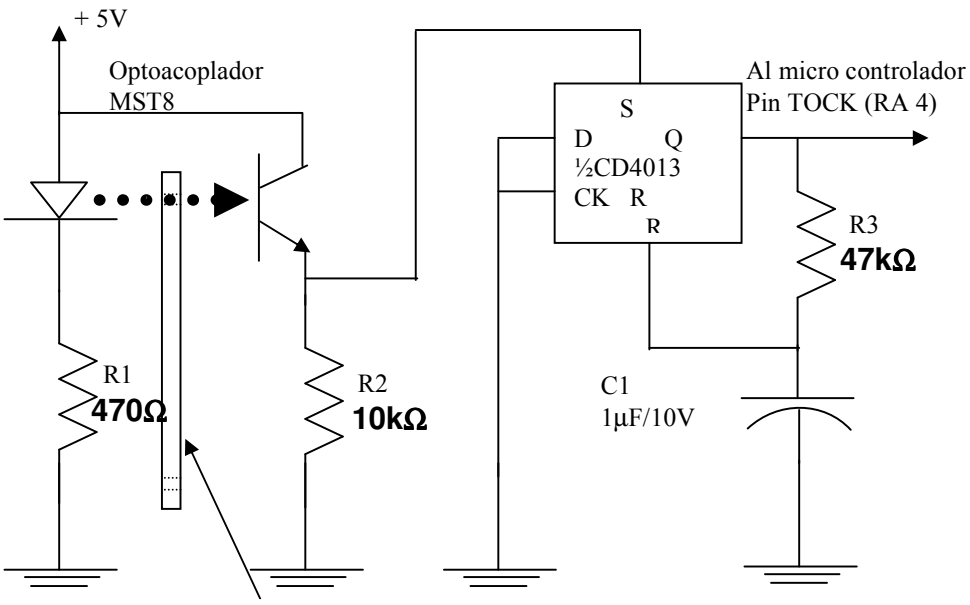


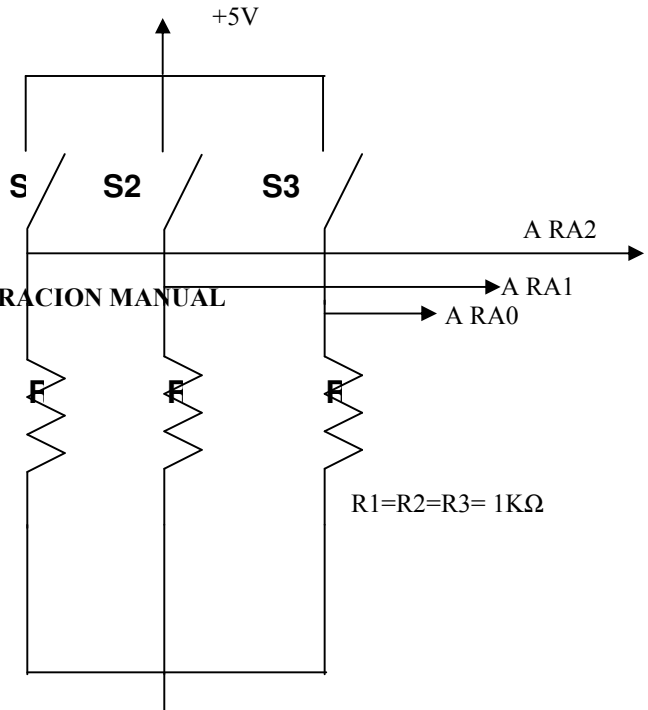
Figura 2. Circuito transductor

El optoacoplador al detectar cada media revolución hace que el transistor se sature produciendo un nivel alto en el emisor, generando de esta forma un SET en el flipflop DATA para conformar un pulso bien definido en la salida Q, es decir este pasa de un nivel bajo a un nivel alto produciendo un flanco de subida requerido en el pin TOCK (RA4) del microcontrolador.

El retardador formado por R3 y C1 se calculó para que la salida Q retome el nivel bajo mediante un reset al flipflop data después de un cuarto de segundo de haber ocurrido un pulso a media revolución del disco.

1.2 CIRCUITO SELECTOR DE kd

El circuito selector de Kd está formado por un juego de tres resistencias y tres interruptores (Dip Switch) que colocan un nivel lógico alto o bajo en los pines RA0, RA1 y RA2 del microcontrolador como se aprecia en la figura 3.



1.3 ETAPA DE CALIBRACION MANUAL

Esta etapa está compuesta por el temporizador electrónico 555 trabajando en modo astable. El objetivo de esta etapa es igualar la lectura digital de la unidad de lectura con la lectura del contador electromecánico para de esta forma poner en funcionamiento todo el sistema.

En la figura 4 se observa la etapa de calibración manual, cuya salida se dirige al pin RA4 a través del suiche SW1, en esta etapa se producen tres frecuencias de oscilación de acuerdo a la selección de los pulsadores P1, P2 y P3.

P1 selecciona al condensador C1=4.7 microfaradios, el cual impone una frecuencia de oscilación de 19.93 Hz.

P2 selecciona al condensador C2=0,1 microfaradios, el cual impone una frecuencia de oscilación de 937.01 Hz.

P3 selecciona al condensador C3=0,0022 microfaradios el cual impone una frecuencia de oscilación de 42,6 KHz.

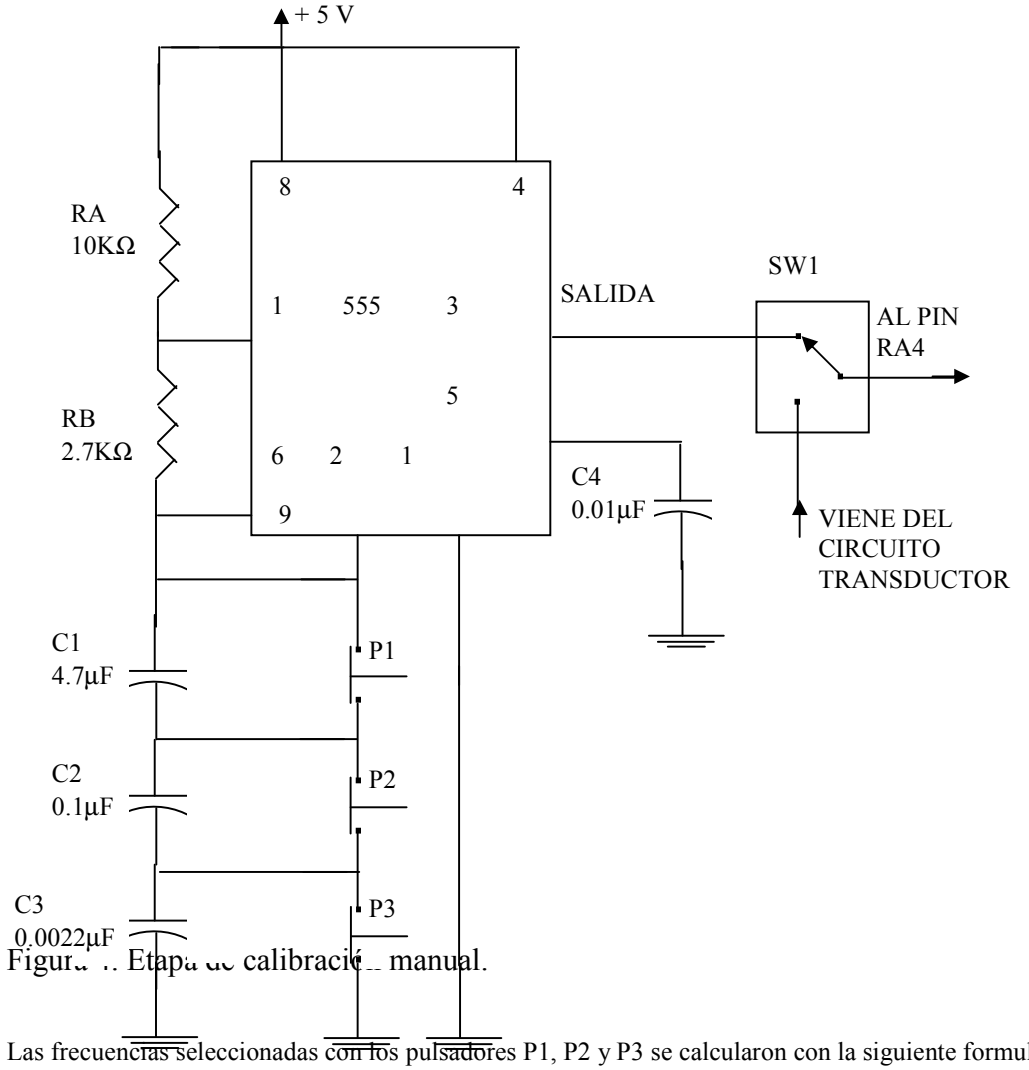


Figura .. Etapa de calibración manual.

Las frecuencias seleccionadas con los pulsadores P1, P2 y P3 se calcularon con la siguiente formula:

T= 0,693 (RA +2RB)*C

Donde: T= período de oscilación en segundos.

f= 1/T =frecuencia en Hz.

La etapa de calibración se diseñó para un ciclo útil de trabajo del 60% (es decir que el pulso se mantiene en alto un 60% del tiempo del período). Esto basado en la siguiente expresión:

C(%)= RA/(RA+2RB)

Donde: C(%)= ciclo útil de trabajo en %.

Se seleccionó RA+RB menor a 20 Megaohmios y los condensadores mayores a 0,001 microfaradios para obtener una estabilidad del 1% según recomendaciones del fabricante.

La calibración se efectúa de la siguiente manera:

- Una ves instalado el equipo y seleccionado el valor de Kd(rev/Kw-h), se procede a su energización.
- Se resetea el microcontrolador.
- Se coloca el SW1 en la posición de calibración manual.
- Se presiona el pulsador de ajuste rápido, hasta alcanzar una lectura cercana a la del contador electromecánico.
- Se presiona el pulsador de ajuste medio, para acercarnos aun más al valor de la lectura del contador.
- Se hace un ajuste fino, hasta igualar las dos lecturas.
- Se coloca SW1 en la posición del circuito transductor, para hacer el conteo de las revoluciones del disco del contador.

1.4 TARJETA DE VISUALIZACION

Para la visualización se utiliza el método de multiplexación de datos. El microcontrolador envía a la tarjeta de visualización a través de los pines RB3, RB2, RB1 Y RBO el valor en binario de los registros que contienen la información referente a las décimas, unidades, decenas, centenas, millares y diez millares de Kw/h consumidos (Registros dig0, dig1, dig2,dig3,dig4,dig5), siendo RB0 el LSB (BIT menos significativo) y RB3 el MSB (BIT más significativo).

Por medio de los pines RB4, RB5, RB6 el microcontrolador envía un ciclo repetitivo que viene del registro TRANS para controlar el encendido de los displays uno a uno. En la figura 5 observamos el circuito de la tarjeta de visualización.

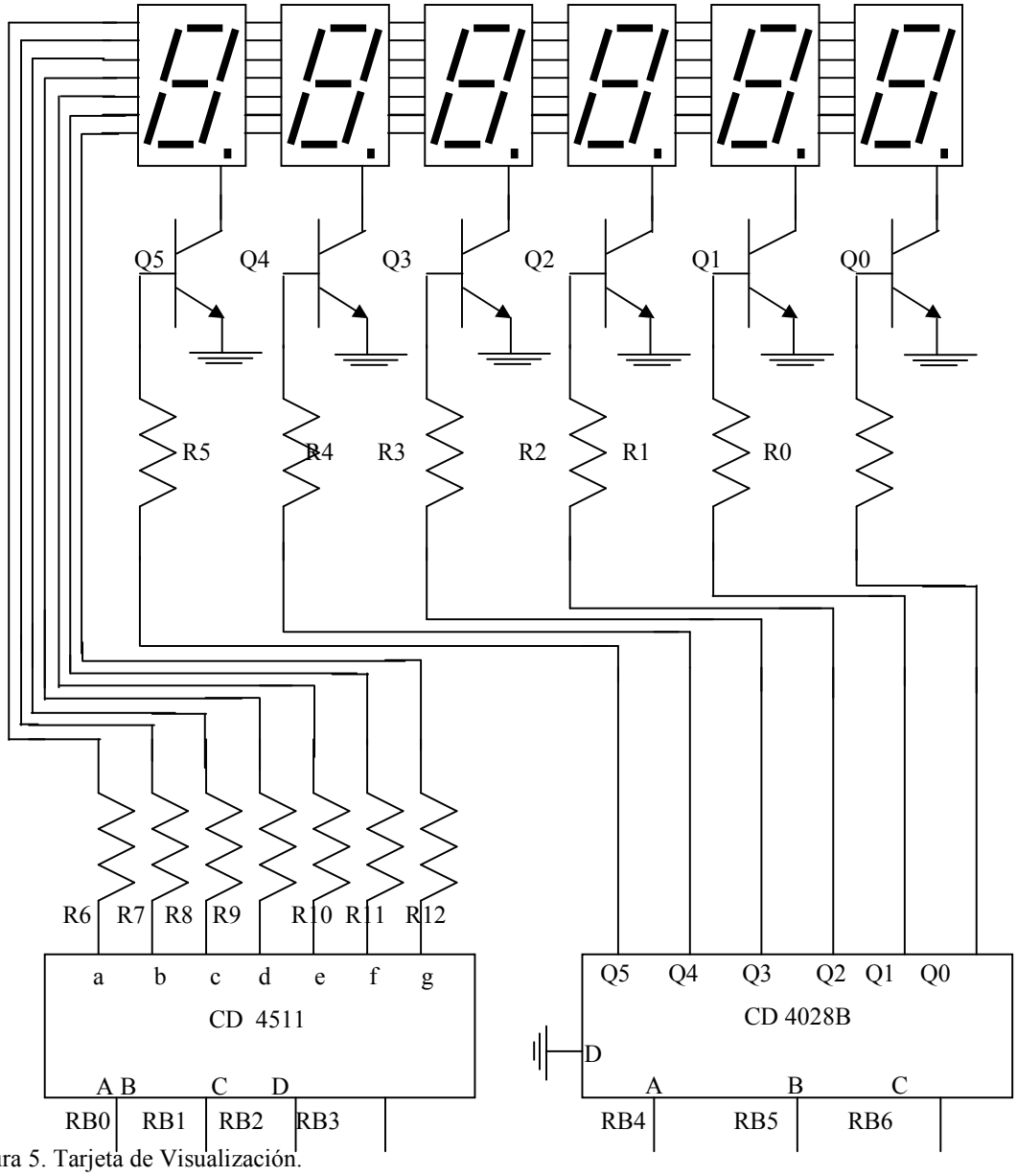


Figura 5. Tarjeta de Visualización.

R0=R1=R2=R3=R4=R5= 2,2KΩ
R6=R7=R8=R9=R10=R11=R12= 100Ω
Q0=Q1=Q2=Q3=Q4=Q5= C945

Los pines RBO a RB3 del microcontrolador llegan al decodificador CD4511 que pasa de binario a 7 segmentos. Por medio de 7 resistencias limitadoras se alimentan todos los segmentos respectivos de los 6 displays.

Los pines RB4, RB5, Y RB6 del microcontrolador llegan al decodificador CD 4028 B que pasa de binario a decimal. Las salidas de este controlan a través de las resistencias R0 a R5 el encendido de los transistores Q0 a Q5 y estos a su vez el encendido del display adecuado en el momento justo.

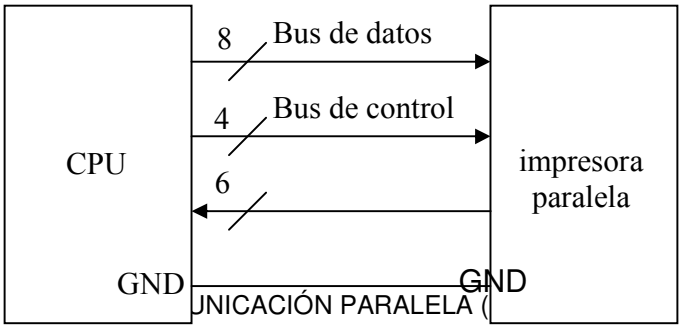
El tiempo que se necesita para sostener el dato en cada display puede variar significativamente dependiendo fundamentalmente del valor de las resistencias limitadoras, del número de dígitos a mostrar y las características propias del display. Experimentalmente mostrar cada dígito durante 3 milisegundos con resistencias limitadoras de 100 ohmios proporcionan un brillo aceptable en un display estándar y una buena visualización a una distancia prudente.

1.5 INTERFASE SERIAL RS-232

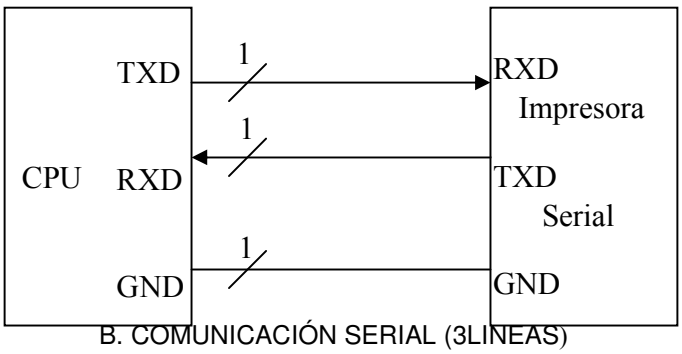
Al iniciarse el boom de los computadores electrónicos, surgió también la necesidad de intercambiar información entre estos de una manera cómoda y rápida. Como se sabe, existen dos formas de intercambiar información binaria o digital: la paralela y la serial. La comunicación paralela transmite todos los bits de un dato de manera simultánea y tiene la ventaja de que la transferencia es rápida, pero la desventaja de necesitar una gran cantidad de hilos o líneas, situación que encarece los costos y se agrava cuando las distancias que separan los equipos

entre los cuales se hace el intercambio es muy grande, debido a la capacitancia entre los conductores, la cual limita el correcto intercambio a unos pocos metros.

La comunicación serial, mientras tanto, transmite un bit a la vez, por lo cual es mucho más lenta, pero posee la ventaja de que necesita un menor número de líneas para la transferencia de la información y la distancias a la cual se puede realizar el intercambio, es mayor; a esto se suma que mediante dispositivos como los módem, la comunicación se pueda extender prácticamente a cualquier lugar del planeta. En la figura 6 se muestra un esquema de las dos formas de comunicación con las ventajas obvias que tiene la comunicación serial al reducir dramáticamente el número de líneas necesarias para la transferencia.



UNICACIÓN PARALELA (



B. COMUNICACIÓN SERIAL (3LINEAS)

Figura 6. Formas de comunicación digital.

Debemos recordar que existen dos formas de comunicación serial: la sincrónica y la asincrónica. En la comunicación sincrónica, además de una línea sobre la que se transfieren los datos, se necesita otra que contenga pulsos de reloj que indique cuando un dato es válido; la duración del bit está determinada por la duración del pulso de sincronismo. En la comunicación asincrónica, los pulsos de reloj no son necesarios y se acude a otros mecanismos para realizar la lectura/escritura de los datos: la duración de cada bit está determinada por la frecuencia de referencia con la cual se realiza la transferencia de datos; por ejemplo, transmitiendo a 1200 bits por segundo(baudios), la duración de cada bit es de 833 microsegundos (el período es el inverso de la frecuencia). Las velocidades de transmisión más comunes son 300,600,1200,2400,9600,14400,28800 baudios.

La figura 7 muestra la estructura de un carácter que se transmite asincrónicamente. Normalmente cuando no se realiza ninguna transferencia de datos, la línea del transmisor es pasiva (idle) y permanece en un estado alto. Para empezar a transmitir datos, el transmisor coloca esta línea en bajo durante el tiempo de un bit, lo cual se conoce como bit de arranque (start bit)

Y a continuación, empieza a transmitir, con el mismo intervalo de tiempo, los bits correspondientes al dato (que pueden ser 7 u 8 bits), empezando por el menos significativo (LSB), y terminando con el más significativo (MSB); al finalizar, se agrega el bit de paridad (parity bit), si es que está activada esta opción, y los bits de parada (stop bit) que pueden ser 1 ó 2, en los cuales la línea regresa a un

estado alto. Al concluir esta operación, el transmisor estará preparado para transmitir el siguiente dato.

Figura 7. Estructuras de un carácter que se transmite serialmente.

El receptor, mientras tanto, no está sincronizado con el receptor, y desconoce cuando va a recibir datos. La transición de alto a bajo de la línea del transmisor activa al receptor y este genera un conteo de tiempo de tal manera que realiza una lectura de la línea medio bit después del evento; si la lectura realizada es un estado alto, asume que la transición ocurrida fue ocasionada por ruido en la línea; si por el contrario, la lectura es un estado bajo, considera como válida la transición y empieza a realizar lecturas secuenciales a intervalos de un bit, hasta conformar el dato transmitido. El receptor puede tomar el bit de paridad para determinar la existencia o no de errores y realizar las acciones correspondientes, al igual que los bits de parada para situaciones similares.

Existen circuitos integrados especializados para manejar las comunicaciones asincrónicas, tales como el UART (Universal Asynchronous Receiver/Transceiver) 8250 de National Semiconductor. El UART, un elemento bidireccional toma datos de un bus paralelo para transmitirlos serialmente y toma datos seriales para colocarlos en el bus paralelo; este dispone de los registros necesarios para almacenar el estado de la comunicación, la velocidad, los bits de parada, el ancho de los datos, el dato recibido, el que se va a transmitir, etc.

los UART fueron desarrollados para manejar niveles lógicos TTL (0 – 5V), y son útiles en circuitos digitales donde las distancias son relativamente cortas; pero

cuando las distancias aumentan, estas señales tienden a degradarse debido al efecto capacitivo de los conductores y a su resistencia eléctrica. El efecto se hace más notorio a medida que se incrementa la frecuencia de la transmisión. Todo esto origina que los datos recibidos no sean iguales a los transmitidos, lo que no se puede permitir en un proceso de transferencia de datos.

Una de las soluciones más obvias en este tipo de situaciones es aumentar los márgenes de voltaje con la cual los datos se están transmitiendo, de tal manera que las perturbaciones causadas se puedan minimizar e incluso ignorar. Pero ya se imaginaran a una decena de fabricantes trabajando cada uno por su lado, para tratar de resolver este problema en particular? Si tres de las soluciones son las mismas, deja de ser una coincidencia.

Ante la gran variedad de equipos, sistemas y protocolos que se implementaron, surgió la necesidad de un acuerdo que permitiera que los equipos de varios fabricantes pudieran comunicarse entre sí. A principios de los años sesenta se desarrollaron varias normas que pretendían hacer compatibles los equipos, pero en 1962 se publicó la que se convirtió en la más popular: la norma RS-232. Esta norma define la interface mecánica, las características, los pines, las señales y los protocolos que debía cumplir la comunicación serial. La norma ha sufrido algunas revisiones, como la RS-232C en 1969 y la EIA/TIA-232E en 1991.

De todas maneras, las normas RS-232 cumplen básicamente con los mismos niveles de voltaje:

Un “1” lógico es un voltaje comprendido entre –5V y –15V en el transmisor y entre –3V y –25V en el receptor.

Un ¨0¨ lógico es un voltaje comprendido entre 5V y 15V en el transmisor y entre 3V y 15V en el receptor.

Estos niveles de voltaje son diferentes a los niveles TTL; por lo tanto, deben existir dispositivos que permitan convertir niveles TTL a niveles RS-232 y viceversa. Uno de los primeros dispositivos que se utilizaron fueron los drivers MC1488 y los receivers MC1489 de Motorola, de los que se desarrollaron versiones mejoradas como los SN75188 Y SN75189 de Texas Instrument y algunos similares de otros fabricantes. Todos los dispositivos anteriormente nombrados necesitan tres niveles de voltaje para su operación cuando el equipo actúa como transmisor y receptor, lo cual no representa ningún problema en computadores tipo PC, ya que se dispone de estos voltajes en la fuente. Pero cuando se trata con sistemas con microcontroladores, en los cuales el espacio es importante y no se puede disponer de voltajes diferentes a 5 voltios, estos circuitos integrados no se pueden utilizar. Para esto se han desarrollado alternativas muy útiles, como los integrados MAX 232.

Es preciso tener presente que la norma RS-232 fue desarrollada hace más de 30 años, época en la cual los requerimientos y capacidades de los equipos era diferente. En la actualidad, esta norma es un poco limitada, tanto para la distancia a la cual se puede transmitir, como para la frecuencia de la señal y el número de transmisores y receptores que pueden estar simultáneamente conectados. Existen otras normas para la comunicación serial, en la cual se incrementa el número de transmisores o receptores, la frecuencia de transmisión, la distancia, etc. Pero a

pesar de esto, los principios básicos siguen siendo los mismos de la comunicación asincrónica y de la interface RS-232.

1.5.1 C.I max 232. El envío de niveles lógicos (bits) a través de cables o líneas de transmisión necesita la conversión a voltajes apropiados. En un circuito lógico o con microprocesador se trabaja con niveles de voltaje inferiores a 0.8 para representar el valor lógico 0 y voltajes mayores a 2.0 para representar el valor lógico 1. Por lo general cuando se trabaja con familias TTL y CMOS se asume que un ¨0¨ es igual a cero voltios y un "1" a +5V.

Cuando la comunicación que se pretende hacer es muy corta, se pueden conectar directamente el transmisor y el receptor para hacer la transferencia de bits usando los mismos niveles lógicos tradicionales de 0y 5V. Pero cuando la distancia es mayor a los dos metros, la información digital se afecta notablemente por acción de la atenuación en el cable, el ancho de banda del mismo y la velocidad con que se transmita. La interface RS-232 es una de las diferentes soluciones que hay para esta situación. Básicamente consiste en cambiar los niveles lógicos de la salida o envío de 0 y 5V a dos niveles de voltaje de magnitud mayor: (+V) para representar el cero lógico Y negativo (-V) para representar el uno. En el equipo receptor de la información se realiza el proceso contrario, los niveles positivo y negativo que lleguen se convierten a los niveles lógicos tradicionales de 0 y 5 V.

En la práctica, los niveles de voltaje los determinan las fuentes de alimentación que se apliquen a los circuitos de la interface; los niveles más comunes son desde

+12V hasta +15V. Una interface RS-232 está compuesta por el circuito transmisor que convierte la señal de bajo voltaje del equipo lógico a los niveles de voltaje alto que se necesitan en la línea de transmisión y un receptor que realiza la función inversa. En los manuales de circuitos integrados se llama line drivers y line receivers, respectivamente, a los circuitos que ejecutan esta conversión de niveles de voltaje.

Por lo general, se utiliza con las interfaces RS-232 cable multipar o cable ribbon con un solo conductor como referencia de tierra. El ruido que se capta a través de la línea aún puede originar problemas. Para reducir el efecto se suele conectar un condensador en paralelo con la salida del circuito transmisor. Según la reglamentación los estándares de la interface RS-232 permiten una separación máxima a una velocidad de transmisión no mayor a 9.6 Kbps. Sin embargo se realizan conexiones a distancias mayores sin ningún problema.

Cuando se trabaja con la interface RS-232 surge la necesidad de dotar la fuente de alimentación de doble polaridad (+V Y –V) tanto al transmisor como al receptor. El MAX232 soluciona esta situación, este pertenece a la familia de line drivers/receivers para la interface RS-232, según la norma EIA-132E V.28/V.24, en aquellas aplicaciones donde no se dispone de fuentes dobles de +12V. El MAX232 necesita solamente una fuente de +5V para su operación; un elevador de voltaje interno, convierte el voltaje de +5V al de doble polaridad de +12V.

Como la mayoría de las aplicaciones de RS-232 necesitan de un emisor y un receptor, el MAX232 incluye en un solo empaque dos parejas completas de drivers y receivers, ver figura 8 distribución de pines del MAX 232.

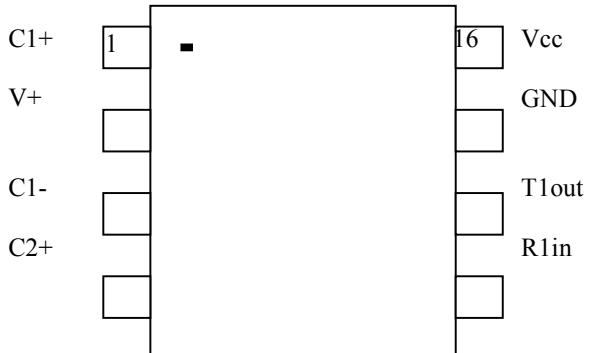


Figura 8. Distribución de pines del MAX 232

Las características de funcionamiento de este circuito integrado son las siguientes:

LIMITES:
Fuente de alimentación –0,3 a +6V
Voltajes de entrada
Tin -0.3V a (Vcc+0.3V)
Rin +30V

Voltajes de salida:
Tout +15V
Rout -0.3V a (Vcc +0.3V)

Protección corto continua
Disipación de potencia 842 mW

CARACTERISTICAS a Vcc=+5V,C1-C4=0,1 microfaradios

	Min.	Tip.	Máx.
TRANSMISOR			
Voltaje de salida (carga 3 Kohmios)	+5V	+8V	
Entrada BAJA		1.4V	0.8V
Entrada ALTA	2V	1.4V	
Velocidad		200 Kb/seg.	
RECEPTOR			
Rango de entrada			+30V
Entrada BAJA	0.8V	1.3V	
Entrada ALTA		1.8V	2.4V
Resistencia de entrada	3Kohmios		7 Kohmios

1.6 MODEMS

Cuando se planteó la necesidad de buscar un medio de transmisión que permitiera conectar dos equipos de datos muy alejados entre sí, se pensó en la red telefónica debido a su enorme difusión. Sin embargo, había un inconveniente que impedía la conexión directa entre los dos equipos: la red era analógica, y los datos, digitales. Se hizo necesario, por tanto, el desarrollo de un equipo que adaptara los datos digitales de forma que éstos pudieran transmitirse a través de un canal analógico telefónico. Este equipo se denominó módem. Ver figura 9.

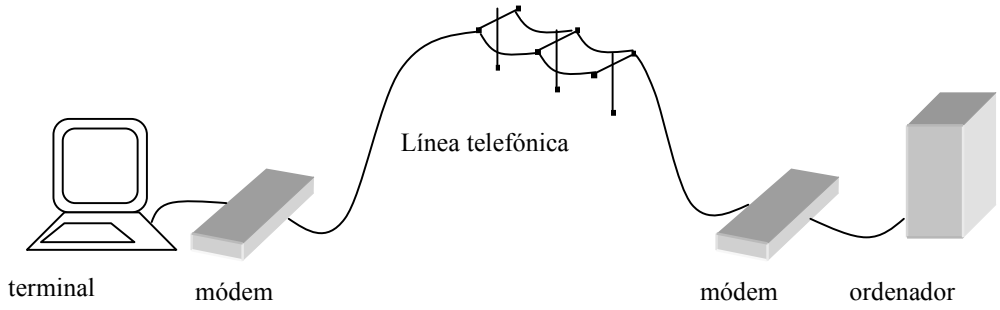


Figura 9. Conexión de equipo de datos a través de módem.

La palabra módem es una contracción de modulador-demodulador. Es fácil adivinar, por tanto, que la adaptación que realiza el módem consiste en la modulación de una portadora con los datos recibidos del terminal. El demodulador, por su parte, demodula los datos recibidos a través de la línea y procedentes de un terminal u ordenador remoto.

Hoy en día estos equipos han evolucionado tanto que el modulador-demodulador descrito anteriormente es solo una pequeña parte de lo que hoy de denomina módem. Los equipos actuales incluyen generadores de secuencias aleatorias, codificadores, ecualizadores, llamada y respuesta automática, facilidades de prueba, control de errores y un sin fin de nuevos elementos más que permiten continuamente aumentar las prestaciones y la velocidad del módem.

1.6.1 Modulación. Cuando se va a transmitir información digital, lo que transmitimos realmente es una señal analógica (portadora), la cual se le modifica una de sus características de acuerdo con la información binaria

que se pretende transmitir. La señal portadora (carrier) es normalmente una onda senoidal la cual está definida por tres características: amplitud máxima, frecuencia y fase. Si transmitimos la señal portadora sin ninguna modificación, estaremos transmitiendo una señal senoidal constante, la cual no transporta ninguna información.

Ahora bien, podemos transmitir una señal de frecuencia y fase constante, pero enviando dos amplitudes distintas, una para representar la información cero ”0” y otra para la información uno “1”. A este tipo de información se le llama Modulación de Amplitud o ASK (Modulación por salto de amplitud). Este tipo de modulación se emplea muy poco ya que es muy susceptible a interferencias en la línea y las velocidades de transmisión son bajas.

De la misma forma podríamos enviar una señal de amplitud y fase constante pero con dos frecuencias, llamada este tipo de modulación, modulación en frecuencia o FSK. Se utiliza una frecuencia determinada para representar la información cero “0” y otra frecuencia distinta para representar la información uno “1”. Este tipo de modulación se suele usar para velocidades de transmisión iguales o inferiores a 1200 bps.

El otro tipo de modulación es la modulación de fase, también conocida como PSK (Modulación por salto de fase) y consiste en mantener la frecuencia y la amplitud de la señal constante y modificar la fase en mas o menos grados dependiendo de la información binaria a transmitir. Esta modulación se utiliza para velocidades superiores a 1200 bps. Aunque en la mayoría de los casos para velocidades superiores se utilizan las modulaciones multinivel como son: DPSK(Modulación de fase diferencial), donde la portadora tiene cuatro estados (en este caso fases) diferentes, cada grupo de 2 bits se codifica como un cambio de fase. QAM(Modulación en cuadratura) donde se aplica una combinación de modulación de fase DPSK y de amplitud ASK

1.6.2 Velocidad de modulación. Baudio es una unidad de velocidad de modulación y viene dada en elementos de señal por segundo o en símbolos por segundo. Con las técnicas simples de modulación, el

número de baudios coincide con el número de bits por segundo, mientras que en las técnicas de modulación multinivel el número de baudios puede ser la mitad o la tercera parte del número de bits por segundo.

La relación entre la velocidad de transmisión serie en bps y la velocidad de modulación en baudio viene definida por la siguiente formula:

$$Vts= n*Vm$$

Donde;

Vts= velocidad de transmisión serie en bps

n= número de bits por baudios empleados en la modulación

Vm= velocidad de modulación

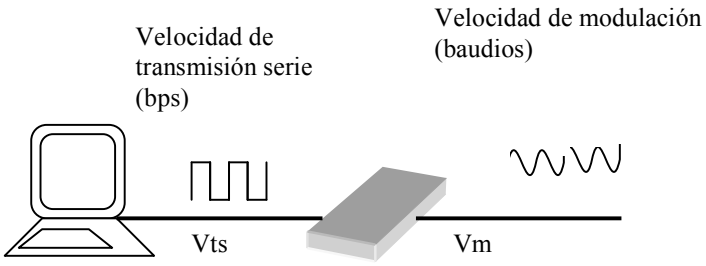


Figura 10. Velocidad de modulación

1.6.3 Modems inteligentes. La incorporación de los microprocesadores y de las memorias RAM, ROM Y EPROM a los módems los ha dotado de una gran variedad de características y funciones que han permitido de hacer del módem una herramienta más cómoda y eficaz entre todas las características incorporadas al módem, las más destacables son tres:

1. La incorporación de un juego de comandos
2. Capacidad de corrección y detección de error
3. Compresión de error

La incorporación de un juego de comandos permite, entre otras cosas, que el módem pueda realizar una marcación telefónica de forma automática o que se pueda seleccionar su modo de operación o realizar determinadas acciones mediante un simple comando. Dichos comandos son tecleados de forma manual por la persona que atiende la comunicación o de forma automática por el programa de comunicaciones.

La capacidad de corrección y detección de error descarga al software de comunicaciones de esta tarea, garantizando que la información intercambiada entre ambos módem está libre de errores.

Por último la compresión de datos consigue mediante una codificación especial disminuir la capacidad de bits de información a transmitir, con lo que en la practica supone un aumento de la velocidad de transmisión de información.

1.6.4 Detección y corrección de errores. Resulta muy extraño que se presenten errores de transmisión en la interfaz entre el ordenador y el módem, por lo que la utilización de técnicas de detección y corrección de errores en el software de comunicaciones solo supone introducir retrocesos innecesarios en la transmisión de información.

Las técnicas utilizadas por los módems en la corrección y detección de errores es muy valiosas pero la metodología es común:

1. Los datos transmitidos por el terminal, por el ordenador, al módem son empaquetados en bloques de caracteres.
2. A cada uno de estos bloques se le aplica un algoritmo para generar uno o más caracteres de redundancia, que son añadidos al final del bloque para su transmisión.
3. El módem receptor le aplica al mismo algoritmo a los bloques recibidos, teniendo que dar como resultado los mismos caracteres de redundancia. Si el resultado de la comparación es satisfactorio, el bloque se da por bueno y se envía al terminal; en caso contrario, se asume que el bloque contiene un error y se le indica al módem distante que retransmita el bloque erróneo.

Como podemos ver, la técnica de corrección y detección de error consiste en retransmitir el bloque erróneo. Eso quiere decir que todos los módems que dispongan de un procedimiento de detección y corrección de error deben tener una memoria intermedia(buffer) adecuada que les permita almacenar los bloques temporalmente

hasta recibir del módem distante la señal de aceptación o rechazo de los mismos. Una vez que el módem receptor acepta un bloque, el módem emisor puede descargar la memoria intermedia y utilizarla para el siguiente bloque.

Es evidente que la cantidad que la cantidad de memoria intermedia que puede contener un módem es limitada, lo cual quiere decir que si en un momento dado el módem receptor rechaza más bloques que los previstos, el emisor se verá desbordado por la información que le sigue llegando desde el terminal. Para evitar eso, se requieren de unos mecanismos que controlen el flujo de datos que el ordenador le envía al módem, y viceversa. A estos mecanismos se le conoce con el nombre de control de flujo.

1.6.5 Control de flujo. Las técnicas de control de flujo (flow control) compensan la diferencia de velocidad existente entre la llegada y salida de datos de un dispositivo. Por ejemplo, imagínese un módem que utiliza la técnica de compresión; el módem está conectado con el otro extremo a 9600 bps, pero como está comprimiendo la información con una relación 2:1, el terminal le envía 19600 bps. Supóngase que en un momento dado, la información que recibe el terminal no es tan susceptible de ser comprimida como la anterior, y por tanto no consigue llegar a la a la relación de compresión 2:1. En este momento, el módem estaría recibiendo más información del terminal de la que puede transmitir. Si no existieran técnicas de control de flujo, toda esa información que recibe el módem de más se perdería.

Todos los módems que utilizan técnicas de detección y corrección de error o técnicas de compresión están forzados, por un lado, a incorporar una memoria intermedia (buffer), y por otro, a utilizar técnicas de control de flujo. El control de flujo es la técnica que previene que se sature la memoria intermedia y que se pierdan datos.

El control de flujo fija dos niveles de ocupación de la memoria intermedia: cuando la memoria intermedia alcanza el nivel alto, el módem le indica al terminal que no siga enviando información, y cuando el nivel de ocupación de la memoria intermedia alcanza el nivel bajo, el módem le indica al terminal que reanude el envío de información.

El control de flujo no sólo se produce en la dirección terminal-módem, sino que también se utiliza en la dirección contraria.

Las técnicas básicas de control de flujo son las siguientes:

1. RTS/CTS

Cuando el ordenador se dispone a transmitir datos, le envía al módem una señal de petición de envío conocida como RTS (Request to send). Esta señal consiste en poner a 1 el contacto 4 de la interfaz RS 232. Si el módem está listo para transmitir responde con la señal de listo para enviar, CTS (Clear to send). Esta señal consiste en activar el contacto 5 de la interfaz RS232. El terminal no transmitirá datos al módem si no está activa la señal CTS. Eso quiere decir que el módem puede controlar el flujo de datos del terminal simplemente activando o desactivando la señal CTS. A este sistema de control de flujo también se le conoce por el nombre de control de flujo hardware (hardware flow control), debido a que el control se realiza por medio de un cable físico que une el ordenador con el módem.

2. XON/XOFF

Este se basa en la existencia de dos caracteres de control, XON Y XOFF, los cuales son utilizados por el módem para indicarle al terminal que detenga o reanude el envío de datos. Debido a que los caracteres XON Y XOFF son generados mediante el software, a este procedimiento también se le conoce como control de flujo software.

El carácter de control XOFF, utilizado por el módem para suspender el flujo de datos, se corresponde con el carácter ASCII 19 (CTRL-S), también es conocido como carácter DC3 (control de dispositivo 3). por su parte, el carácter de control XON permite reanudar el envío de datos. Este carácter se corresponde con el código ASCII 17 (CTRL-Q), y también es conocido como carácter DC1 (control de dispositivo 1).

El control de flujo software presenta varios inconvenientes. El primero es que enviar señales XON Y XOFF consume tiempo; esto es, mientras se envían las señales XON Y XOFF no se puede enviar datos, lo cual

disminuye el rendimiento. El segundo es que si los caracteres ASCII 17 Y 19 de los caracteres de control XOFF Y XON aparecen en los datos, el software tendrá que indicar alguna forma que se trata de información de datos y no de los caracteres de control de flujo, lo cual implica mas información redundante.

4. ENQ/ACQ

Este método consiste en que el terminal, antes de transmitirle datos al módem, le envía un mensaje ENQ(petición), a lo que el módem debe responder con un mensaje ACQ (aceptación). Cada vez que el terminal recibe un mensaje ACQ en respuesta a su mensaje ENQ, transmite un bloque de datos de aproximadamente 2000 caracteres. El módem controla el flujo con el mensaje ACQ.

1.6.6 Comandos Hayes. Los comandos Hayes consisten actualmente en un juego de comandos básicos y en una extensión de comandos. El juego de comandos básicos es común a todos los comandos Hayes y compatibles; sin embargo, el juego de comandos extendidos sólo es aplicable aquellos módem que dispongan de esas características o modos de operación. Eso quiere decir que si disponemos de un determinado software de comunicaciones y queremos estar seguros de que es capaz de aprovechar al máximo las características de nuestro módem, la única forma de tener certeza es comprobando que soporta el modelo específico de módem que estamos usando. En cualquier otro caso, la única seguridad será que el software compatible Hayes hará uso del juego de comandos básicos.

Todos los comandos Hayes comienzan siempre con un código de atención del módem, seguido del comando o comandos deseados. El código de atención es la secuencia de caracteres AT, la cual puede ser especificada tanto por caracteres en mayúsculas como en minúsculas. Esa característica ha hecho que los comandos Hayes también se le conozca como comandos AT.

para la recepción e interpretación de comandos, los modems Smartmodems de Hayes disponen de un buffers de 40 caracteres. Eso quiere decir que admite una línea de comandos de una longitud máxima de 40 caracteres. Hay que tener en cuenta que esos cuarenta caracteres no están incluidos ni el código de atención ni

los espacios entre comandos. Algunos modems compatibles Hayes disponen de un buffer de 80 caracteres, aunque la mayoría de los programas de comunicaciones solo transmiten un máximo de 40 caracteres para asegurarse la compatibilidad. Si se desea enviar comandos cuya suma total de caracteres sea mayor de 40, simplemente se envían dos líneas de comandos.

El formato básico para enviar comandos a un módem compatible Hayes es el siguiente:

AT comando[parámetros] comando[parámetros]... Intro

Como vemos, cada línea de comandos debe estar precedida de las letras AT, seguida por los comandos adecuados con sus correspondientes parámetros. Los parámetros de los comandos suelen ser dígitos numéricos y sirven para definir un estado especifico del comando(por ejemplo, H0 o H1). Toda línea de comandos debe terminar con el carácter retorno de carro (intro).

Cuadro 2. Comandos principales del juego de comandos básicos de Hayes

COMANDOS PRINCIPALES DEL JUEGO DE COMANDOS BÁSICOS	
COMANDO	DESCRIPCIÓN
AT	Atención. Debe preceder a todas las ordenes restantes.
A	Fuerza el modo respuesta (answer).
A/	Vuelve a ejecutar la última orden enviada.
B	Selecciona el tipo de modulación del módem.
C	Transmisor activado o desactivado.
D	Marca un número telefónico.
E	Habilita o inhabilita el eco de caracteres a la pantalla.
F	Conmuta entre dúplex y semidúplex.
H	Cuelga o descuelga el teléfono.
I	Petición del código de identificación o petición de la suma de control.
L	Selección del volumen del altavoz.
M	Activa o desactiva el altavoz.
N	Negociación de la velocidad.
O	Volver a conexión en línea.
P	Marcación de un número mediante el sistema de pulsos (decádico).
Q	Petición del módem para enviar o inhibir el envío de códigos de resultado.
R	Cambia las frecuencias del módem de originador a destinatario de la llamada.
S	Fija el valor del registro.
T	Marcación de un número mediante el sistema de tonos(multifrecuencia).
V	Devolver los códigos de resultado con palabras o números.
W	Selección de los mensajes de progreso de la negociación.
X	Uso del conjunto de códigos de resultado básico o extendido.
Y	Habilitar o inhabilitar la desconexión por espacios largos.
Z	Reinicializar el módem.
+++	Comando escape.

1.6.7 Modos de operación del módem. Los modems llamados inteligentes, o sencillamente, aquellos que son capaces de recibir órdenes y actuar en consecuencia, tienen cuatro modos de operación en cuanto a su disponibilidad a recibir órdenes. Estos modos de operación son:

1. Modo comando local
2. Modo negociación
3. Modo en línea

4. Modo comando en línea

Cuando encendemos el módem este siempre está dispuesto a recibir ordenes del terminal al que está conectado. Este es el modo de operación llamado modo comando local (local comand mode). El usuario puede transmitirle comandos al módem sencillamente tecleando los caracteres adecuados y enviándolos al puerto de comunicaciones donde está conectado el módem. Esta operación puede realizarla mediante simples comandos del DOS o mediante un programa de comunicaciones. Si se está utilizando un programa de comunicaciones, para transmitirle comandos al módem sólo hay que poner el programa en modo terminal o local y teclear AT seguido del comando correspondiente.

Cuando un módem realiza una llamada, el módem distante le envía una señal portadora después de descolgar, y a continuación entran en un proceso de negociación durante el cual entre los dos módems determinan los parámetros de la comunicación(entre otros, la velocidad de transmisión). Mientras el módem está en este estado de negociación, se dice que está en modo negociación(handshaking mode). Hay que decir que no todos los módems tienen la posibilidad de negociar sus parámetros con el módem remoto.

Una vez que han sido definidos todos los parámetros y se ha establecido la conexión, el módem solo se dedica a modular los datos que recibe del ordenador al que está conectado y de demodular los datos que recibe de la línea. Durante este tiempo se dice que el módem está en modo en línea (on-line mode). En este período, el módem no atiende a ningún parámetro, ya que entiende que todas las informaciones procedentes del terminal son datos que tiene que modular y enviar al terminal distante.

Se puede salir del modo comando en línea y volver al modo comando local cortando la comunicación. En modo local podemos volver a enviarle comandos al módem. No obstante, existe también la posibilidad de enviarle comandos al módem sin que para ello tengamos que cortar la comunicación. Eso se consigue con la llamada secuencia escape (+++). Si después de un período de inactividad, que debe ser mayor de un segundo, enviamos la secuencia de escape al módem, éste se sitúa de nuevo a la espera de recibir un comando si cortar

la conexión. A este estado se le llama modo comando en línea (on-line comand mode). A veces también se le llama modo comando fuera de línea.

1.6.8 Registros s del módem. Son una localización especial de memoria que dispone el módem para guardar determinados parámetros de operación, así como determinadas configuraciones especificas. Los valores de los registros S pueden ser consultados y modificados directamente por el usuario mediante el correspondiente comando AT. Por otro lado, los programas de comunicaciones suelen incorporar sus propios procedimientos para que el usuario pueda consultar y modificar estos valores de una forma más fácil e intuitiva.

El comando AT de consulta del valor de los registros S es ATSn?, mientras que el comando de modificación de dichos registros es ATSn=valor. Sí, por ejemplo, queremos saber cuántos segundos espera el módem el tono de llamada antes de realizar la marcación (registro S6) debemos enviar el siguiente comando al módem: ATS6?

A lo que el módem responderá con un valor entre 2 Y 255. Si quisiéramos fijar este valor en 4 segundos, el comando que debemos enviar al módem es: ATS6=4

Hay que tener en cuenta que no todos los módem disponen de los parámetros S, ni todos de los que disponen de los registros S tienen el mismo número de ellos. En la tabla 3 puede verse una relación de los registros S con una descripción de su significado, el rango de valores entre los que pueden configurarse y su valor por defecto.

Cuadro 3. Registros S del módem

REGISTROS S			
REGISTRO	DESCRIPCION	RANGO	DEFECTO
S0	Número de llamadas antes de descolgar.	0-255	0
S1	Cuenta del número de llamadas recibidas.	0-255	0
S2	Código ASCII del carácter escape.	0-127	43
S3	Código ASCII del carácter de retorno de carro, CR.	0-127	13
S4	Código ASCII del carácter de salto de línea, LF.	0-127	10
S5	Código ASCII del carácter de retorno (Backspace)	0-32,	8

		127	
S6	Tiempo de espera del tono antes de marcar (seg).	2-255	2
S7	Tiempo de espera de portadora antes de colgar (segundos).	1-255	50
S8	Tiempo de pausa causado por la coma (seg).	0-255	2
S9	Tiempo de respuesta de detección de portadora (en unidades de 1/10 de segundo).	1-255	5
S10	Tiempo de espera entre la perdida de portadora y el colgado (en unidades de 1/10 segundos).	1-255	14
S11	Duración del tiempo de tono y espaciado (en miliseg).	50-255	95
S12	Tiempo guarda de la secuencia de escape (en unidades de 20 miliseg).	0-255	50
S16	Prueba que se está realizando.	0-6	-
S18	Selección del temporizador de prueba.	0-255	0
S25	Tiempo de detección de cambio de DTR (en unidades de 1/100 seg).	0-255	5
S30	Temporización automática	0-255	0
S31	Selección del carácter XON.	0-255	17(DC1)
S32	Selección del carácter XOFF.	0-255	19(DC3)
S36	Tipo de negociación de bajada de velocidad automática (fallback)	0,1,3,4,5,7	7
S37	Máxima velocidad de línea del módem.	0-12,15,26,29,3334	0

Cuadro 4. Registros S del módem

REGISTROS S			
REGISTRO	DESCRIPCION	RANGO	DEFECTO
S49	Limite bajo del buffer (bytes).	1-249	64
S50	Limite bajo del buffer (bytes).	2-255	192
S69	Tamaño de la ventana de la capa de enlace.	1-15	15
S70	Número máximo de retransmisiones.	0-255	10
S71	Temporización de la capa de enlace (1/10seg).	1-255	2
S72	Temporización de perdida de bandera(segundos).	1-255	30
S73	Temporización por falta de actividad (segundos).	1-255	5
S82	Técnica de señalización de ruptura.	3,7,128	128
S86	Código del motivo del fallo de conexión.	0-19	-
S91	Ajuste del nivel de transmisión por red telefónica (Dbm)	0-15	10
S95	Opciones del mensaje de negociación.	1,2,4,8,32	0
S97	Tiempo de prueba del modo V22/V22 bis en	15-70	30

	V32(décimas de segundos).		
S105	Tamaño de trama V42 (octetos)	4-9	7
S108	Selector de calidad de señal.	0-3	1
S109	Selector de velocidad de portadora (valores decimales)	0-4094	4094
S110	Selector V32/V32 bis.	0-2	2
S113	Transmisión del tono de llamada (0-no, 1-si).	0-1	0

1.6.9 Circuitos principales de RS-232. Las señales que se intercambian entre el terminal y el módem en el proceso de una comunicación son las siguientes:

GND. Contacto 1. Tierra de protección (protective ground). Este contacto está generalmente conectado al mismo chasis del equipo, e incluso puede estar conectado a una señal de tierra externa. Esta señal también se puede utilizar para apantallar un cable protegido, de forma que se minimicen las interferencias producidas en los entornos con alto nivel de ruido. Hay que aclarar que la referencia común para todas las señales no es este contacto, sino el contacto 7.

SG. Contacto 7. Tierra de señal (Signal Ground). Este contacto es la referencia de todo el resto de las señales de la interfaz, incluidas las señales de datos, señal de reloj y señales de control. La tensión de esta señal siempre debe ser 0 voltios. En teoría, los contactos 1 y 7 deben ser independientes, pero en la practica frecuentemente están unidos formando una señal de tierra común.

TD. Contacto 2. Transmisión de datos (Transmited Data). Este circuito es utilizado para transmitir las señales de datos desde el equipo terminal (ETD) al módem (ETCD). cuando no se transmite ningún dato, este contacto debe mantener la señal lógica 1. Para que el terminal pueda transmitir datos por el contacto 2, los circuitos RTS, CTS, DSR Y DTR deben tener antes una tensión alta. Este contacto también se conoce como TXD.

RTS. Contacto 4. Petición de envío (Reques to Send). La señal de este circuito es enviada desde el terminal (ETD) al módem (ETCD) para preparar el módem para la transmisión. Una vez hecho esto, y antes de empezar a transmitir datos, el terminal debe recibir la señal CTS por el contacto 5. Ambas señales, RTS/CTS, también pueden ser utilizadas para controlar el flujo de datos entre el módem y el terminal. Para

que estas señales puedan ser reconocidas como indicadores de flujo de datos, tanto el módem como el software de comunicaciones deben ser configurados para mantener un control de flujo RTS/CTS, también llamado control de flujo hardware. Cuando un módem opera de forma asíncrona, el software de comunicaciones suele mantener la señal RTS constantemente en alto, indicando que el módem puede enviar datos al terminal en cualquier momento.

CTS. Contacto 5. Preparado para transmitir (Clear to Send). Este circuito se utiliza para indicarle al terminal que el módem está listo para transmitir. El módem activará esta señal después de que el terminal active su señal RTS. Este circuito también puede ser utilizado junto con RTS como control del flujo de datos entre el terminal y el módem. Al igual que con la señal RTS, para que CTS pueda ser reconocida como indicador de flujo de datos, tanto el módem como el software de comunicaciones deben ser configurados para mantener un control de flujo RTS/CTS.

CD. Contacto 8. Detección de Portadora (Carrier Detect). A este circuito también se le conoce con el nombre de detector de la señal de línea recibida, RLSD (Received Line Signal Detector), o como detección de portadora de datos, DCD (Data Carrier Detect). Una señal en este circuito le indica al terminal que el módem está recibiendo una señal de portadora del módem remoto. La señal de portadora tiene que estar presente durante todo el tiempo que dure la comunicación, se transmitan datos o no. Por tanto, si el terminal no detecta la señal CD, dará por terminada la comunicación por pérdida de portadora. En este caso el software de comunicaciones dará un mensaje similar a pérdida de portadora (Carrier Lost) para indicar esta condición. En el caso de que el módem disponga de indicadores luminosos, la presencia de esta señal también ilumina el indicador CD del módem.

RD. Contacto 3. Recepción de datos (Receive Data). Los datos que van demodulando el módem los envía al terminal por este contacto si el módem no tiene ningún dato que enviar al terminal, debe mantener este circuito en estado no activo (OFF, estado binario 1). A este contacto también se le conoce como RXD.

DSR. Contacto 6. Módem preparado (Data Set Ready). La señal de este circuito indica el estado del módem. Cuando este circuito está activo (valor lógico 0), indica que el módem está conectado a la línea telefónica y está listo para transmitir datos. Este contacto también puede ser utilizado por el módem para indicar que ha terminado un proceso de autorrevisión o que la marcación del número telefónico ha sido efectuada con éxito.

DTR. Contacto 20. Terminal de datos preparado (Data Terminal Ready). Cuando esta señal está activa, le indica al módem que el terminal está encendido y listo para una comunicación. Si la señal no está activa, el módem cortará cualquier comunicación que esté en curso. Este circuito controla, por tanto, la conexión del módem a la línea telefónica.

RI. Contacto 22. Indicador de llamada (Ring Indicator). Este circuito le indica al terminal que está siendo recibida una señal de llamada por el canal de comunicaciones. Este circuito es utilizado por aquellos modems que está en modo respuesta automática, para indicarle al terminal que se está recibiendo una llamada. En respuesta a esta señal de llamada, el terminal le pasa una tensión al contacto 20 (circuito DTR). esta tensión le dice al módem que descuelgue y atienda la llamada.

CG. Contacto 21. Detector de calidad (Quality Detector). Las señales de este circuito son transmitidas desde el módem al terminal siempre que el módem detecta una alta probabilidad de error en la recepción de los datos debido a una mala calidad de la línea. Este circuito permanecerá en estado activo cuando la calidad de la señal es aceptable, cambiando al estado no activo si la calidad es inadecuada.

CH/CI. Contacto 23. Selector de velocidad (Data Signal Rate Selector). Cuando el módem detecta una mala calidad de la línea y desactiva la señal CG, si este estado es mantenido durante un tiempo predeterminado, el terminal puede indicarle al módem que cambie su velocidad de operación por una más baja. Para hacer este cambio de velocidad se utiliza el contacto 23. El terminal pone el contacto 23 en estado activo para una velocidad de operación más elevada, y lo pone en estado no activo para una velocidad de operación más baja. Esta decisión de cambio de velocidad también puede ser tomada por el módem. Cuando

es el terminal quien selecciona la velocidad de operación, la señal del contacto 23 va del terminal al módem, y el circuito es conocido como circuito CH. Si es el módem quien determina la velocidad de operación, la señal del contacto 23 va del módem al terminal, y el circuito es conocido como circuito CI.

Cuadro 5. Conexiones RS 232

CONEXIONES RS-232				
NUMERO CONTACTO	IDENTIFICAC.	MNEMONICO	NOMBRE COMPLETO	ORIGEN DE LA SEÑAL
DATOS				
2	BA	TD	Transmisión de datos	ETD
3	BB	RD	Recepción de datos	ETCD
CONTROL DE FLUJO				
6	CC	DSR	Módem preparado	ETCD
20	CD	DTR	Terminal de datos preparado	ETD
4	CA	RTS	Petición de envío	ETD
5	CB	CTS	Preparado para transmitir	ETCD
LINEAS DE MODEM				
8	CF	CD	Detección de portadora	ETCD
22	CE	RI	Indicador de llamada	ETCD
TIERRA COMUN				
7	AB	SG	Tierra de señal	
CONEXIONES MENOS USADAS				
1	AA	GND	Tierra de protección	ETCD
12	SCF		Detección de portadora secundario	ETD
13	SCB		Preparado para transmitir secundario	ETD
14	SBA		Transmisión de datos secundario	ETCD
15	DB		Sincronismo en transmisión por ETCD	ETCD
16	SBB		Recepción de datos secundario	ETD
17	DD		Sincronismo en transmisión por ETD	ETCD
19	SCA		Petición de envío secundario	ETD
21	CG		Detector de calidad de la señal línea	ETD
23	CH		Selector de velocidad binaria	ETD
23	CI		Selector de velocidad binaria	ETCD
24	DA		Sincronismo en recepción	ETD

1.6.10 Proceso de comunicación. En el flujo de datos entre el ordenador (ETD) y el módem (ETCD) existen tres circuitos principales. El circuito 2, que es por donde circulan los datos del ordenador al módem (transmisión); el circuito 3, que es por donde circulan los datos del módem al ordenador (recepción), y el circuito 7, que es la señal de tierra a la que están referidas las tensiones de los circuitos anteriores.

Para que se produzca un intercambio de datos entre el ordenador (ETD) y el módem (ETCD), antes cada uno de ellos tiene que saber que el otro está conectado y listo para recibir los datos que pretende transmitir. Esto quiere decir que el ordenador (ETD) no iniciará ninguna acción si antes no comprueba que el circuito 6, DSR (Data Set Ready, módem listo), está en estado activo (valor binario 0). Este circuito le indica al ordenador (ETD) que el módem (ETCD) está conectado a la línea telefónica y está listo para transmitir datos. De la misma forma, el terminal le indica al módem que está preparado activando el circuito 20, señal DTR (Data Terminal Ready, terminal de datos activo). Hay que decir que algunos terminales (Software de comunicaciones) mantiene siempre activa la señal DTR, pero otros sólo la activan cuando reciben la señal de llamada por el contacto 22.

Una vez que el ordenador (ETD) ha comprobado que el módem (ETCD) está activo, pone el contacto 4, RTS (Reques to send, petición de envío), en estado activo para indicarle que a continuación le va a transmitir datos. A esta petición, el módem responde enviando el circuito 5, CTS (Clear to send, listo indicator, indicador de llamada). Si el terminal es de los que no siempre tiene activo el contacto 20, lo activará después de recibir la señal RI del módem. A continuación, el módem descuelga y el módem distante transmite un tono de portadora. El tono de portadora debe ser mantenido durante toda la comunicación, ya que será la señal que indique, por un lado, la continuidad de la conexión establecida entre los dos circuitos, y por otro, el hecho de que en el otro extremo sigue habiendo un equipo activo. Cuando el módem detecta la portadora, activa la señal CD (circuito 8), y no la desactiva hasta que la señal portadora no desaparezca al final de la comunicación. Una vez realizado este preámbulo, se lleva a cabo el intercambio de datos.

En el proceso de comunicación, los contactos 4(RTS) Y 5(CTS) son también utilizados para controlar el flujo de datos entre el terminal (ETD) y el módem (ETCD). tanto si el terminal o el módem se ven saturados, cada uno puede ser que el otro interrumpa temporalmente la transmisión desactivando la señal del circuito 4 o 5, respectivamente. Al activar de nuevo cualquiera de estos circuitos, se reanudará la transmisión.

Todo el proceso descrito anteriormente está de acuerdo con la norma RS-232, la cual establece un conector de 25 contactos en cada extremo, estando conectado cada contacto de un extremo con su idéntico en el otro extremo. No obstante, algunos modems utilizan un cable más simple, donde, por ejemplo, no existe la línea 5 y 8 del conector del PC. De la misma forma, también se ahorran los hilos 6 y 20, haciendo un puente entre ellos en ambos extremos. Con esta configuración no se controla el flujo pero funciona.

1.7 MICROCONTROLADOR (PIC 16C84)

Son numerosas las ocasiones en las que es necesario almacenar datos con el fin de que estos permanezcan a nuestra disposición, para leerlos o modificarlos, a pesar de que se presenten cortes de energía o que se desconecte el sistema. Han sido innumerables las soluciones a esta necesidad, y ellas van desde la memoria RAM, alimentadas por baterías de litio para brindar alimentación de respaldo en caso de cortes de energía, hasta dispositivos magnéticos para almacenar la información; más recientemente, las memorias paralelas EEPROM han sido una solución más eficiente. Como se recordará, las memorias EEPROM son borrables y programables eléctricamente; los datos permanecen inalterados, a menos que se sobre escriba sobre ellos, utilizando los protocolos adecuados.

Con todo lo anterior, las soluciones planteadas tienen un inconveniente grande: el dispositivo que se utiliza como memoria de datos se encuentra por fuera del circuito integrado del procesador; por lo tanto se necesita, además de espacio adicional para éste, diseñar la disposición de las líneas d e control, de datos, de direcciones, etc., lo que involucra necesariamente inconvenientes, tiempo y, por supuesto, dinero.

Este microcontrolador resuelve con creces la necesidad planteada, ya que tiene incorporada una memoria EEPROM de datos de 64 posiciones, cada una de 8 bits; aquí, no se precisan baterías de respaldo para mantener los datos, ni espacio

adicional, ni trazar líneas complejas entre la CPU y la memoria; todo está inmerso en el mismo circuito integrado.

Este microcontrolador es el segundo miembro de una familia extendida y está muy emparentado con el PIC16C71, más que con los PIC16C5X, y tiene con éste las siguientes características comunes:

Solo 18 pines, de los cuales 13 son de entrada/salida, con control individual de dirección

Conjunto de instrucciones de sólo 35 elementos

Memoria de programas de 1024 posiciones, de 14 bits cada una

36 registros de propósito general de 8 bits cada uno (SRAM)

15 registros especiales de hardware

Pila de 8 niveles

Cuatro fuentes de interrupción

Capacidad de corriente para manejar LEDs directamente

Temporizador / contador de 8 bits con preescalador programable de 8 bits

Circuito de vigilancia (Watchdog Timer) incorporado

Cuatro opciones para el oscilador

Seguridad para protección del código del programa

Modo de consumo de baja corriente

Programación en serie o en paralelo. Esta opción permite utilizar solamente dos líneas para transmitir los códigos correspondientes al programa.

1.7.1 Estructura interna. Esta se basa en la arquitectura harvard, en la cual el programa y los datos se accesan desde memorias separadas, lo que posibilita que las instrucciones y los datos posean longitudes diferentes. Esta misma estructura es la que permite la superposición de los ciclos de búsqueda y ejecución de las instrucciones, lo cual se ve reflejado en una mayor velocidad del microcontrolador.

1.7.2 Organización de la memoria de programa. Debido a que el PIC 16C84 tiene un contador de programa de 13 bits, tiene una capacidad de direccionamiento de 8K * 14, pero solamente tiene implementado el primer 1K * 14 (0000h hasta 03FFh). Accesos a posiciones superiores a 3FFh causarán un solapamiento con el espacio del primer 1K. El vector de reset se localiza en la dirección 0000h, mientras que el de interrupciones en la 0004h, como se puede observar en la figura 11.

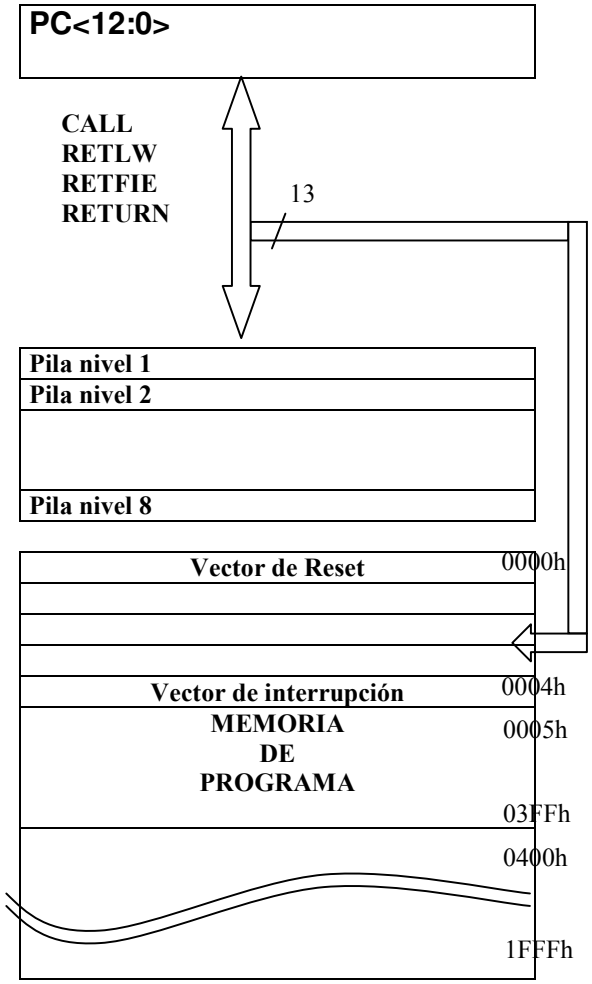


Figura 11. Memoria de programa.

1.7.3 organización de los registros. Los registros están organizados como arreglos (páginas) de 128 posiciones de 8 bits cada una (128*8), como se puede observar en la figura 12; todas las posiciones se pueden accesar directamente o indirectamente (a través del registro selector FSR). Estos registros de datos conforman la memoria del microcontrolador. Para paginar los registros, existen dos bits dentro del registro STATUS que se especializan en ello. Cada página tiene implementados únicamente los primeros 48 registros (2Fh). Las primeras 12

posiciones en cada página, contienen registros de función especial. De acuerdo con la figura 12 y considerando que todos los registros físicos se pueden leer o modificar, la siguiente es la descripción de estos:

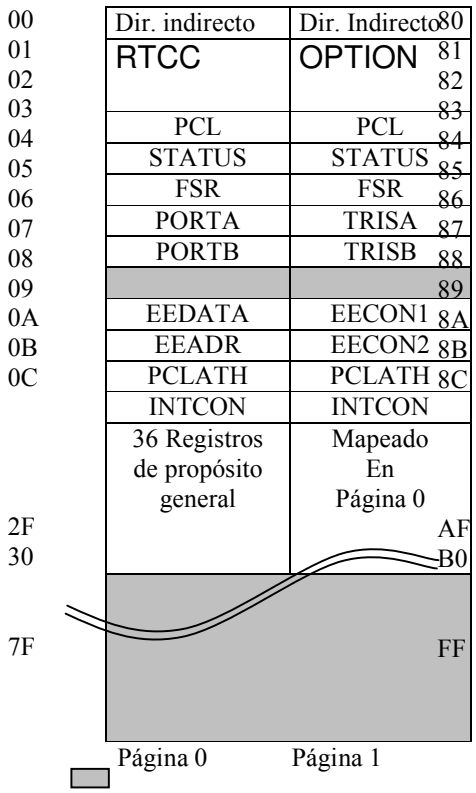
00h o INDO: Registro para direccionamiento indirecto de datos. Este no es un registro disponible físicamente; utiliza el contenido del FSR y el bit IRP del registro STATUS para seleccionar indirectamente la memoria de datos o RAM del usuario; la instrucción determinará que se debe realizar con el registro señalado.

01h o TMRO: Contador y reloj de tiempo real. Temporizador/contador de 8 bits. Este se puede incrementar con una señal externa aplicada al pin RA4/TOCK1 o de acuerdo a una señal interna proveniente del reloj de instrucciones del microcontrolador. La rata de incremento del registro se puede determinar por medio de un preescalador, localizado en el registro OPTION. Como una mejora con respecto a sus antecesores, se le ha agregado la generación de interrupción cuando se rebasa la cuenta (el paso de 0FFh a 00h).

02h o PCL:Contador de programa. Se utiliza para direccionar las palabras de 14 bits del programa del usuario que se encuentra almacenado en la memoria ROM; este contador de programa es de 13 bits de ancho. Sobre el byte bajo, PCL se puede escribir o leer directamente, mientras que sobre el byte alto, no. El byte alto se puede escribir mediante el registro PCLATH(0Ah). A diferencia de los PIC 16C5X, el 16C84 ante una condición de reset inicia el contador de programa con todos sus bits en ¨cero¨. Durante la ejecución normal del programa, y dado que

todas las instrucciones ocupan sólo una posición de memoria, el contador se incrementa en uno con cada instrucción, al menos que se trate de alguna instrucción especial.

En una instrucción CALL o GOTO, las posiciones PC<10:0> se cargan desde el código de operación de la instrucción, mientras que las posiciones PC<11:12> lo hacen desde el PCLATH<4:3>. Como solamente 1K está implementado, el código de operación de la instrucción puede contener la dirección destino. En otras instrucciones donde PCL es el destino, PC<12:8> se carga directamente desde el PCLATH<4:0>.



Posiciones no implementadas

Figura 12.Organización de los Registros

03h o STATUS: Registro de estados. Contiene el estado aritmético de la ALU, la causa del reset y los bits de preselección de página para la memoria de datos. La figura muestra los bits correspondientes a este registro. A diferencia con sus antecesores, los bits 5 y 6 (RP0 Y RP1) son los bits de selección de página para el direccionamiento directo de la memoria de datos (en los PIC16C5X estos bits seleccionan las páginas de la memoria de programa); solamente RP0 se usa en los PIC16C84. RP1 se puede utilizar como un bit de propósito general de lectura/escritura. Los bits TO Y PD no se pueden modificar por un proceso de escritura; ellos muestran la condición por la cual se ocasionó el último reset.

IRP	RP1	RP0	T0	PD	Z	DC	C

MSB LSB

Dirección: 03h

Condición de reset: 000??XXXb

IRP: Selector de página para direccionamiento indirecto, este bit no se utiliza efectivamente en el PIC16C84, por lo que se puede utilizar como un bit de propósito general.

RP1,0: Selectores de página para direccionamiento directo. Solamente RP0 se utiliza en el PIC16C84. RP1 se puede utilizar como un bit de propósito general.

T0: Time out o bit de finalización del temporizador. Se coloca en cero cuando el circuito de vigilancia Watchdog finaliza la temporización.

PD: Power Down o bir de bajo consumo. Se coloca en cero por la instrucción SLEEP.

Z: Zero o bit de cero. Se coloca en 1 cuando el resultado de una operación lógica o aritmética es cero.

DC: Digit Carry o bit de acarreo de dígito. En operaciones aritméticas se activa cuando hay acarreo entreb el bit 3 y el 4.

C: Carry o bit de acarreo. En instrucciones aritméticas se activa cuando se presenta acarreo desde el bit más significativo del resultado.

Figura 13. Registro status

04h o FSR: Registro selector de registros. En asocio con el registro INDO, se utiliza para seleccionar indirectamente los otros registros disponibles. Mientras que los antecesores poseían sólo 5 bits activos, en un microcontrolador se poseen los 8 bits. Si en el programa no se utilizan llamados indirectas, este registro se puede utilizar como un registro de propósito general.

05h o PORTA: Puerto de entrada/salida de 5 bits. Este puerto, al igual que todos sus similares en los PIC, puede leerse o escribirse como si se tratara de un registro cualquiera. El registro de control de este puerto está localizado en la página 1, en la posición 85h.

06h o PORTB: Puerto de 8 bits. Al igual que en todos los PICs, este puede leerse o escribirse como si se tratara de un registro cualquiera; algunos de sus pines tienen funciones alternas en la generación de interrupciones. El registro de control para la configuración de la función de sus pines se localiza en la página1, en la dirección 86h.

08h o EEDATA: Registro de datos. Este registro contiene el dato que se va a escribir en la EEPROM o el que se leyó de ésta.

09h o EEADR: Registro de la dirección. Aquí, se mantiene la dirección de la posición de la EEPROM a ser accesada, bien sea para una operación de lectura o para una de escritura.

0Ah o PCLATH: Registro para la parte alta de la dirección. Este contiene la parte alta del contador de programa y no es directamente accesible.

0Bh o INTCON: Registro para el control de interrupciones. Es el encargado del manejo de las interrupciones. y contiene los bits que se muestran en la figura 14.

GIE	EEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF
MSB							LSB

Dirección: ***0Bh***

Condición de reset: ***0000000XB***

GIE: Global Interrupt Enable o Habilitador general de interrupciones.
0: deshabilita todas las interrupciones
1: habilita las interrupciones

EEIE: EEPROM Write Interrupt Enable o habilitación de interrupción por escritura de la EEPROM.
0: la deshabilita
1: la habilita

TOIE: TMRO Interrupt Enable o Habilitación de interrupción del temporizador TMRO
0: la deshabilita
1: la habilita

INTE: INT Interrrupt Enable o Habilitación de interrupción INT.
0: la deshabilita
1: la habilita

RBIE: RBIF Interrupt Enable o Habilitación de interrupción RBIF.
0: la deshabilita
1: la habilita

TOIF: TMRO Overflow Interrupt Flag o bandera de la interrupción por sobrepasamiento del TMRO.
Se coloca en 1 cuando el TMRO pasa de 0FFh a 00h; ésta debe ser puesta a cero por Programa.

INTF: INT Interrupt Flag o Bandera de interrupción INT.
Se coloca en 1 cuando la interrupción INT (RB<0>) ocurre; ésta debe ser puesta a Cero por programa.

RBIF: RB Port Change Interrupt Flag o Bandera de interrupción por cambio en el puerto RB.
Se coloca en 1 cuando una de las entradas RB<7:4>cambia; ésta debe ser puesta a Cero por programa.

Figura 14. Registro INTCON.

81h u OPTION: Registro de configuración múltiple. Posee varios bits para configurar el preescalador, la interrupción externa, el timer y las características del puerto B. Los bits contenidos se muestran en la figura 15. El preescalador es compartido entre el RTCC y el WDT; su asignación es mutuamente excluyente, ya que solamente puede uno de ellos ser preescalado a la vez.

RBPU	INTEDG	RTS	RTE	PSA	PS2	PS1	PS0
MSB							LSB

Dirección: 81h
Condición de reset: 11111111b

RBPU: PortB Pull-up Enable o Habilitación de Pull-up del puertoB.
0: habilita la Pull-ups internas
1: las deshabilita

INTEDG: INT Interrupt Edge Selector o Selector de flanco de la interrupción INT.
0: flanco de bajada
1: flanco de subida

RTS: TMRO Signal Source o Fuente de la señal de TMRO.
0: ciclo de instrucciones interno (temporizador)
1: transición en el pin RA4/TOCK1 (contador)

RTE: TMRO Signal Edge o flanco de la señal TMRO
0: incremento en transición de bajo a alto
1: incremento en transición de alto a bajo

PSA: Prescaler Assignment o asignación del preescalador
0: TMRO (contador / temporizador)
1: WDT (circuito de vigilancia)

PS2,1,0: Prescaler Value o valores del prescalador.

Figura 15. Registro OPCION.

85h o TRISA: Registro de configuración del puerto A. Como ya se mencionó, es el registro de control para el puerto A. Un ¨cero¨ en el bit correspondiente al pin lo configura como salida, mientras que un ¨uno¨ lo hace como entrada.

86h o TRISB: Registro de configuración del puertoB. Orientado hacia el control del puerto B. Son válidas las mismas consideraciones del registro anterior.

88h o EECON1: Registro para el control de la memoria de datos y sólo destina cinco bits para ello, los más bajos; los tres bits superiores permanecen sin implementar. En la figura 16 se muestran las funciones de estos bits.

U	U	U	EEIF	WRERR	WREN	WR	RD
MSB							LSB

Dirección: 88h
Condición de reset: 0000X000b

U: Unimplemented No implementadas.
Estos bits se leen como ceros.

EEIF: EEPROM Write Completion Interrupt Flag o bandera de finalización de la escritura. Se coloca en ¨uno¨cuando finaliza con éxito la escritura en la EEPROM de datos; se debe colocar en ¨cero¨ por programa. El bit de habilitación correspondiente es el EEIE, localizado en el registro INTCON (0B<6>).

WRERR: Write Error Flag o bandera de error de escritura. Se coloca en 1 cuando la operación de escritura termina prematuramente, debido a cualquier condición de reset.

WREN: Write Enable o habilitación de escritura. Si se coloca en ¨cero¨ no permite las operaciones de escritura; en ¨uno¨ las habilita.

WR: Write Control o control de escritura.
Al colocarse en ¨uno¨ inicia un ciclo de escritura. Este bit sólo es puesto a ¨cero¨ por hardware, una vez la escritura termina.

RD: Read Control o control de lectura.
Al colocarse en ¨uno¨ se inicia una lectura de la EEPROM de datos, la cual toma un ciclo de reloj de instrucciones. Este bit sólo se limpia (se coloca en ¨cero¨) por hardware, al finalizar la lectura de la posición de la EEPROM.

Figura 16. Registro EECON1.

89h o EECON2: Registro auxiliar para control de la memoria de datos. Registro que no está implementado físicamente en el microcontrolador, pero que es necesario en las operaciones de escritura en la EEPROM de datos; ante cualquier intento de lectura se obtendrán ¨ceros¨.

0Ch a 2Fh: Registros de propósito general. Estas 36 posiciones están implementadas en la memoria RAM estática, la cual conforma el área de trabajo del usuario; a ellas también se accede cuando en la página 1 se direccionan las posiciones 8Ch a Afh. Esto se ha diseñado así para evitar un excesivo cambio de páginas en el manejo de la RAM del usuario, agilizando los procesos que se estén llevando a cabo y descomplicando la labor del programador. Otro registro que se debe tener en cuenta, es el registro de trabajo W, el cual realiza las funciones de un acumulador de otros procesadores. Este registro participa en la mayoría de las instrucciones.

La pila. Aunque la pila no se considera un registro, debemos hablar de ella. La pila no hace parte de los mapas de memoria de programa o de datos y su puntero no está disponible para el programador. Los antecedentes del PIC16C84 o del 71 poseían únicamente dos niveles de pila, lo cual permitía realizar el anidamiento de dos rutinas, sin que se presentaran problemas de pérdida del programa; aquí se han aumentado los niveles, hasta llegar a ocho.

Circuito de vigilancia. Este funciona con su propio oscilador RC, el cual no requiere componentes externos; esto permite que el WDT continúe corriendo, aún si el reloj del microcontrolador se detiene, por ejemplo por una instrucción SLEEP. El rebase del conteo del WDT genera en el dispositivo una condición de reset, el cual se puede evitar borrando periódicamente, por programa, la cuenta del WDT. El circuito de vigilancia se puede deshabilitar permanentemente programando el fusible WDTE como 0.

El período nominal del WDT es de 18ms. Este tiempo varía con la temperatura, el voltaje de alimentación y las condiciones en el proceso de fabricación de las partes. Si se requieren períodos más largos, se puede utilizar el preescalador del registro OPTION, logrando períodos hasta de 2,3seg.

1.7.4 Memoria de datos EEPROM. El PIC16C84 tiene una memoria EEPROM de datos de 64 posiciones, de 8 bits cada una. El acceso a estas posiciones se consigue a través de dos registros de la RAM:

El registro EEADR (posición 09), que debe contener la dirección de la posición de la EEPROM a ser accesada.

El registro EEDATA (posición 08), que contiene el dato de 8 bits que se va a escribir o el que obtuvo de la última lectura.

Adicionalmente, existen dos registros de control: el EECON1 (88h), que posee cinco bits que manejan las operaciones de lectura/escritura y el EECON2 (89h), que aunque no es un registro físico, es necesario para realizar las operaciones de escritura.

La lectura toma un ciclo del reloj de instrucciones; mientras que la escritura, por se controlada por un temporizador incorporado y requerir una operación previa de borrado de la posición de interés, tiene un tiempo nominal de 10 milisegundo; este tiempo puede variar con la temperatura y el voltaje. Según el fabricante, el número típico de ciclos de borrado/escritura de la EEPROM de datos es de 1.000.000.

1.7.5 Interrupciones. Este microcontrolador incluye el manejo de interrupciones lo cual trae, sin lugar a dudas grandes ventajas, el PIC16C84 posee cuatro fuentes de interrupción:

- Interrupción externa
- Finalización de temporizador/contador
- Finalización de escritura en la EEPROM de datos
- Cambios en las líneas RB4 a RB7

El registro 0B4 o INTCON contiene las banderas de las interrupciones INT, cambio en el puerto B y sobrepasamiento del TIMER, al igual que el control para habilitar o deshabilitar cada una de las fuentes de interrupción, incluida la de escritura. Sólo la bandera de finalización de la escritura reside en el registro 88h (EECON1<4>).

Si el bit GIE (Global Interrupt Enable) se coloca en 0, deshabilita todas las interrupciones. Cuando una interrupción es atendida, el bit GIE se coloca en 0 para deshabilitar otras interrupciones que se pueden presentar, la dirección de retorno se coloca en la pila y el PC se carga con la dirección 04h. Una vez en la rutina de servicio, la fuente de la interrupción se puede determinar examinando las banderas de interrupción. La bandera respectiva se debe colocar, por software, en cero antes de regresar de la interrupción, para evitar interrupciones recursivas.

La interrupción RETFIE permite al usuario retornar de la interrupción, a la vez que habilita de nuevo las interrupciones, al colocar el bit GIE en uno. Debe tenerse presente que solamente el contador de programa es puesto en la pila al atenderse la interrupción; por lo tanto, es conveniente que el programador tenga cuidado con el registro de estados y el de trabajo, ya que se pueden producir resultados inesperados si dentro de ella se modifican.

Interrupción externa. La interrupción externa actúa sobre el pin RB0/INT y se puede configurar para activarse con el flanco de subida o el de bajada, de acuerdo al bit INTEDG (OPTION<6>). Cuando se presenta un flanco válido en el pin INT, la bandera INTF (INTCON<1>) se coloca en uno. La interrupción se puede deshabilitar colocando el bit de control INTE (INTCON<4>) en cero; cuando se atiende la interrupción, a través de la rutina de servicio, INTF se debe colocar en cero por software, antes de rehabilitar la interrupción. La interrupción puede reactivar al microcontrolador después de la interrupción SLEEP, si previamente el

bit INTE fue habilitado. El estado del bit GIE decide si el procesador salta o no al vector de interrupción después de haberse reactivado.

Interrupción por finalización de temporización. La superación del conteo máximo (0FFh) en el TMRO colocará el bit T0IF en uno (INTCON<2>). El bit de control respectivo es TOIE (INTCON<5>).

Interrupción por cambio en el puerto RB. Un cambio en la entrada en el puerto B<7:4> colocará en uno el bit RBIF(INTCON<0>). El bit de control respectivo es RBIE (INTCON<3>).

Interrupción por finalización de la escritura. Cuando la escritura de un dato en la EEPROM finaliza, se coloca en uno el bit EEIF (EECON1<4>). El bit de control respectivo es EEIE (INTCON<6>).

1.7.6 Opciones del oscilador. Al igual que sus predecesores, el PIC16C84 posee cuatro opciones para el oscilador; XT o cristal, LP o de baja potencia, el HS o de alta velocidad y el RC u oscilador excitado por una red de resistencia y condensador. Cualquiera de las cuatro opciones del oscilador puede elegirse a través de unos fusibles destinados para este propósito.

1.7.7 Fusibles EPROM. **El PIC16C84 posee cinco fusibles, cada uno de los cuales es un bit. Estos fusibles se pueden programar para seleccionar varias configuraciones del dispositivo: tipo de oscilador, protección de código,**

habilitación del circuito de vigilancia y el temporizador al encendido. Los bits se localizan en la posición de memoria 2007h, posición a la cual el usuario sólo tiene acceso durante la programación del microcontrolador. Cuando se programa la protección de código, el contenido de cada posición de la memoria no se puede leer completamente, de tal manera que el código del programa no se puede reconstruir. Adicionalmente, todas las posiciones de memoria del programa se protegen contra la programación.

Una vez protegido el código, el fusible de protección sólo puede ser borrado (puesto a 1) si se borra toda la memoria del programa y la de datos.

1.7.8 Las pull-ups internas. Cada uno de los pines del puerto B tiene un débil elemento pull-up interno (aprox 250 micro A típico); este elemento es automáticamente desconectado cuando el pin se configura como salida. Adicionalmente, el bit RBPU (OPTION<7>) controla todos estos elementos, los cuales están deshabilitados ante una condición de reset. Estos elementos pull-up son especialmente útiles cuando el microcontrolador va a colocarse en el modo de bajo consumo, ya que ayuda a no tener las entradas flotantes, significando una reducción en el consumo de corriente.

1.7.9 Condición de reset. El PIC16C84 admite diferentes tipos de reset:

- Al encendido (POR)
- MCLR durante la operación normal

- MCLR durante el modo de bajo consumo
- El rebase del conteo del temporizador WDT durante la operación normal
- El rebase del conteo del temporiazador WDT durante el modo de bajo consumo.

El reset al encendido se consigue gracias a dos temporizadores. El primero de ellos es el oscilador Start-Up Timer OST, orientado a mantener el microcontrolador en reset hasta que el oscilador del cristal es estable. El segudo es el Power-Up Timer PWRT o temporizador al encendido, que provee un retardo fijo de 72ms (nominal) en el encendido únicamente, diseñado para mantener la parte en reset mientras la fuente se estabiliza. Para utilizar estos temporizadores, sólo basta con conectar el pin MCLR a Vdd, evitándose utilizar las tradicionales redes RC externas del reset. El reset por MCLR se consigue llevando momentáneamente este pin a un estado lógico bajo, mientras que el WDT produce el reset cuando el temporizador rebasa su cuenta, o sea que pasa de 0FFh a 00h.

El modo permite que el microcontrolador exija muy bajo consumo de corriente, lo que se traduce en un ahorro significativo de energía. A este modo se ingresa después de la ejecución de una instrucción SLEEP. Si está habilitado, el circuito de vigilancia WDT se limpiará, pero permanecerá corriendo. El bit PD se colocará a 0, el bit TO en 1 y el oscilador se detendrá. Los puertos conservan la condición que tenían antes de la ejecución de la instrucción. Para el más bajo consumo en este modo, todos los pines de entrada/salida deben estar conectados a 5 voltios o a tierra, evitando que

queden entradas flotantes. La entrada RTCC también debe estar a Vdd, y el pin MCLR a un nivel lógico alto. La recuperación de este modo es posible a través del reset, de la terminación del conteo del circuito de vigilancia o de la generación de una interrupción.

1.7.10 El conjunto de instrucciones. Estas se clasifican en orientadas a registros, orientadas al bit y operaciones literales y de control. Cada instrucción es una palabra de 14 bits, dividida en un código de operación, el cual especifica la orden a ejecutar y uno o más operandos sobre los que se actúa. En la FIGURA KKK se muestra el conjunto de instrucciones de estos microcontroladores. Como se observa en total son 35 las instrucciones, las cuales tardan un ciclo de máquina, excepción hecha en los saltos, que toman dos ciclos como sucedía con el PIC16C71, el 16C84 es una versión mejorada de los PIC 16C5X y los programas escritos para estos últimos se pueden trasladar fácilmente a éste, con pocos cambios.

Para quienes están familiarizados con los PIC16C5X encontrarán básicamente las siguientes modificaciones:

- La longitud de las instrucciones se inctrementó a 14 bits, lo cual permite mayor paginado de memoria de programa y de registros.
- El latch del contador de programa (PCLATH) es más ancho, lo que permite omitir la paginación de la memoria de programa.

- La paginación de la memoria de datos se ha redefinido ligeramente, de tal manera que se elimina la necesidad de los bits PA2,PA1 Y PA0 en el registro de estados.
- Se cuenta con cuatro nuevas instrucciones: RETURN, RETFIE, ADDLW y SUBLW.
- La manera en que se configuran los puertos (con la instrucción TRIS) y se asignaba el preescalador (con la instrucción OPTION) ha sido modificada de tal forma que los registros OPTION y TRIS ahora son direccionables; aún así, se conservan como in strucciones para mantener la compatibilidad con los PIC16C5X.
- Se agregaron capacidades de interrupción. El vector de interrupción es 004h.
- El tamaño de la pila se incrementó a 8 niveles.
- El vector de reset se modificó a 0000h.
- Reactivación despues de la instrucción SLEEP a través de interrupciones.
- El pin de entrada RTCC es ahora un pin del puerto A.
- El ancho del registro FSR se aumentó a 8 bits.
- La posición 07 no está implementada.
- Posibilidad de programar el microcontrolador en el sistema sobre el cual esté funcionando, utilizando solamente cinco pines.

Cuadro 6. Conjunto de instrucciones de los PIC 16C84

Operaciones orientadas a registros			
Nemotécnico	Operación	Cód. de operación msb lsb	Estados afectados
ADDWF f,d	Sumar w y f	00 0111dfff ffff	C,DC,Z
ANDWF f,d	AND entre w y f	00 0101dfff ffff	Z
CLRF f	Limpiar f	00 00011fff ffff	Z
CLRW -	Limpiar w	00 00010xxx xxxx	Z
COMF f,d	Complementar f	00 1001dfff ffff	Z
DECF f,d	Decrementrar f	00 0011dfff ffff	Z
DECFSZ f,d	Decrementar f , saltar si cero	000 1011dfff ffff	
INCF f,d	Incrementar f	00 1010dfff ffff	Z
INCFSZ f,d	Incrementar f, saltar si cero	00 1111dfff ffff	
IORWF f,d	**Or entre w y f**	00 0100dfff ffff	Z
MOVF f,d	Mover f	00 1000dfff ffff	Z
MOVWF f	Mover w a f	00 00001fff ffff	Z
NOP -	No operación	00 00000xx0 0000	
RLF f,d	Rotar a la izquierda a través del carry	00 1100dfff ffff	C
RRF f,d	Rotar a la derecha a través del carry	00 1100dfff ffff	C
SUBWF f,d	Restar w de f	00 0010dfff ffff	C,DC,Z
SWAPF f,d	Intercambiar nibbles de f	00 1110dfff ffff	
XORWF f,d	Or exclusiva entre w y f	00 0110dfff ffff	Z
Operaciones orientadas a bits			
BCF f,b	Limpiar bit b de f	01 00bbbfff ffff	
BSF f,b	Activar bit b de f	01 01bbbfff ffff	
BTFSC f,b	Probar bit b de f, saltar si es cero	01 10bbbfff ffff	
BTFSS f,b	Probar bit b de f, saltar si es uno	01 11bbbfff ffff	
Operaciones literales y de control			
ADDLW k	Sumar literal k a w	11 111xkkkk kkkk	C,DC,Z
ANDLW k	AND entre k y w	11 1001kkkk kkkk	Z
CALL k	Llamar subrutina	10 0kkkkkkk kkkk	
CLRWDT -	Limpiar WDT	00 00000110 0100	TO,PD
GOTO k	Salta a dirección k	10 1kkkkkkk kkkk	
IORLW k	OR entre k y w	11 1000kkkk kkkk	Z
MOVLW k	Cargar a w con literal k	11 00xxkkkk kkkk	
RETFIE -	Retornar de interrupción	00 00000000 1001	
RETLW k	Retornar y cargar a w con k	11 01xxkkkk kkkk	
RETURN -	Retornar de subrutina	00 00000000 1000	
SLEEP -	Ir al modo de bajo consumo	00 00000110 0011	TO,PD
SUBLW k	Restarle k a w	11 110xkkkk kkkk	C,DC,Z
XORLW k	Or exclusiva entre k y w	11 1010kkkk kkkk	Z

1.8 TECNICAS BASICAS DE PROGRAMACION DE MICROCONTROLADORES PICs.

La programación del PIC 16C84 se efectúa a través de 35 instrucciones, lenguaje ensamblador cruzado y utilizando el software de programación PARALLAX PIC16CXX ASEMBLER V 1.4.

Para llegar al programa final se hicieron varios subprogramas, luego estos se enlazaron hasta llegar al objetivo final. Antes de analizar estos subprogramas y el programa final se analizara varias técnicas y subprogramas utilizadas en la programación de los PICs.

1. Configuración de los pínes del puerto A y B como entradas o salidas.

   Ejemplo: configurar el puerto A y B de la siguiente forma

Puerto A= 5 pines

RA4	RA3	RA2	RA1	RA0
O	I	O	O	O

Puerto B= 8 pines

RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
I	I	I	I	O	O	O	O

Donde:

I= entrada

O= salida

Un 1 (uno) lógico determina las entradas y el 0 (cero) lógico las salidas.

Para el puerto A el valor será 01000 que corresponde a 8h en hexadecimal y el puerto B 11110000 que corresponde a F0h en hexadecimal.

La configuración se hace de la siguiente forma:

```
Movlw 0h      ;  cargar a W con 08h
Tris portA    ;   para configurar así el puerto A

Movlw 0Fh    ;   cargar a W con 0F0h
Tris portB    ;   para configurar así el puerto B
```

Observe que todas las operaciones se realizan a través del registro de trabajo W.

2. Configuración de registros.

   Ejemplo: configurar el registro OPTION con el valor 60h

```
Movlw 060h      ; se carga a W con 60h
Movwf option    ; se carga el registro option con W
```

De esta manera se configura el registro option así:

|

RBPU	INTEDG	RTS	RTE	PSA	PS2	PS1	PS0
0	1	1	0	0	0	0	0

Esto habilita las pull-up internas, selecciona el flanco de subida de la interrupción INT, selecciona el pin RA4/TOCK como fuente de la señal del TMRO, asigna el preescalador como contador y asigna el preescalador del TMRO con una relación de 1:2 (es decir; por cada 2 pulsos en el pín RA4, el registro RTCC se incrementa en 1.

3. cargar un registro de propósito general con W.

   Ejemplo: cargar el registro RTCC a W

```
Movf RTCC, W  ; cargar el registro W con RTCC
```

4. preguntar por el valor de un registro.

Ejemplo: el registro digi0 de propósito general se quiere mostrar si es menor de 10, si es 10 sédebe resetear.

```
Movf digi0, W; mueve el registro digi0 a W
Xorlw 0Ah          ; pregunta si el registro es igual a 10
Btfss status,2
Goto displays      ; si no muestrelo
  Movlw 0              ; si es igual limpie el registro digi0
  Movwf digi0
```

En la rutina display se muestra el digi0, obsérvese que la pregunta se realiza con la operación lógica Xor, veamos que pasa con digi0=10

```
Digi0   0Ah=   1010
Xor con 0Ah=   1010
Resultado   =   0000
```

El resultado de la operación lógica es 0 (cero).

Luego se prueba el bit 2 del registro status (indicador de ceros) con la orden Btfss la cual hace saltar una línea en el programa cuando el bit a probar es uno.

5. Enmascaramiento. El enmascaramiento se utiliza cuando queremos ocultar algunos bits que hacen Partes de un registro que no son importantes para la rutina.
   Ejemplo: enmascarar los bits 3 y 4 del puerto A, es decir los bits RA3 y RA4.

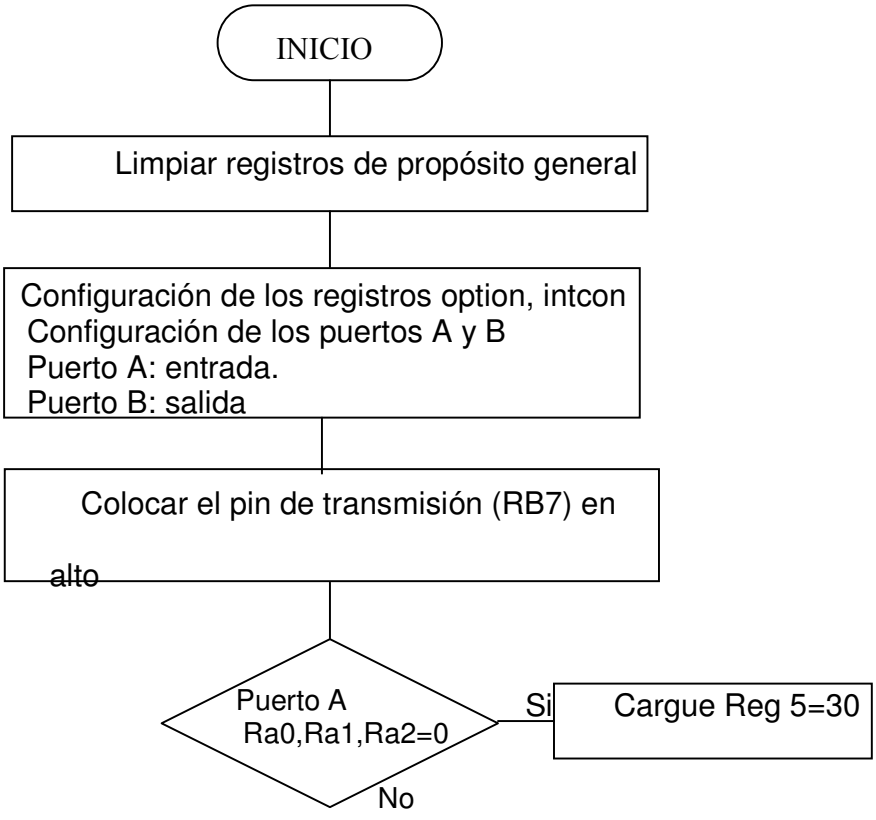
	RA4	RA3	RA2	RA1	RA0
Haciendo AND con 7	0	0	1	1	1
Quedaría	0	0	RA2	RA1	RA0

Las instrucciones quedarían así:

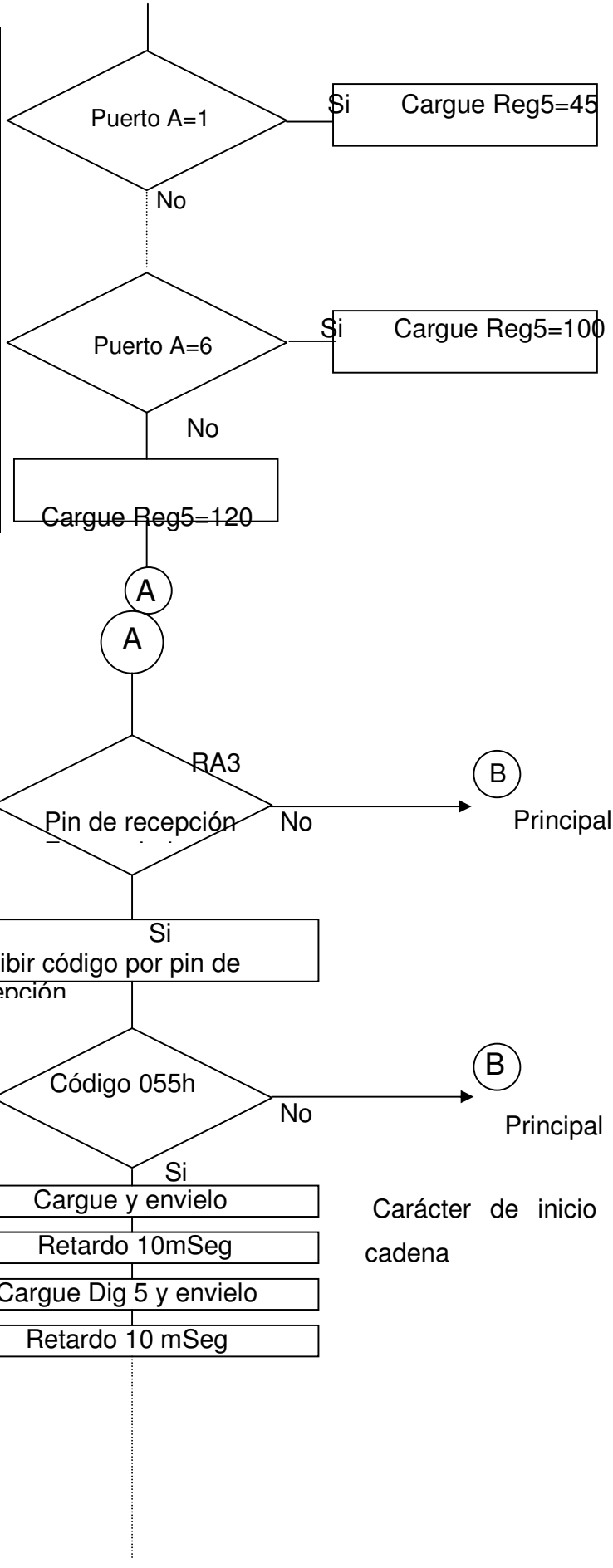
```
Movf portA , W      ; cargar W con portA
Andlw  7            ; hacer AND con 7 y W, el resultado queda cargado
                      en W
```

1.8.1 Programa principal lector.ASM

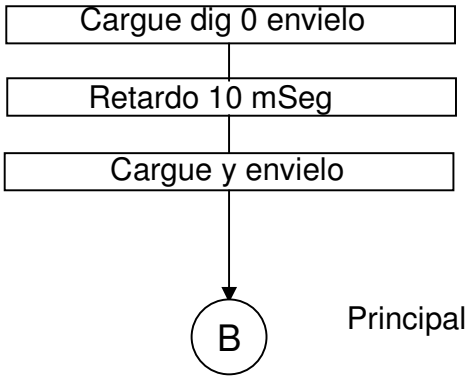
El diagrama de flujo del programa en el microcontrolador es el siguiente:

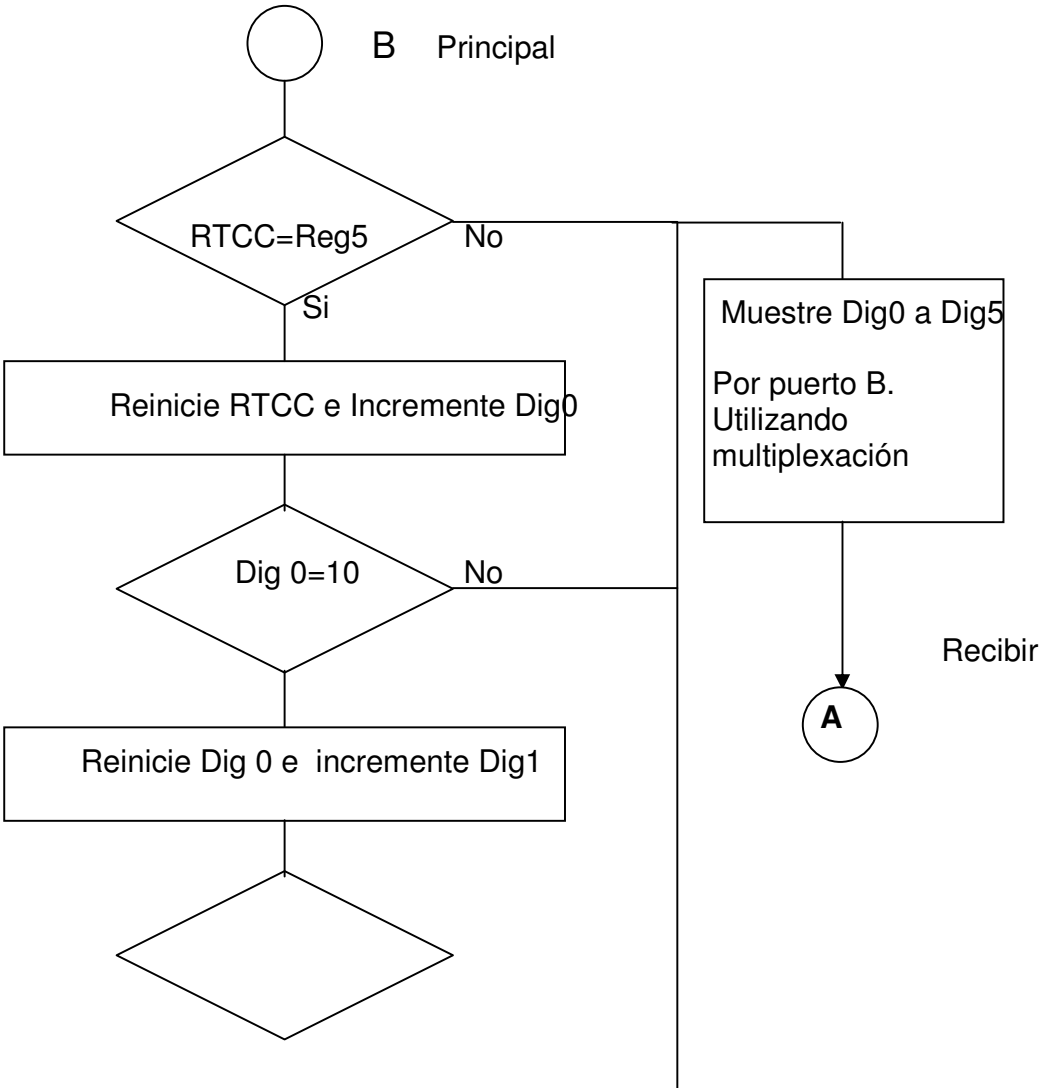


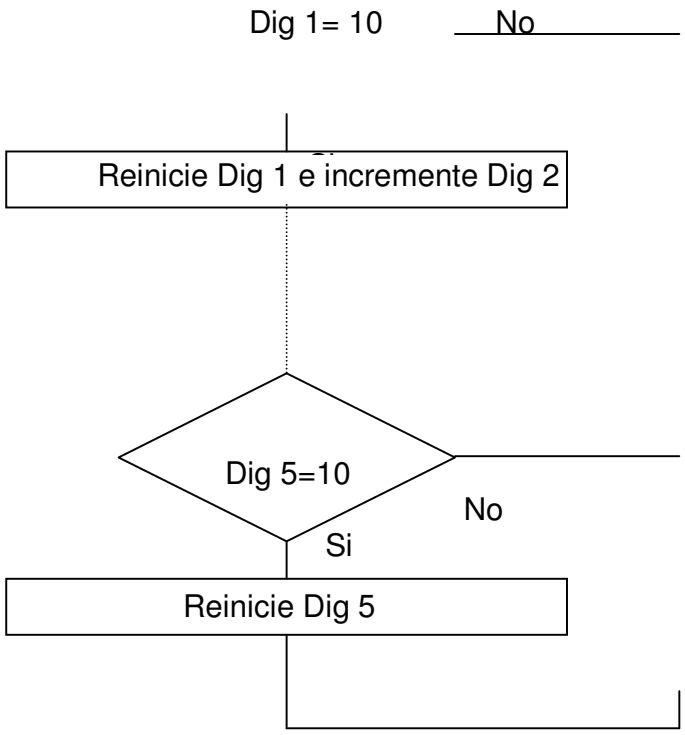
Puerto A	Reg 5	Kd
0	30	300
1	45	450
2	50	500
3	60	600
4	70	700
5	75	750
6	100	1000
7	120	1200



Carácter de inicio de cadena


Carácter de final de
cadena
Cargue dig 0 envielo
Retardo 10 mSeg
Cargue y envielo
B
Principal


B
Principal
RTCC=Reg5
No
Si
Muestre Dig0 a Dig5
Por puerto B.
Utilizando
multiplexación
Reinicie RTCC e Incremente Dig0
Dig 0=10
No
Recibir
A
Reinicie Dig 0 e incremente Dig1



A continuación se presenta el programa del microcontroladr con los respectivos comentarios.

```
===== Parallax PIC16Cxx Assembler v1.4 =====
digi0 equ 0ch  ;definición de la posición de memoria de los registros y bits
                utilizados en el programa
digi1 equu 0dh
digi2 equ 0eh
digi3 equ 0fh
digi4 equ 010h
digi5 equ 011h
trans equ 012h
tempo equ 013h
```

```
reg1 equ 014h
reg2 equ 015h
reg3 equ 016h
reg4 equ 02fh
reg5 equ 018h
status equ 03h
porta equ 05h
trisa equ 85h
portb equ 06h
trisb equ 86h
rtcc equ 01h
intcon equ 0bh
rp0 equ 5
option equ 01h
c equ 0
tx equ 7
rd equ 3
recepc equ 019h
transm equ 01ah
conta equ 01bh
retard equ 01ch

inicio  clrf digi0      ;limpia todos los registros de los
        clrf digi1      ;dígitos y trans del multipexaje
```

```
        clrf digi2
        clrf digi3
        clrf digi4
        clrf digi5
        clrf trans
        bsf status,rp0  ;activa la pag 1 de la memoria de programa
        movlw 060h      ;para configurar el registro option
        movwf option
        bcf status,rp0  ;regresa a la pag 0
        movlw 01fh      ;configura todo el puerto A como pines
        tris porta      ;de entrada
        movlw 0         ;se deshabilitan todas las interrupciones
        movwf intcon
        movlw 0         ;configura todo el puerto B como pines
        tris portb      ;de salida
        bsf portb,tx    ;coloca el pin de transmisión en alto

cero    movf porta,w    ;carga w con el valor del puerto A
        andlw 07h       ;enmascara RA3 y RA4 del puerto A
        xorlw 0h        ;pregunta sí el valor del puerto A
        btfss status,2  ;es cero
        goto uno        ;si no es cero ir a uno
        movlw 30        ;si es cero cargue reg5 con 30
        movwf reg5      ;Kd=300
```

```
        goto recibir        ;ir a la rutina recibir
uno     movf porta,w
        andlw 07h
        xorlw 01h
        btfss status,2
        goto dos
        movlw 45        ;si es uno Kd=450
        movwf reg5
        goto recibir
dos     movf porta,w
        andlw 07h
        xorlw 02h
        btfss status,2
        goto tres
        movlw 50        ;si es dos kd=500
        movwf reg5
        goto recibir
tres    movf porta,w
        andlw 07h
        xorlw 03h
        btfss status,2
        goto cuatro
        movlw 60        ;si es tres kd=600
        movwf reg5
```

```
        goto recibir
cuatro  movf porta,w
        andlw 07h
        xorlw 04h
        btfss status,2
        goto cinco
        movlw 70        ;si es cuatro Kd=700
        movwf reg5
        goto recibir
cinco   movf porta,w
        andlw 07h
        xorlw 05h
        btfss status,2
        goto seis
        movlw 75        ;si es cinco Kd=750
        movwf reg5
        goto recibir
seis    movf porta,w
        andlw 07h
        xorlw 06h
        btfss status,2
        goto siete
        movlw 100       ;si es seis Kd=1000
        movwf reg5
```

```
          goto recibir
siete     movlw 120        ;si es siete Kd=1200
          movwf reg5

recibir   clrf recepc      ;limpia el registro recepción
          btfsc porta,rd   ;¿ línea de recepción esta en bajo?
          goto principal   ;no, entonces vaya a principal
          call unoymedio   ;si, llamar rutina unoymedio bits
rcvr      movlw 8
          movwf conta      ;carga él numero de bits a transmitir
rnext     bcf status,c     ;limpia el bit de acarreo
          btfsc porta,rd   ;preguntar por el estado de la línea
          bsf status,c     ;activar carry sí esta en alto
          rrf recepc       ;rotar registro de recepción
          call unbit       ;llamar rutina de un bit
          decfsz conta     ;decrementar contador y saltar si es cero
          goto rnext       ;repita hasta completar el dato
código    movf recepc,w    ;cargar w con el dato recibido
          xorlw 055h
          btfss status,2   ;¿el dato recibido es 55h?
          goto principal   ;no, entonces vaya a principal
          movlw 02fh       ;cargar carácter de inicio /
          call enviar1     ;llamar rutina de envío
          call retardo     ;llamar rutina de retardo
```

```
            movf digi5,w        ;cargar w con el dígito 5
            call enviar         ;llamar rutina de envío
            call retardo        ;llamar rutina de retardo
            movf digi4,w
            call enviar
            call retardo
            movf digi3,w
            call enviar
            call retardo
            movf digi2,w
            call enviar
            call retardo
            movf digi1,w
            call enviar
            call retardo
            movf digi0,w
            call enviar
            call retardo
            movlw 02fh          ;cargar carácter final /
            call enviar1        ;llamar rutina enviar

principal   movf rtcc,w     ;mueva rtcc a w
              xorwf reg5,0
              btfss status,2   ;rtcc es igual a una décima de kw/h
```

```
goto display     ;no, ir a la rutina display
movlw 0
movwf rtcc       ;si, reinicie rtcc
incf digi0,1     ;incremente el dígito 0 en uno
movf digi0,0
xorlw 0ah
btfss status,2   ;¿dígito 0 es igual a 10?
goto display     ;no, ir a la rutina display
movlw 0
movwf digi0      ;si, reinicie dígito 0
incf digi1,1
movf digi1,0
xorlw 0ah
btfss status,2
goto display
movlw 0
movwf digi1
incf digi2,1
movf digi2,0
xorlw 0ah
btfss status,2
goto display
movlw 0
movwf digi2
```

```
                incf digi3,1


                movf digi3,0
                xorlw 0ah
                btfss status,2
               goto display
               movlw 0
               movwf digi3
               incf digi4,1
               movf digi4,0
               xorlw 0ah
               btfss status,2
               goto display
               movlw 0
               movwf digi4
               incf digi5,1
               movf digi5,0
               xorlw 0ah
               btfss status,2
               goto display
               movlw 0
               movwf digi5
display    movlw 8fh       ;carga dirección del display cero y
               movwf trans     ;mantiene el pin de transmisión en alto
```

```
            movf digi0,w
            call muestra      ;llamar rutina para mostrar el dígito 0
            movlw 9fh
            movwf trans
            movf digi1,w
            call muestra      ;llamar rutina para mostrar el dígito 1
            movlw 0afh
            movwf trans
            movf digi2,w
            call muestra
            movlw 0bfh
            movwf trans
            movf digi3,w
            call muestra
            movlw 0cfh
            movwf trans
            movf digi4,w
            call muestra
            movlw 0dfh
            movwf trans
            movf digi5,w
            call muestra
            goto recibir
muestra     addlw 240
```

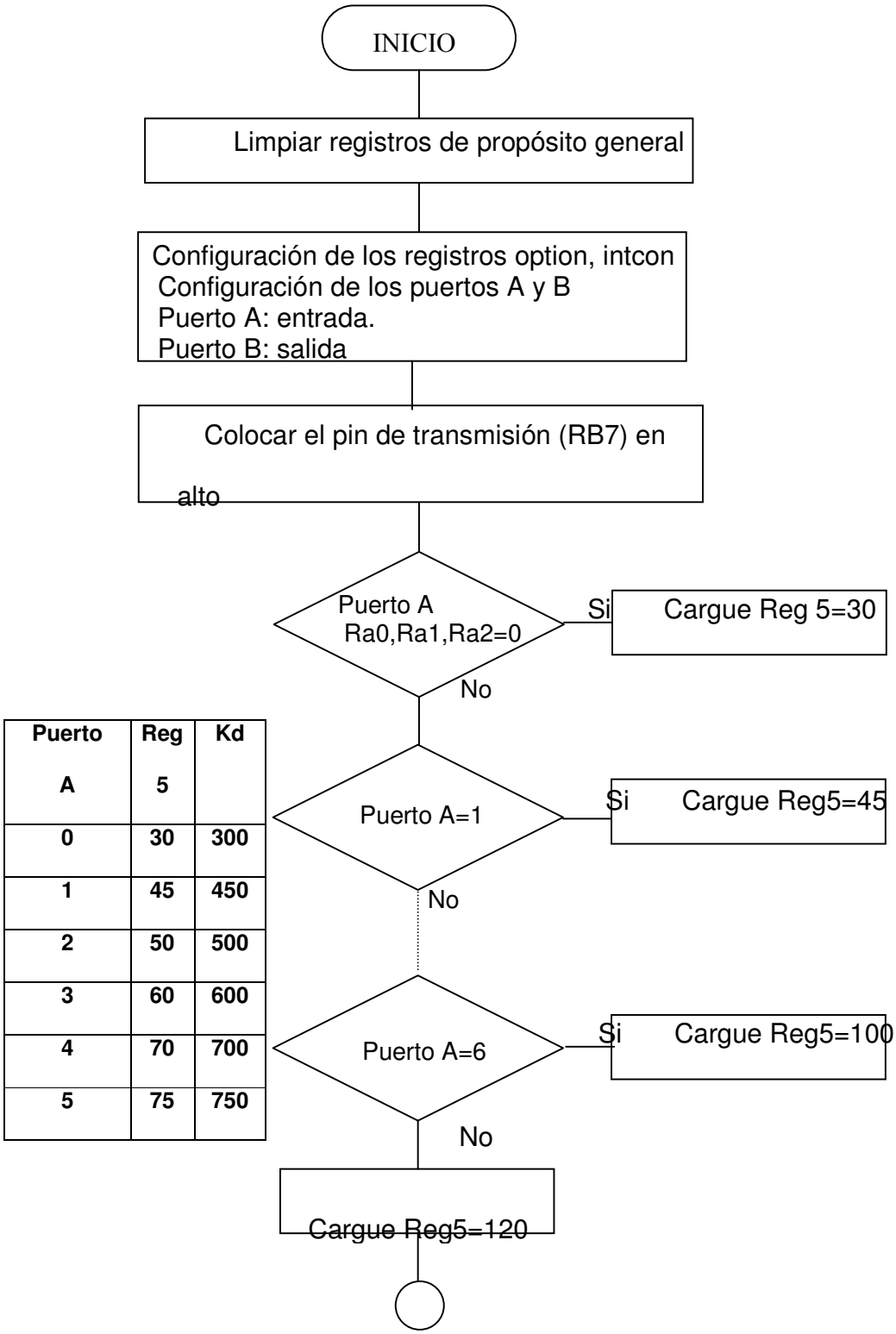
```
                    andwf trans,0    ;envía al puerto b el dígito(parte baja) y el
                                     valor del registro trans del display correspondiente
                    movwf portb
                    retlw 0          ;retornar

enviar    addlw 030h     ;adicionar 30hpara obtener dígitos como carácter
enviar1   movwf transm   ;lleva el contenido de w a transmisión
xmrt      movlw 8        ;cargar el contador con él numero de bits
          movwf conta    ;a transmitir
          bcf portb,tx   ;colocar línea de transmisión en bajo
          call unbit     ;para generar bit de arranque
xnext     bcf portb,tx   ;colocar línea de transmisión en bajo
          bcf status,c   ;limpiar carry
          rrf transm     ;rotar registro de transmisión
          btfsc status,c ;preguntar por el carry
          bsf portb,tx   ;si es cero, colocar línea en alto
          call unbit     ;llamar retardo de unbit
          decfsz conta   ;decrementar contador, saltar si es cero
          goto xnext     ;repetir hasta terminar
          bsf portb,tx   ;colocar línea de transmisión en alto
          call unbit     ;llamar retardo de un bit(bit de parada)
          retlw 0        ;retornar

unoymedio  movlw 222     ;cargar 1250 µs aproximadamente
```

```
                goto startup    ;ir a ejecutar tiempo
 unbit          movlw 148       ;cargar 833 µs aproximadamente
startup         movwf retard ;llevar valor de carga al retardo
redo            nop
                nop
                decfsz retard ;decrementar retardo, saltar si es cero
                goto redo       ;repetir hasta terminar
                retlw 0         ;retornar

retardo         movlw 5         ;rutina de retardo de 10 µs aprox.
                movwf reg1
                movlw 250
deca            movwf reg2
dec             nop
                decfsz reg2
                goto dec

                decfsz reg1
                goto deca
                retlw 0
                end             ;fin del ensamblador
```

===== Errors: 0 =====

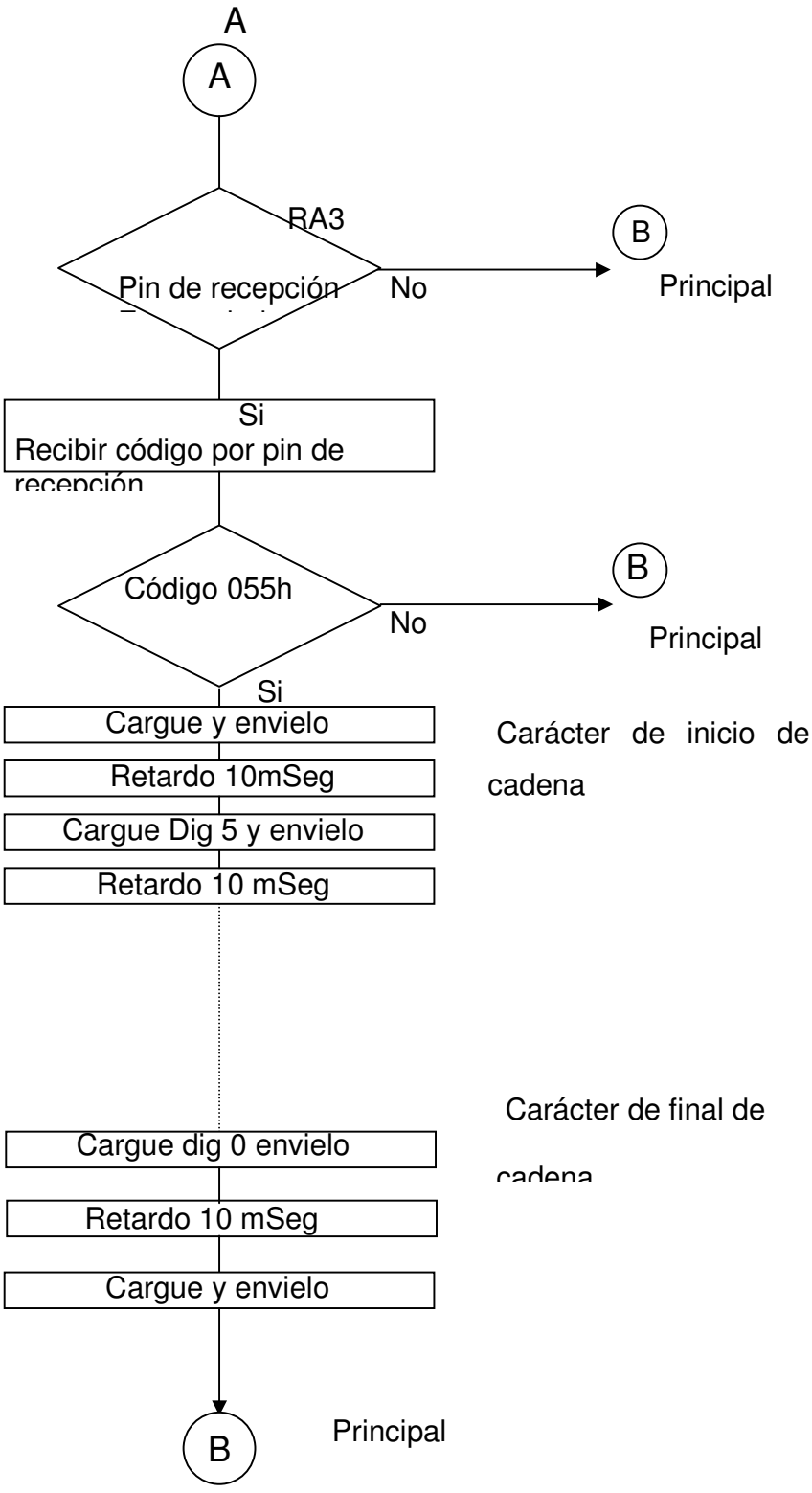
1.8.1 Programa principal lector.ASM

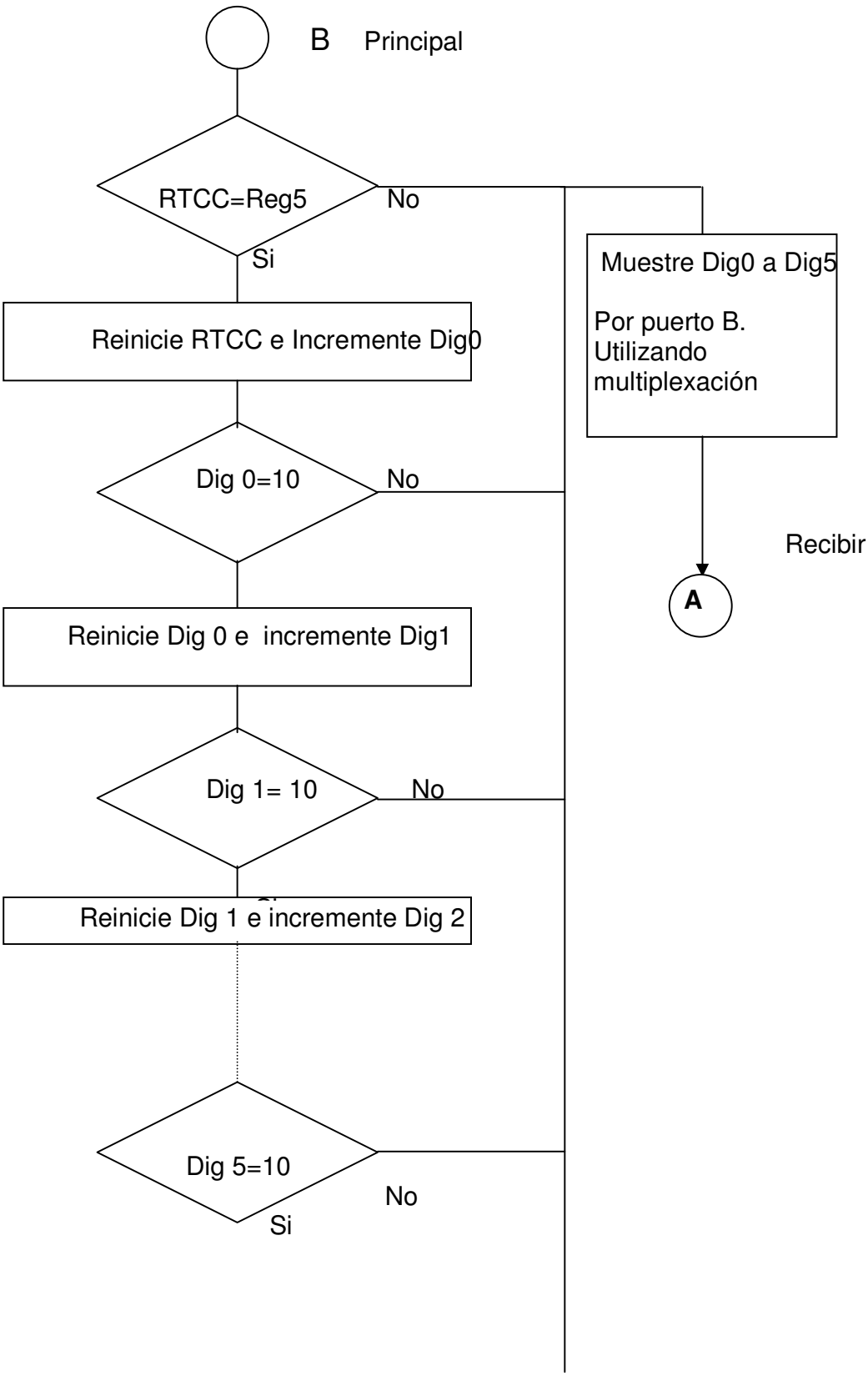
El diagrama de flujo del programa en el microcontrolador es el siguiente:

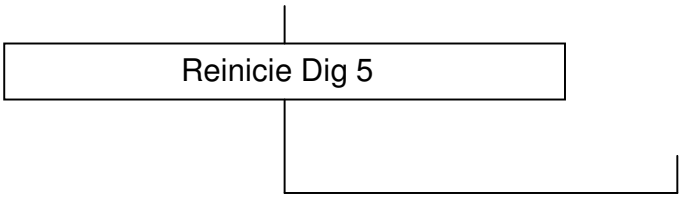


Puerto A	Reg 5	Kd
0	30	300
1	45	450
2	50	500
3	60	600
4	70	700
5	75	750

6	**100**	**1000**
7	**120**	**1200**




B
Principal
RTCC=Reg5
No
Si
Muestre Dig0 a Dig5
Por puerto B.
Utilizando
multiplexación
Reinicie RTCC e Incremente Dig0
Dig 0=10
No
Recibir
A
Reinicie Dig 0 e incremente Dig1
Dig 1= 10
No
Reinicie Dig 1 e incremente Dig 2
Dig 5=10
No
Si



A continuación se presenta el programa del microcontrolador con los respectivos comentarios.

```
===== Parallax PIC16Cxx Assembler v1.4 =====

digi0 equ 0ch  ;definición de la posición de memoria de los registros y bits
               utilizados en el programa

digi1 equ 0dh

digi2 equ 0eh

digi3 equ 0fh

digi4 equ 010h

digi5 equ 011h

trans equ 012h

tempo equ 013h

reg1 equ 014h

reg2 equ 015h

reg3 equ 016h

reg4 equ 02fh

reg5 equ 018h

status equ 03h

porta equ 05h

trisa equ 85h
```

```
portb equ 06h
trisb equ 86h
rtcc equ 01h
intcon equ 0bh
rp0 equ 5
option equ 01h
c equ 0
tx equ 7
rd equ 3
recepc equ 019h
transm equ 01ah
conta equ 01bh
retard equ 01ch

inicio  clrf digi0      ;limpia todos los registros de los
        clrf digi1      ;dígitos y trans del multipexaje
        clrf digi2
        clrf digi3
        clrf digi4
        clrf digi5
        clrf trans
        bsf status,rp0  ;activa la pag 1 de la memoria de programa
        movlw 060h      ;para configurar el registro option
        movwf option
```

```
        bcf status,rp0  ;regresa a la pag 0
        movlw 01fh      ;configura todo el puerto A como pines
        tris porta      ;de entrada
        movlw 0         ;se deshabilitan todas las interrupciones
        movwf intcon
        movlw 0         ;configura todo el puerto B como pines
        tris portb      ;de salida
        bsf portb,tx    ;coloca el pin de transmisión en alto

cero    movf porta,w    ;carga w con el valor del puerto A
        andlw 07h       ;enmascara RA3 y RA4 del puerto A
        xorlw 0h        ;pregunta sí el valor del puerto A
        btfss status,2  ;es cero
        goto uno        ;si no es cero ir a uno
        movlw 30        ;si es cero cargue reg5 con 30
        movwf reg5      ;Kd=300
        goto recibir    ;ir a la rutina recibir
uno     movf porta,w
        andlw 07h
        xorlw 01h
        btfss status,2
        goto dos
        movlw 45        ;si es uno Kd=450
        movwf reg5
```

```
        goto recibir
dos     movf porta,w
        andlw 07h
        xorlw 02h
        btfss status,2
        goto tres
        movlw 50        ;si es dos kd=500
        movwf reg5
        goto recibir
tres    movf porta,w
        andlw 07h
        xorlw 03h
        btfss status,2
        goto cuatro
        movlw 60        ;si es tres kd=600
        movwf reg5
        goto recibir
cuatro  movf porta,w
         andlw 07h
         xorlw 04h
         btfss status,2
         goto cinco
         movlw 70        ;si es cuatro Kd=700
         movwf reg5
```

```
            goto recibir
cinco     movf porta,w
            andlw 07h
            xorlw 05h
            btfss status,2
            goto seis
            movlw 75          ;si es cinco Kd=750
            movwf reg5
            goto recibir
seis       movf porta,w
            andlw 07h
            xorlw 06h
            btfss status,2
            goto siete
            movlw 100         ;si es seis Kd=1000
            movwf reg5
            goto recibir
siete     movlw 120         ;si es siete Kd=1200
            movwf reg5

recibir    clrf recepc            ;limpia el registro recepción
            btfsc porta,rd         ;¿ línea de recepción esta en bajo?
            goto principal         ;no, entonces vaya a principal
            call unoymedio      ;si, llamar rutina unoymedio bits
```

```
rcvr    movlw 8
        movwf conta      ;carga él numero de bits a transmitir
rnext   bcf status,c     ;limpia el bit de acarreo
        btfsc porta,rd   ;preguntar por el estado de la línea
        bsf status,c     ;activar carry sí esta en alto
        rrf recepc       ;rotar registro de recepción
        call unbit       ;llamar rutina de un bit
        decfsz conta     ;decrementar contador y saltar si es cero
        goto rnext       ;repita hasta completar el dato
código  movf recepc,w    ;cargar w con el dato recibido
        xorlw 055h
        btfss status,2   ;¿el dato recibido es 55h?
        goto principal   ;no, entonces vaya a principal
        movlw 02fh       ;cargar carácter de inicio /
        call enviar1     ;llamar rutina de envío
        call retardo     ;llamar rutina de retardo
        movf digi5,w     ;cargar w con el dígito 5
        call enviar      ;llamar rutina de envío
        call retardo     ;llamar rutina de retardo
        movf digi4,w
        call enviar
        call retardo
        movf digi3,w
        call enviar
```

```
            call retardo
            movf digi2,w
            call enviar
            call retardo
            movf digi1,w
            call enviar
            call retardo
            movf digi0,w
            call enviar
            call retardo
            movlw 02fh          ;cargar carácter final /
            call enviar1        ;llamar rutina enviar

principal   movf rtcc,w      ;mueva rtcc a w
              xorwf reg5,0
              btfss status,2   ;rtcc es igual a una décima de kw/h
              goto display     ;no, ir a la rutina display
              movlw 0
              movwf rtcc       ;si, reinicie rtcc
              incf digi0,1     ;incremente el dígito 0 en uno
              movf digi0,0
              xorlw 0ah
              btfss status,2   ;¿dígito 0 es igual a 10?
              goto display     ;no, ir a la rutina display
```

```
movlw 0
movwf digi0      ;si, reinicie dígito 0
incf digi1,1
movf digi1,0
xorlw 0ah
btfss status,2
goto display
movlw 0
movwf digi1
incf digi2,1
movf digi2,0
xorlw 0ah
btfss status,2
goto display
movlw 0
movwf digi2
incf digi3,1

movf digi3,0
xorlw 0ah
btfss status,2
goto display
movlw 0
movwf digi3
```

```
            incf digi4,1
            movf digi4,0
            xorlw 0ah
            btfss status,2
            goto display
            movlw 0
            movwf digi4
            incf digi5,1
            movf digi5,0
            xorlw 0ah
            btfss status,2
            goto display
            movlw 0
            movwf digi5
display     movlw 8fh       ;carga dirección del display cero y
            movwf trans     ;mantiene el pin de transmisión en alto
            movf digi0,w
            call muestra     ;llamar rutina para mostrar el dígito 0
            movlw 9fh
            movwf trans
            movf digi1,w
            call muestra     ;llamar rutina para mostrar el dígito 1
            movlw 0afh
            movwf trans
```

```
            movf digi2,w
            call muestra
            movlw 0bfh
            movwf trans
            movf digi3,w
            call muestra
            movlw 0cfh
            movwf trans
            movf digi4,w
            call muestra
            movlw 0dfh
            movwf trans
            movf digi5,w
            call muestra
            goto recibir
muestra  addlw 240
                  andwf trans,0    ;envía al puerto b el dígito(parte baja) y el
                        valor del registro trans del display correspondiente
             movwf portb
             retlw 0              ;retornar

enviar    addlw 030h      ;adicionar 30hpara obtener dígitos como carácter
 enviar1  movwf transm   ;lleva el contenido de w a transmisión
 xmrt      movlw 8          ;cargar el contador con él numero de bits
```

```
                movwf conta      ;a transmitir
                bcf portb,tx     ;colocar línea de transmisión en bajo
                call unbit       ;para generar bit de arranque
        xnext   bcf portb,tx     ;colocar línea de transmisión en bajo
                bcf status,c     ;limpiar carry
                rrf transm       ;rotar registro de transmisión
                btfsc status,c   ;preguntar por el carry
                bsf portb,tx     ;si es cero, colocar línea en alto
                call unbit       ;llamar retardo de unbit
                decfsz conta     ;decrementar contador, saltar si es cero
                goto xnext       ;repetir hasta terminar
                bsf portb,tx     ;colocar línea de transmisión en alto
                call unbit       ;llamar retardo de un bit(bit de parada)
                retlw 0          ;retornar

unoymedio  movlw 222     ;cargar 1250 µs aproximadamente
           goto startup  ;ir a ejecutar tiempo
 unbit     movlw 148     ;cargar 833 µs aproximadamente
startup    movwf retard ;llevar valor de carga al retardo
redo       nop
           nop
           decfsz retard ;decrementar retardo, saltar si es cero
           goto redo     ;repetir hasta terminar
```

```
                retlw 0          ;retornar

retardo         movlw 5        ;rutina de retardo de 10 μs aprox.
                movwf reg1
                movlw 250
deca            movwf reg2
dec             nop
                decfsz reg2
                goto dec

                decfsz reg1
                goto deca
                retlw 0
                end            ;fin del ensamblador
```

===== Errors: 0 =====

1.9 FUENTE DE PODER

Para cubrir las fallas debidas a las variaciones en el voltaje de entrada, la fuente de poder aceptará un rango de variación comprendido entre 80 y 240 VAC. Para tal caso se utilizará un transformador con cuatro devanados en la parte de alta tensión (max 240 VAC) y dos devanados en la parte de baja tensión (10-

13VAC), uno de estos devanados (secundario) tendrá una capacidad de corriente de 1 Amperio para alimentar todos los circuitos de la unidad de lectura, el otro (terciario) tendrá una capacidad de 0.1 A para alimentar los circuitos de control de la fuente de poder.

Cada devanado de la parte de alta tensión manejará un rango de voltaje, los límites en cada rango deben tener la misma relación para que el voltaje en la salida del transformador se mantenga dentro de límites preestablecidos, así:

Primer rango: 80 - K80 Voltios

Segundo rango: K80 - k^2 (80) Voltios

Tercer rango: K^2(80) - K^3(80) Voltios

Cuarto rango: K^3(80) - K^4(80) Voltios = 240 Voltios

De esta manera: K^4 = 240/80=3

K=1.32

Quedando los rangos así:

Primer rango: 80 - 105 Voltios

Segundo rango: 105 - 138 Voltios

Tercer rango: 138 - 182 Voltios

Cuarto rango: 182 - 240 Voltios

Cuando el voltaje es ligeramente superior a 80 voltios (primer rango), el voltaje en el secundario debe ser de 10 voltios. La relación de transformación es A1=80/10=8, si el voltaje es ligeramente inferior a 105 voltios, el voltaje en el secundario será de Vs= Vp/8=105/8=13.125 voltios.

Cuando el voltaje es ligeramente superior a 105 voltios (segundo rango), el voltaje en el secundario debe ser de 10 voltios. La relación de transformación es A2= 105/10=10.5, si el voltaje es ligeramente inferior a 138 voltios, el voltaje en el secundario será de Vs= Vp/10.5= 138/10.5=13.14 voltios.

De igual forma en el tercer rango la relación de transformación será A3= 13.8 y el voltaje secundario en el límite superior de 13.18 voltios.

En el cuarto rango la relación de transformación A4= 18.2 y el voltaje secundario en el límite superior de 13.18 voltios.

Nótese que el voltaje secundario varía de 10 a 13.18 voltios en todos los rangos.

1.9.1 Calculo del transformador. La capacidad de corriente en secundario del transformador es de 1Amperio, por tanto la potencia del secundario del transformador es:

$$P2=V2*I2= (13.18V)*(1A)= 13.18\ VA$$

La capacidad de corriente en el terciario del transformador es de 0,1 Amperio, por lo tanto la potencia del terciario del transformador es:

$$P3=V3*I3=(13.18V)*(0.1\ A)= 1.31\ VA$$

La potencia de salida del transformador será:

$$PS=P2+P3= 13.18\ VA + 1.31VA = 14.49\ VA$$

La potencia en el primario es:

P1= PS/n = 14.49 VA/0.9=16.1 VA

Donde:

P1: potencia en el primario

P2: potencia en el secundario

PS: potencia de salida del transformador

n: eficiencia del transformador, la cual es del 90% para transformadores menores de 500VA.

Por tanto el área de la sección transversal será:

S= 1.5(P1)^1/2 = 1.5(15.5)^1/2= aprox. 6 cm^2

Las espiras por voltio:

E/V = 38/S= 38/S= 6,33 espiras /voltio

Esta relación la tendremos cuando nos situemos en el límite superior del voltaje de cada rango así:

En el primario

Primer rango: 105V*6,33E/V = 665 espiras.
Segundo rango: 138V*6,33E/V=874 espiras.

Tercer rango: 182V*6,33E/V=1153 espiras.

Cuarto rango: 240*6,33E/V=1520 espiras.

En el secundario

13,18V*6,33E/V=84 espiras.

Para establecer los calibres de cada conductor se calculan las corrientes en cada devanado así:

En el primario

Primer rango: I=P1/V= 15,5 VA/105V= 0,147 A, calibre 31 AWG.

Segundo rango: I=P1/V= 15,5 VA/138V=0,112 A, calibre 32 AWG.

Tercer rango: I=P1/V= 15,5 VA/182V=0,085 A, calibre 33 AWG.

Cuarto rango: I=P1/V= 15,5VA/240V=0,064 A, calibre 34 AWG

En el secundario

Tiene una capacidad de 1A , calibre 22 AWG.

La figura 17 muestra el transformador, indicando el número de espiras de cada devanado y sus respectivos calibres.

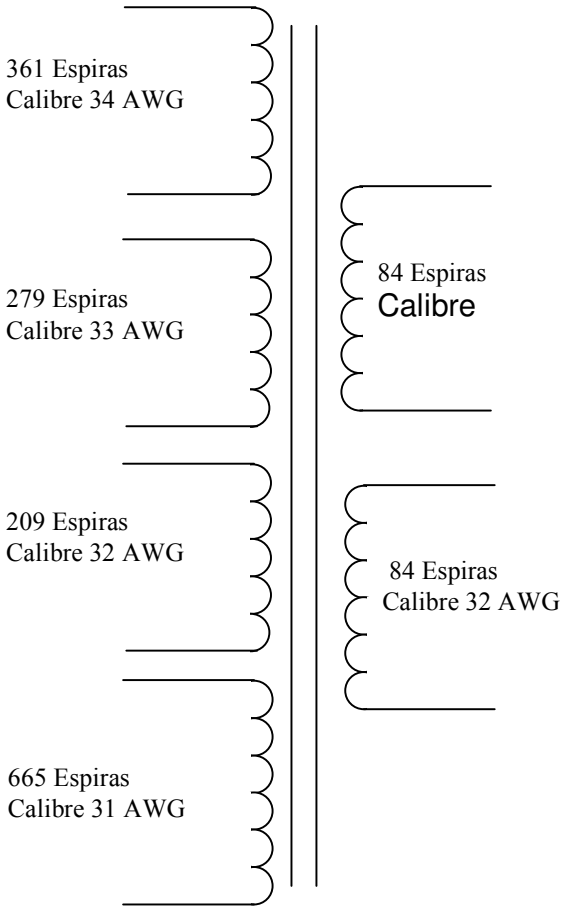


Figura 17. Numero de espiras del transformador.

1.9.2 principio de operación de la fuente de poder: La regulación de voltaje se realiza agregando espiras en el devanado primario a medida que el voltaje de alimentación se incrementa. Esta operación se realiza por medio de tres relés (RY1, RY2 y RY3). RY1 se acciona cuando el voltaje en la línea supera los 105 VAC, RY2 cuando el voltaje supera los 138 VAC y RY3 cuando el voltaje supera 182 VAC. Como se puede apreciar los relés se accionan en los límites superiores de los rangos de variación de voltaje.

Los relés RY1, RY2 y RY3 operan gracias a la acción de tres amplificadores operacionales de cuatro que posee el IC LM324. Estos trabajan como comparadores de voltaje, teniendo a través de sus terminales inversores un voltaje de 3,9V fijado por el diodo zener Z1. las entradas no inversoras van a tres puntos del divisor de voltaje de entrada formado R1, R2, R3 y R4 (Puntos 1, 2 y 3). De tal forma que el AO1(Amplificador operacional 1) cambie su salida de bajo a alto cuando en el punto 1 el voltaje esté unos milivoltios por encima de los 3,9V fijados por el zener en la entrada inversora, esto se da cuando el voltaje de entrada es mayor a 105VAC. De esta misma forma en el AO2 (Amplificador operacional 2) su salida pasa de bajo a alto cuando el voltaje de línea supera los 138 VAC y en el AO3 (Amplificador operacional 3) su salida pasa de bajo a alto cuando el voltaje de la línea supera los 182 VAC.

El voltaje de la línea es rectificado por medio del diodo D1 y filtrado por el condensador C1, obteniéndose un voltaje DC aproximadamente igual a 1,414 veces el voltaje de la línea, este voltaje luego es aplicado al divisor de tensión formado por R1, R2, R3 y R4. Este divisor de tensión es calculado para disipar menos de 500 mW cuando el voltaje de alimentación es el máximo (240 VAC) y una corriente de 0,45 mA cuando el voltaje de entrada es mínimo.

Las resistencias R5, R6 y R7 polarizan las bases de los transistores Q1, Q2 y Q3 que activan los relés RY1, RY2 y RY3 cuando el voltaje en las salidas de los amplificadores operacionales AO1, AO2 y AO3 pasa de bajo a alto. Los diodos D2, D3 y D4 protegen a los transistores Q1, Q2 y Q3 en el momento de la desconexión de los relés debido al carácter inductivo de sus bobinas.

El transformador posee un primario con cuatro devanados, un secundario y un devanado terciario. El devanado terciario proporciona la alimentación al circuito comparador a través del regulador 7812, el devanado secundario proporciona alimentación de 9 V a través del regulador 7809 que alimenta el módem.

Los 9V a la salida del regulador 7809 alimentan el regulador 7805 de 5V que alimenta la tarjeta principal y la tarjeta de visualización.

El microcontrolador es alimentado por el circuito de respaldo, el cual suministra de manera automática 3,5V al microcontrolador en caso de que falle la fuente de alimentación a fin de evitar la perdida de datos. En estado de espera las baterías reciben una carga de 20mA aproximadamente a través de R11 y Q4. Cuando falla la fuente de alimentación Q4 la aísla de la carga y Q5 conduce para hacer el cambio a la energía de la batería.

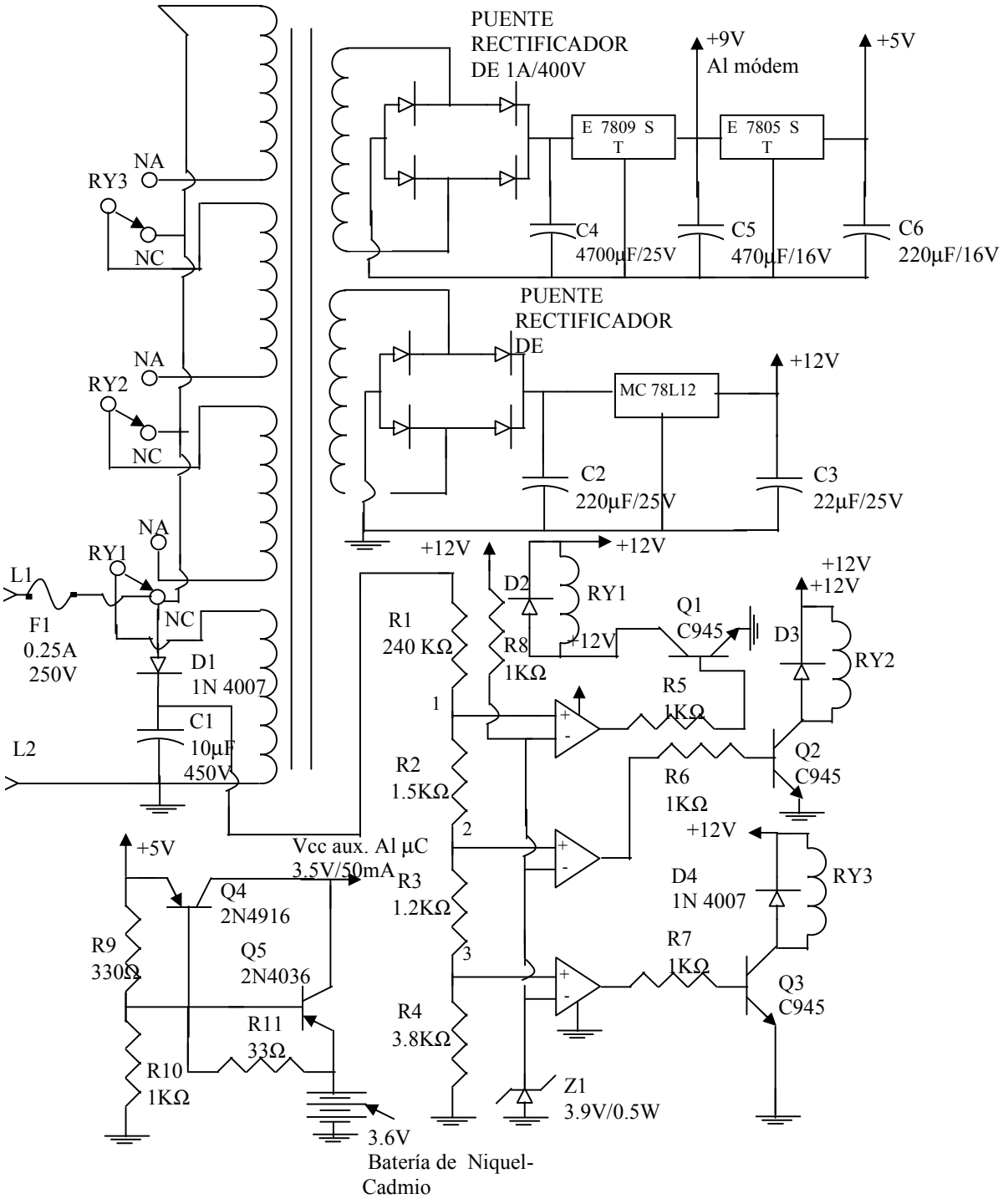


Figura 18. Fuente de poder.

Figura 3. Circuito selector de Kd.

Los valores de Kd se programan según la tabla 1. Estos valores de Kd que aparecen en la tabla fueron suministrados por el departamento de calibración de contadores de la ELECTRIFICADORA DE BOLIVAR, como los 8 valores de Kd mas utilizados en la ciudad de Cartagena.

Cualquier otro valor de Kd puede ser utilizado cambiando el valor a cargar en el registro 5 en el momento de la programación, teniendo en cuenta que este valor debe ser entero y su décima parte debe ser menor o igual a 255 (FF) que es el valor máximo con el que se puede cargar un registro en los PIC16CXX.

Cuadro 1. Programación del Kd(Rev/Kwh) en el microcontrolador

Ra2	Ra1	Ra0	Registro5	Kd(Rev/Kwh)
0	0	0	30	300
0	0	1	45	450
0	1	0	50	500
0	1	1	60	600
1	0	0	70	700
1	0	1	75	750
1	1	0	100	1000
1	1	1	120	1200

2.2 DISEÑO DE LA UNIDAD DE CONSULTA

2.2.1 Diseño de la base de datos.

Para el manejo de la información del proyecto LACEE en la unidad de consulta, se crea el siguiente archivo de base de datos: "LACEE.MDB".

Es un archivo de base de datos que permite conectividad con Microsoft ACCESS; creada con la versión 3.5 del motor jet de Microsoft (es un programa que proporciona los principales medios para construir la interfaz entre Visual Basic y las bases de datos que permiten almacenar en un solo archivo todas las tablas y demás información sobre la base de datos.

2.2.1.1 Descripción de tablas

A continuación se describen cada una de las tablas que hacen parte del archivo de base de datos "LACEE.MDB".

a. Tabla usuarios.

Descripción: es una tabla que contiene la información básica de cada usuario.
Procesos: creación. Inserción, modificación, eliminación y consulta de registros.

Indice: teléfono, es un campo único y es requerido.
Cuadro 7. Tabla usuarios

CAMPO	TIPO	TAMAÑO (BYTES)	DESCRIPCIÓN
NUMERO	LONG	4	Se auto incrementa con cada registro de la tabla
NOMBRE	TEXTO	40	Nombre y apellidos del usuario
DIRECCION	TEXTO	60	Dirección del usuario
TELEFONO	TEXTO	8	Número telefónico del usuario,

			donde estará conectada la unidad de lectura

b. Tabla consumo.

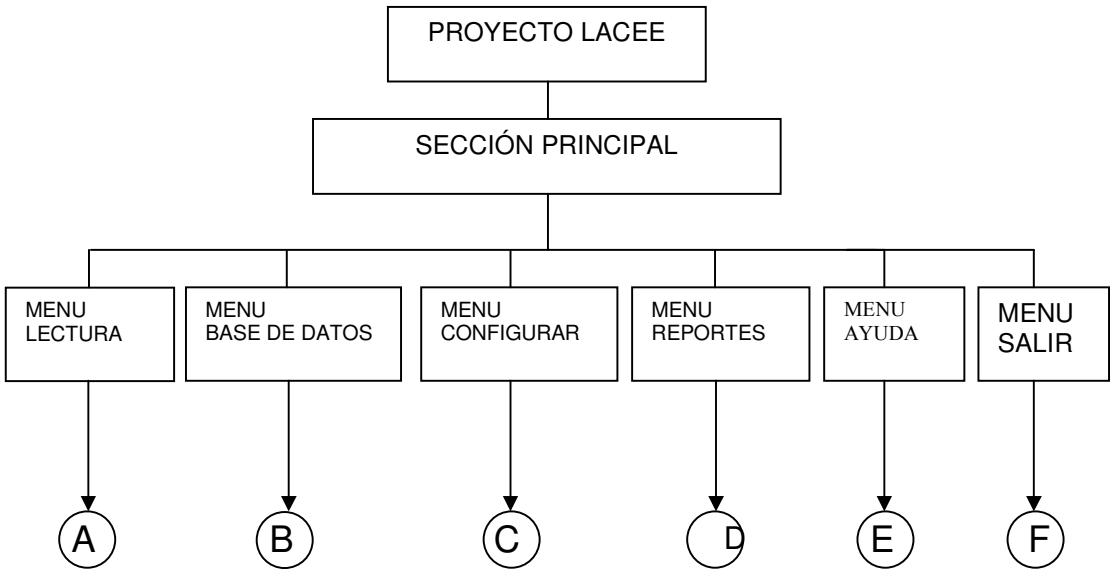
Descripción: en esta tabla se almacena toda la información de la lectura del consumo realizada a cada usuario.
Procesos: creación. Inserción y consulta de registros.
Indice: teléfono, es un campo único y requerido.

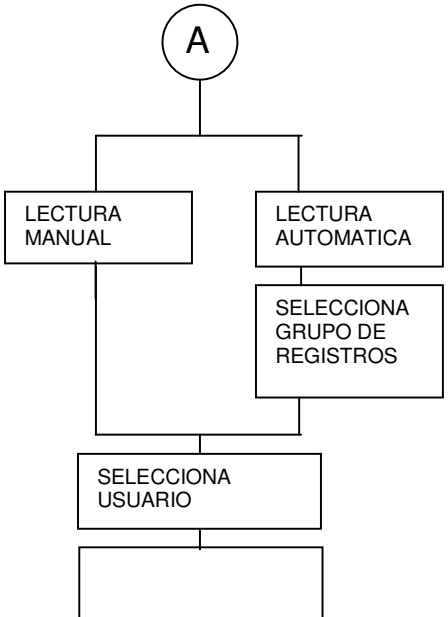
CAMPO	TIPO	TAMAÑO (BYTES)	DESCRIPCION
NUMERO	LONG	4	Se auto incrementa con cada registro de la tabla
TELEFONO	TEXTO	8	Número telefónico del usuario, al cual se le realiza la lectura del consumo
FECHA	TEXTO	10	Fecha en la cual se realiza la lectura del consumo
HORA	TEXTO	12	Hora en la cual se realiza la lectura de consumo
LECT-ACT	DOUBLE	8	Valor de la lectura actual que se obtiene durante el proceso de la comunicación
LECT-ANT	DOUBLE	8	Valor de la lectura anterior que se obtuvo en el proceso

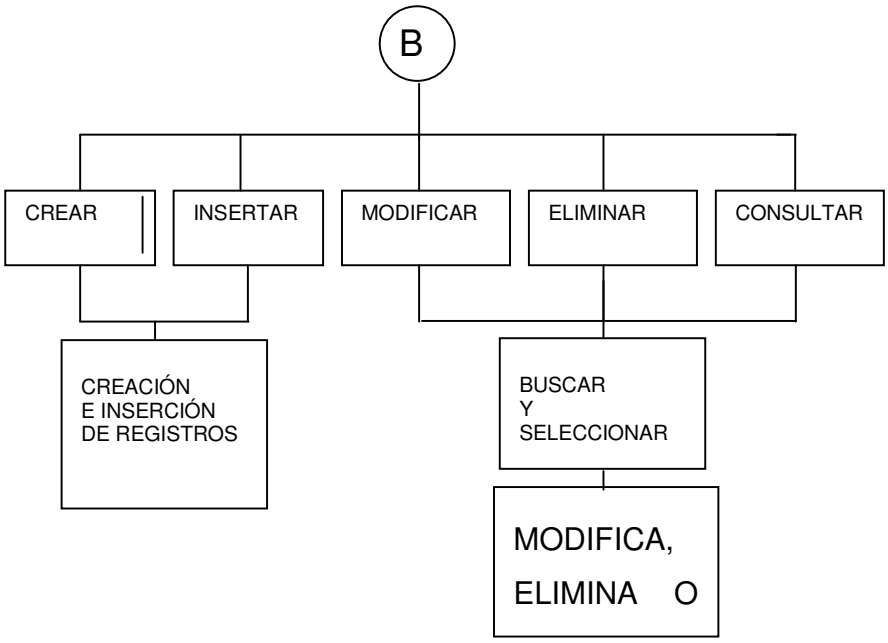
			de comunicación
CONSUMO	DOUBLE	8	Lect., actual-Lect. anterior

2.2.2 Diseño modular

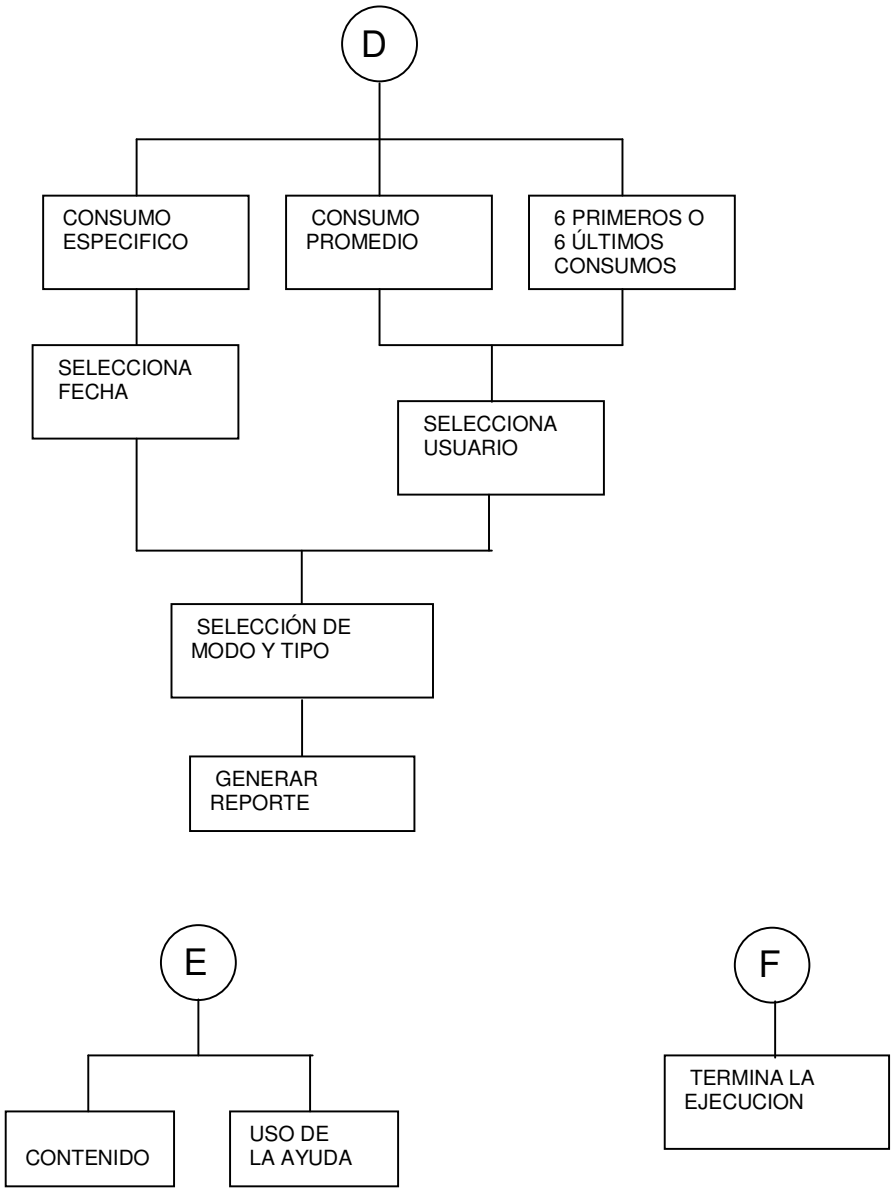
El diagrama visual del contenido es el siguiente:






B
CREAR
INSERTAR
MODIFICAR
ELIMINAR
CONSULTAR
CREACIÓN
E INSERCIÓN
DE REGISTROS
BUSCAR
Y
SELECCIONAR
MODIFICA,
ELIMINA O


C
CONFIGURA EL
PUERTO DE
COMUNICACIONES


D
CONSUMO ESPECIFICO
CONSUMO PROMEDIO
6 PRIMEROS O 6 ÚLTIMOS CONSUMOS
SELECCIONA FECHA
SELECCIONA USUARIO
SELECCIÓN DE MODO Y TIPO
GENERAR REPORTE
E
CONTENIDO
USO DE LA AYUDA
F
TERMINA LA EJECUCION

2.2.2.1 Descripción de los modulos

Sección principal: a partir de este modulo se ejecutan lo sub modulos que controlan el funcionamiento de la aplicación: menú lecturas, menú base de datos, menú configuración, menú reportes, menú ayuda, y salir de la aplicación. La ventana principal se utiliza como contenedor del resto de ventanas de la aplicación.

Menú lecturas: es el modulo a partir del cual se acceden a las funciones o sub modulos, lectura manual y lectura automática.

Lectura manual: a través de este modulo se accede a la tabla usuarios de la base de datos LACEE.MDB con el propósito de seleccionar a un usuario a el cual se le realiza la lectura del valor del consumo. Este modulo controla la ejecución del sub modulo proceso de comunicación y además controla la ejecución del sub modulo resultado y almacenamiento de la lectura.

Proceso de comunicación: en este modulo se realiza todo el proceso que controla la recepción y envío de datos a través del puerto serial. Una ves seleccionado el usuario de la base de datos, este proceso se encarga de enviar la señal al puerto de comunicaciones para realizar la llamada y se queda esperando la respuesta. Una ves recibido el mensaje de respuesta (valor de la lectura o algún error), se procede a mostrar el resultado de la lectura y se almacena si no ha ocurrido algún error.

Resultado y almacenamiento de la lectura: este modulo visualiza el valor de la lectura obtenida en el proceso de comunicación y de almacenar dicho valor en la tabla de consumo de base de datos LACEE.MDB. Si ocurre algún error en el proceso de comunicación el resultado de la lectura es cero y el valor no se almacena.

Lectura automática: a través de este modulo se accede a la tabla usuarios de la base de datos LACEE.MDB con el fín de realizar la selección de un grupo de usuarios a los cuales se le realizan la lectura del valor del consumo de manera automática. Este modulo controla la ejecución del sub modulo filtrar o seleccionar grupo de registros, el sub modulo proceso de comunicación y el sub modulo resultado y almacenamiento de la lectura.

Menú base de datos: es el modulo principal a partir del cual se maneja o manipula toda la información de la tabla usuarios de la base de datos LACEE.MDB. Controla la ejecución de los submodulos crear, insertar, modificar y eliminar.

Crear, insertar, modificar y eliminar: estos sub modulos están contenidos en una sola sección de código, la cual se ejecuta según la opción escogida en el menú base de datos. Al seleccionar la opción crear, si es primera ves que se realiza esta opción, se crean los archivos de base de datos LACEE.MDB; una ves creada la base de datos se podrán insertar registros de forma inmediata o escogiendo la opción insertar. Cuando se selecciona la opción modificar, la base de datos

permite la modificación de los registros. Con la opción eliminar, se puede eliminar de la base de datos el registro seleccionado.

Estos modulos controlan la ejecución del submodulo buscar, el cual permite seleccionar un registro de la base de datos LACEE.MDB para luego ejecutar a modificación o eliminación de dicho registro.

Menú configuración: este modulo controla la ejecución del sub modulo configurar puerto, en este sub modulo se establecen las propiedades del puerto de comunicaciones, con el fín de poder realizar las lecturas de consumo.

Menú reportes:a partir de este modulo se generan los reportes por impresora o pantalla. Estos reportes son creados a partir de consultas que se realizan a la tabla consumos de la base de datos LACEE.MDB. Controla la ejecución de los diferentes tipos de reportes como 6 primeros/últimos consumos, consumo promedio y consumo específico.

6 primeros/últimos consumos: este modulo es el encargado de realizar los reportes 6 primeros o 6 últimos consumos, se permite que el usuario del software seleccione el lugar de impresión del reporte.

Consumo promedio: este modulo se encarga de generar el reporte del consumo promedio correspondiente a el usuario seleccionado. Controla la presentación del reporte, modo gráfico o listado, en pantalla o impresora.

Consumo específico: por medio de este modulo se genera un listado con el valor del consumo, dada una fecha. Este listado se obtiene luego de hacer una consulta sobre la tabla consumos de la base de datos LACEE.MDB.

Menú ayuda: es el modulo principal que se encarga de presentar al usuario del software un archivo de ayuda sobre la aplicación y utilización del mismo. Controla la ejecución de los módulos contenido y uso de la ayuda.

Contenido: en este modulo se le presenta al usuario del software, el contenido general de la ayuda sobre la aplicación, de acuerdo con la escogencia, presenta los diferentes temas de ayuda que este abarca.

Uso de la ayuda: en este módulo se presenta al usuario del software, el contenido general del uso de windows.

Para el desarrollo del diseño modular y el diseño de la interfaz de entrada y salida se utilizó el lenguaje de programación Visual Basic versión 5.0, el cual permitió dicho desarrollo de una forma rápida, sencilla y practica.

El código del proyecto LACEE por modulos es el siguiente:

CODIGO DEL PROYECTO LACEE ESPECIFICADO POR MODULOS

Sección Principal

```
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'Este formulario corresponde a la ventana principal del proyecto.~
'Es un formulario MDI padre, el cual contiene el menú y la barra ~
'de herramientas.                                                 ~
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Private Sub MDIForm_Activate()
 Unload FormSplash
End Sub

'Esta subrutina se ejecuta cuando el formulario principal se carga
'en memoria.
Private Sub MDIForm_Load()
 On Error GoTo CapturaErr
  Load FormConfig
  App.HelpFile = App.Path + "\LACEE.HLP"
 Exit Sub
CapturaErr:
 Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
 Exit Sub
End Sub

'Esta subrutina se ejecuta cuando se selecciona la opción Lectura
'Automatica del Menú Lecturas
Private Sub Mnu1Auto_Click()
 'El siguiente código verifica si existe el archivo de base de datos
 Dim BD As Database
 On Error GoTo CapturaErr
  Set BD = OpenDatabase(App.Path + "\LACEE.MDB")
  FormLecAuto.Show 1
  BD.Close
 Exit Sub
CapturaErr:
 Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
 Exit Sub
End Sub

'Esta subrutina se ejecuta cuando se selecciona la opción Lectura Manual
'del Menú Lecturas
Private Sub Mnu1Manual_Click()
 'El siguiente código verifica si existe el archivo de base de datos
 Dim BD As Database
 On Error GoTo CapturaErr
  Set BD = OpenDatabase(App.Path + "\LACEE.MDB")
  FormLecMan.Show 1
  BD.Close
 Exit Sub
CapturaErr:
 Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
 Exit Sub
End Sub
'Esta subrutina se ejecuta cuando se selecciona la opción Consultar
'del Menú Base de Datos
Private Sub Mnu2Consul_Click()
 Dim BD As Database
 OpcMnu = 5
```

```
 Unload FormOpcMnuBD
 On Error GoTo CapturaErr
  Set BD = OpenDatabase(App.Path + "\LACEE.MDB")
  FormOpcMnuBD.Show
 Exit Sub
CapturaErr:
 Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
 Exit Sub
End Sub

'Esta subrutina se ejecuta cuando se selecciona la opción Crear
'del Menú Base de Datos
Private Sub Mnu2Crear_Click()
 Dim BD As Database, WS As Workspace
 Dim Tabla1, Tabla2 As TableDef
 Dim Cam1, Cam2, Cam3, Cam4, Campo2(6) As Field
 Dim Idx1, Idx2 As Index
 Dim Relac As Relation
 Dim NomBD As String, x As Integer

 On Error GoTo CapturaErr
  Set WS = DBEngine.Workspaces(0)
  NomBD = App.Path + "\LACEE.MDB"
  Set BD = WS.CreateDatabase(NomBD, dbLangGeneral)
  Set Tabla1 = BD.CreateTableDef("Usuarios")
  Set Tabla2 = BD.CreateTableDef("Consumos")
  Set Cam1 = Tabla1.CreateField("Numero", dbLong)
  Cam1.Attributes = dbAutoIncrField
  Set Cam2 = Tabla1.CreateField("Nombre", dbText, 30)
  Set Cam3 = Tabla1.CreateField("Direccion", dbText, 50)
  Set Cam4 = Tabla1.CreateField("Telefono", dbText, 7)
  Set Campo2(0) = Tabla2.CreateField("Numero", dbLong)
  Campo2(0).Attributes = dbAutoIncrField
  Set Campo2(1) = Tabla2.CreateField("Telefono", dbText, 7)
  Set Campo2(2) = Tabla2.CreateField("Fecha", dbText, 10)
  Set Campo2(3) = Tabla2.CreateField("Hora", dbText, 12)
  Set Campo2(4) = Tabla2.CreateField("Consumo", dbDouble)
  Set Campo2(5) = Tabla2.CreateField("LecAnte", dbDouble)
  Set Campo2(6) = Tabla2.CreateField("LecActu", dbDouble)
  BD.Close

  'creacion de indices
  Set BD = OpenDatabase(App.Path + "\LACEE.MDB")
  Set Tabla1 = BD!Usuarios
  Set Idx1 = Tabla1.CreateIndex("Telefono")
  Set Tabla2 = BD!Consumos
  Set Idx2 = Tabla2.CreateIndex("Telefono")
  With Idx1
   .Primary = True
   .Unique = True
   Set Cam3 = .CreateField("Telefono")
   Cam3.Name = "Telefono"
   Cam3.Type = dbText
   Cam3.Size = 7
   .Fields.Append Cam3
  End With
  With Idx2
```

```
    .Primary = True
    .Unique = False
    Set Cam2 = .CreateField("Telefono")
    Cam2.Name = "Telefono"
    Cam2.Type = dbText
    Cam2.Size = 7
    .Fields.Append Cam2
   End With
   Tabla1.Indexes.Append Idx1
   Tabla2.Indexes.Append Idx2

   'Creación de Relaciones
   Set Relac = BD.CreateRelation("Consu_Usuario", "Usuarios", "Consumos",_
       dbRelationUpdateCascade + dbRelationDeleteCascade)
   Set Cam1 = Relac.CreateField("Telefono", dbText, 7)
   Cam1.ForeignName = "Telefono"
   Relac.Fields.Append Cam1
   BD.Relations.Append Relac
   BD.Close
   OpcMnu = 1
   Unload FormOpcMnuBD
   FormOpcMnuBD.Show
Exit Sub
CapturaErr:
 Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
 Exit Sub
End Sub

'Esta subrutina se ejecuta cuando se selecciona la opción Eliminar
'del Menú Base de Datos
Private Sub Mnu2Eliminar_Click()
 Dim BD As Database
 OpcMnu = 4
 Unload FormOpcMnuBD
 On Error GoTo CapturaErr
  Set BD = OpenDatabase(App.Path + "\LACEE.MDB")
  FormOpcMnuBD.Show
 Exit Sub
CapturaErr:
 Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
 Exit Sub
End Sub

'Esta subrutina se ejecuta cuando se selecciona la opción Insertar
'del Menú Base de Datos
Private Sub Mnu2Insertar_Click()
 Dim BD As Database
 OpcMnu = 2
 Unload FormOpcMnuBD
 On Error GoTo CapturaErr
  Set BD = OpenDatabase(App.Path + "\LACEE.MDB")
  FormOpcMnuBD.Show
 Exit Sub
CapturaErr:
 Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
 Exit Sub
End Sub
```

```
'Esta subrutina se ejecuta cuando se selecciona la opción Modificar
'del Menú Base de Datos
Private Sub Mnu2Mod_Click()
 Dim BD As Database
 OpcMnu = 3
 Unload FormOpcMnuBD
 On Error GoTo CapturaErr
  Set BD = OpenDatabase(App.Path + "\LACEE.MDB")
  FormOpcMnuBD.Show
 Exit Sub
CapturaErr:
 Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
 Exit Sub
End Sub

'Esta subrutina se ejecuta cuando se selecciona el Menú configuración.
Private Sub MnuConfig_Click()
 FormConfig.Show 1
End Sub

'Esta subrutina se ejecuta cuando se selecciona la opción Consumo
'Específico del Menú Reportes
'del menú Reportes
Private Sub Mnu4Especifico_Click()
 FormConEsp.Show 1
End Sub

'Esta subrutina se ejecuta cuando se selecciona la opción 6 Primeros
'Consumos del Menú Reportes
Private Sub Mnu4Primeros_Click()
 OpcMnu = 1
 FormReportesPC.Show 1
End Sub

'Esta subrutina se ejecuta cuando se selecciona la opción Consumo
'Promedio del Menú Reportes
Private Sub Mnu4Promedio_Click()
 FormConProm.Show 1
End Sub

'Esta subrutina se ejecuta cuando se selecciona la opción 6 Ultimos
'Consumos del Menú Reportes
Private Sub Mnu4Ultimos_Click()
 OpcMnu = 2
 FormReportesPC.Show 1
End Sub

'Esta subrutina se ejecuta cuando se selecciona la opción Contenido del -
'Menú Ayuda
Private Sub Mnu6_Click()
 Dim vr As Long
 vr = WinHelp(hwnd, (App.HelpFile), HELP_INDEX, CLng(0))
End Sub

'Esta subrutina se ejecuta cuando se selecciona la opción Buscar del -
'Menú Auda
```

```
Private Sub Mnu6Bus_Click()
 Dim vr As Long
 vr = WinHelp(hwnd, (App.HelpFile), HELP_PARTIALKEY, CLng(0))
End Sub

'Esta subrutina se ejecuta cuando se selecciona la opción Uso de la Ayuda
'del Menú Ayuda
Private Sub Mnu6Uso_Click()
 Dim vr As Long
 vr = WinHelp(hwnd, (App.HelpFile), HELP_HELPONHELP, CLng(0))
End Sub

'Esta subrutina se ejecuta cuando se selecciona el Menú Salir.
Private Sub MnuSalir_Click()
 End
End Sub
```

Lectura Manual

```
Private Sub BtnAceptar_Click()
 ResultLect = 1
 MAC = True
 FormProgreso.Show 1
 FormResLec.Show 1
End Sub


Private Sub BtnCancelar_Click()
 Unload FormLecMan
End Sub

Private Sub Form_Load()
On Error GoTo CapturaError
' El siguiente código le asigna la Base de Datos al control Data1 de este
formulario
 Data1.DatabaseName = App.Path + "\LACEE.MDB"
 Data1.RecordSource = "Usuarios"
 Data1.RecordsetType = vbRSTypeDynaset
 Data1.Refresh
 Data1.Recordset.MoveFirst
 'Coloca en el combo la expresión 0 de la lista "Nombre"
 CmbCampo.Text = CmbCampo.List(0)
' Captura de errores
Salir:
  Exit Sub
CapturaError:
  MsgBox Err.Description & Chr$(13) & Err
  Resume Salir
End Sub

Private Sub TxtTele_Change()
'Busca en la base de datos la primera ocurrencia que se encuentre de lo
'que se ha digitado en el texto
```

```
Data1.Recordset.FindFirst (CmbCampo.Text & " LIKE " & "'" & TxtTele.Text_
                           & "*'")
End Sub

Private Sub TxtTele_KeyPress(KeyAscii As Integer)
  If KeyAscii = 13 Then BtnAceptar.SetFocus
End Sub
```

Lectura Automática

```
Private Opc As Integer ' Variable utilizada para almacenar la opción
seleccionada

Private Sub BtnAceptar_Click()
On Error GoTo CapturaError
 Parar = False
 Select Case Opc
  Case 0 ' Selecciona a todos los usuarios
    Data1.RecordSource = "Usuarios"
    Data1.RecordsetType = vbRSTypeDynaset
    Data1.Refresh
    Data1.Recordset.MoveFirst
    Set ControldeDato = Data1.Recordset
  Case 4 ' Selecciona a los usuarios que se obtienen del filtro
    Set Data1.Recordset = ControldeDato
    Data1.Recordset.MoveFirst
 End Select

 ' este codigo es de la lectura automatica
 Do Until Data1.Recordset.EOF
   ResultLect = 2
   MAC = True
   FormProgreso.Show 1
   Unload FormProgreso
   FormResLec.Show 1
   Data1.Recordset.MoveNext
   If Parar Then Exit Do
 Loop
 ' Se despliega la ventana con las posibles lecturas erroneas
 FormResAut.Show 1

' Captura de errores
Salir:
  Exit Sub
CapturaError:
  MsgBox Err.Description & Chr$(13) & Err
  Resume Salir

End Sub

Private Sub BtnCancelar_Click()
 Unload FormLecAuto
End Sub

Private Sub BtnFiltro_Click()
```

```
Set ControldeDato = Data1.Recordset
 FormFiltro.Show 1
 Set Data1.Recordset = ControldeDato
 Data1.Recordset.MoveFirst
 BtnAceptar.SetFocus
End Sub


Private Sub BtnFiltro_MouseDown(Button As Integer, Shift As Integer, X As
Single, Y As Single)
 Option1(4).Value = True
End Sub

Private Sub Form_Load()
On Error GoTo CapturaError
' El siguiente código le asigna la Base de Datos al control Data1 de este
' formulario
 Data1.DatabaseName = App.Path + "\LACEE.MDB"
 Data1.RecordSource = "Usuarios"
 Data1.RecordsetType = vbRSTypeDynaset
 Data1.Refresh
 Data1.Recordset.MoveFirst

' Captura de errores
Salir:
  Exit Sub
CapturaError:
  MsgBox Err.Description & Chr$(13) & Err
  Resume Salir
End Sub

Private Sub Option1_Click(Index As Integer)
 Opc = Index
 Select Case Index
  Case 0
  Case 4
    BtnFiltro_Click
 End Select
End Sub
```

Selecciona Grupo de Registros

```
Private Sub BtnAceptar_Click()
 TxtParametro_KeyPress (13)
 If Data1.Recordset.RecordCount > 0 Then Set ControldeDato = _
      Data1.Recordset
 Unload FormFiltro
End Sub

Private Sub BtnCancelar_Click()
 Unload FormFiltro
End Sub

Private Sub CmbCampo_Change()
 TxtParametro.SetFocus
End Sub
```

```
Private Sub Form_Load()
 Set Data1.Recordset = ControldeDato
 CmbCampo.Text = CmbCampo.List(0)
End Sub



Private Sub TxtParametro_KeyPress(KeyAscii As Integer)
 Dim MSQL As String
 If KeyAscii = 13 Then
   MSQL = "SELECT * FROM Usuarios WHERE "
   MSQL = MSQL & CmbCampo.Text & " LIKE " & "'" & TxtParametro.Text & "'"
   Data1.RecordSource = MSQL
   BtnAceptar.SetFocus
 End If
End Sub
```

Proceso de Comunicación

```
Private Telefono_Usuario As String ' Variable de nivel de modulo
'utilizada para almacenar el numero telefonico con el se realizará la
'conección con el microprocesador
Private Bandera As Boolean  ' Variable que indica cuando se esta enviando
'la lectura del contador
Private Conec As Boolean ' Variable utilizada para activar o desactivar
'los mensajes devueltos por el modem
Private Falla As String ' Variable utilizada para almacenar la falla
'ocurrida en la comunicacion


Private Sub Conectar()
 MAC = False
 On Error GoTo CapturaError
 'Cierra el puerto si esta abierto
 If MSComm1.PortOpen Then
  MSComm1.PortOpen = False
 End If
 'Abre el puerto
 MSComm1.PortOpen = True
'Marca el numero
 MSComm1.Output = "ATDT" & Telefono_Usuario
 'Activa la lectura de los mensajes de control devueltos por el modem
 Conec = True ' Activa la lectura de mesajes devueltos por el modem
Salir:
 Exit Sub
CapturaError:
 If Err = 68 Then
  MsgBox "PUERTO DE COMUNICACIONES : No esta disponible"
 Else
  MsgBox Err.Description & Err
 End If
 If MSComm1.PortOpen Then
   MSComm1.Output = "ATH" & vbCrLf
   MSComm1.PortOpen = False
 End If
```

```
  Bandera = False ' Inicializa la bandera
  Conec = False ' Desactiva la lectura de mesajes devueltos por el modem
  'cierra la ventana de progreso
  BtnCancelar_Click
  Resume Salir
End Sub


Private Sub BtnCancelar_Click()
 MAC = False
 Unload Me
End Sub


Private Sub BtnCancelar_MouseDown(Button As Integer, Shift As Integer, X
As Single, Y As Single)
 If ResultLect = 2 Then
   MAC = False
   Parar = True
   Unload Me
 End If
End Sub

Private Sub Form_Activate()
If MAC Then
 'Se inicializa la variable de ErrorDeLect en falso
 ErrorDeLect = True
 Falla = "0"
 'Se inicializa el modem y se abre el puerto de comunicaciones
On Error GoTo CapturaError
 If MSComm1.PortOpen Then
   MSComm1.PortOpen = False
 End If
 MSComm1.CommPort = Puerto
 MSComm1.Settings = Configuracion
 MSComm1.PortOpen = True
 Bandera = False ' Inicializa la bandera
 Conec = False ' Desactiva la lectura de mesajes devueltos por el modem
 Timer1.Enabled = True
 Select Case ResultLect
    Case 1 ' Fue llamado por Lectura Manual
     'se realiza la conección con el microprocesador
     Telefono_Usuario = FormLecMan.Data1.Recordset.Fields("Telefono")
     Conectar
    Case 2 ' Fue llamado por Lectura Automatica
     'se realiza un ciclo para realizar la coneccion conel
     'microprocesador por cada usuario seleccionado
     'se realiza la coneccion con el microprocesador
     Telefono_Usuario = FormLecAuto.Data1.Recordset.Fields("Telefono")
     Conectar
 End Select
'Captura de errores
Salir:
 Exit Sub
CapturaError:
 MsgBox Err.Description & Chr$(13) & Err
 'Limpia el buffer de recepcion
```

```
 MSComm1.InBufferCount = 0
 'Desconectar
  If MSComm1.PortOpen Then
   MSComm1.Output = "ATH" & vbCrLf
   MSComm1.PortOpen = False
  End If
  Bandera = False ' Inicializa la bandera
  Conec = False ' Desactiva la lectura de mesajes devueltos por el modem
  'cierra la ventana de progreso
  BtnCancelar_Click
  Resume Salir
End If
End Sub

Private Sub Form_Unload(Cancel As Integer)
  Dim MSG As Variant
   MAC = False
  'Si elpuerto esta aun abierto entonces se cierra
  If MSComm1.PortOpen Then
   MSComm1.Output = "ATH" & vbCrLf
   MSComm1.PortOpen = False
  End If
  'Si se obtuvo alguna lectura se guarda el valor
  Lectura = (Val(Text2.Text) / 10)
  'Si ocurrio algun error en la lectura, se muestra el mensaje
  If ResultLect = 1 Then
    Select Case Falla
      Case "3" 'No Carrier
        MSG = MsgBox("No hay señal portadora", vbCritical, "")
      Case "4" 'Error
        MSG = MsgBox("Error en la comunicación", vbCritical, "")
      Case "6" 'No Dialtone
        MSG = MsgBox("No hay señal o tono de marcación", vbCritical, "")
      Case "7" 'Busy
        MSG = MsgBox("La línea del usuario esta ocupada", vbCritical, "")
      Case "8" 'No Answer
        MSG = MsgBox("La Unidad de Lectura no responde", vbCritical, "")
    End Select
   End If
End Sub


Private Sub MSComm1_OnComm()
 Dim EVMsg$
 Dim ERMsg$
 Static CONTADOR As Integer
 ' El valor de la propiedad CommEvent nos indica la naturaleza
 ' del evento o error ocurrido.
 Select Case MSComm1.CommEvent
  ' Eventos.
  Case comEvReceive
    Dim Buffer As Variant, Dummy1 As Integer
    Dim NumCar As String
    If Err Then MsgBox Err.Description, 48
    Do While MSComm1.PortOpen
     NumCar = MSComm1.InBufferCount
     If NumCar Then
```

```
Buffer = MSComm1.Input
Text1.SelText = Left(Buffer, NumCar)

If Conec Then
  Select Case Left(Buffer, NumCar)
     Case "0" ' OK
     Case "1" ' CONNECT
         Bandera = False ' Inicializa la bandera
         Text2.Text = ""
         MSComm1.Output = "u" & vbCrLf  ' Solicito lectura
         MSComm1.Output = "U" & vbCrLf  ' Solicito lectura
     Case "2" ' RING
     Case "3" ' NO CARRIER
         Falla = "3"
         CONTADOR = 0
         'Desconectar
         If MSComm1.PortOpen Then
          MSComm1.Output = "ATH" & vbCrLf
          MSComm1.PortOpen = False
         End If
         Bandera = False ' Inicializa la bandera
         Conec = False ' Desactiva la lectura de mesajes devueltos
         'por el modem
         'cierra la ventana de progreso
         BtnCancelar_Click
     Case "4" ' ERROR
         Falla = "4"
         CONTADOR = 0
         'Desconectar
         If MSComm1.PortOpen Then
          MSComm1.Output = "ATH" & vbCrLf
          MSComm1.PortOpen = False
         End If
         Bandera = False ' Inicializa la bandera
         Conec = False ' Desactiva la lectura de mesajes devueltos
         'por el modem
         'cierra la ventana de progreso
         BtnCancelar_Click
     Case "5" ' CONNECT 1200"
         Bandera = False ' Inicializa la bandera
         Text2.Text = ""
         MSComm1.Output = "u" & vbCrLf  ' Solicito lectura
         MSComm1.Output = "U" & vbCrLf  ' Solicito lectura
     Case "6" ' NO DIALTONE
         Falla = "6"
         CONTADOR = 0
         'Desconectar
         If MSComm1.PortOpen Then
          MSComm1.Output = "ATH" & vbCrLf
          MSComm1.PortOpen = False
         End If
         Bandera = False ' Inicializa la bandera
         Conec = False ' Desactiva la lectura de mesajes devueltos
         'por el modem
         'cierra la ventana de progreso
         BtnCancelar_Click
      Case "7" ' BUSY
```

```
                Falla = "7"
                CONTADOR = 0
                'Desconectar
                If MSComm1.PortOpen Then
                 MSComm1.Output = "ATH" & vbCrLf
                 MSComm1.PortOpen = False
                End If
                Bandera = False ' Inicializa la bandera
                Conec = False ' Desactiva la lectura de mesajes devueltos
                'por el modem
                'cierra la ventana de progreso
                BtnCancelar_Click
             Case "8" ' NO ANSWER
                Falla = "8"
                CONTADOR = 0
                'Desconectar
                If MSComm1.PortOpen Then
                 MSComm1.Output = "ATH" & vbCrLf
                 MSComm1.PortOpen = False
                End If
                Bandera = False ' Inicializa la bandera
                Conec = False ' Desactiva la lectura de mesajes devueltos
                'por el modem
                'cierra la ventana de progreso
                BtnCancelar_Click
          End Select
       End If

       If Left(Buffer, NumCar) = "/" Then
          Bandera = Not Bandera
          Conec = Not Conec
          CONTADOR = CONTADOR + 1
       End If

       If CONTADOR = 2 Then
        CONTADOR = 0
        MAC = False
        ErrorDeLect = False
         'Desconectar
          If MSComm1.PortOpen Then
            MSComm1.Output = "ATH" & vbCrLf
            MSComm1.PortOpen = False
          End If
          Bandera = False ' Inicializa la bandera
          Conec = False ' Desactiva la lectura de mesajes devueltos por el
          'modem
          'cierra la ventana de progreso
          BtnCancelar_Click
        End If
      End If
      Dummy1 = DoEvents()
     Loop
End Sub

Private Sub Timer1_Timer()
 Static SW As Boolean
 If SW Then
```

```
  Line1(1).BorderStyle = 4
  Line1(0).BorderStyle = 5
  SW = Not (SW)
 Else
  Line1(1).BorderStyle = 5
  Line1(0).BorderStyle = 4
  SW = Not (SW)
 End If
 If BarraProgreso.Value < 100 Then
  BarraProgreso.Value = BarraProgreso.Value + 10
 Else
  BarraProgreso.Value = 0
 End If
End Sub
```

Resultado de la Lectura

```
Private Sub BtnAceptar_Click()
On Error GoTo CapturaError
If (Not ErrorDeLect) Or (ResultLect = 2) Then
 Data1.Recordset.AddNew
 With Data1.Recordset
    .Fields("Telefono") = TxtTelefono
    .Fields("Fecha") = Date
    .Fields("Hora") = Time
    .Fields("Consumo") = Val(TxtConsumo)
    .Fields("LecAnte") = Val(TxtLecAnt)
    .Fields("LecActu") = Val(TxtLecAct)
 End With
 Data1.Recordset.Update
End If
 Unload FormResLec
' Captura de errores
Salir:
  Exit Sub
CapturaError:
  MsgBox Err.Description & Chr$(13) & Err
  Resume Salir
End Sub

Private Sub Form_Activate()
 If ResultLect = 2 Then BtnAceptar_Click
 Unload FormProgreso
End Sub

Private Sub Form_Load()
On Error GoTo CapturaError
 Dim MiRS As Recordset
 'Centra el formulario en pantalla
 Me.Top = ((MDIFormPpal.ScaleHeight - Me.ScaleHeight) / 2)
 Me.Left = ((MDIFormPpal.ScaleWidth - Me.ScaleWidth) / 2)
 Select Case ResultLect
 Case 1 ' lectura manual
   TxtNombre.Text = FormLecMan.Data1.Recordset.Fields("Nombre")
   TxtDireccion.Text = FormLecMan.Data1.Recordset.Fields("Direccion")
   TxtTelefono.Text = FormLecMan.Data1.Recordset.Fields("Telefono")
```

```
 Case 2 ' esto es de prueba de lectura automatica
   TxtNombre.Text = FormLecAuto.Data1.Recordset.Fields("Nombre")
   TxtDireccion.Text = FormLecAuto.Data1.Recordset.Fields("Direccion")
   TxtTelefono.Text = FormLecAuto.Data1.Recordset.Fields("Telefono")
   Me.Left = MDIFormPpal.Width + 100
 End Select

 Data1.DatabaseName = App.Path + "\LACEE.MDB"
 Data1.RecordSource = "Consumos"
 Data1.RecordsetType = vbRSTypeDynaset
 Data1.Refresh
 MSQL = "SELECT Count(*) AS Numero FROM Consumos WHERE "
 MSQL = MSQL & "Telefono = '" & TxtTelefono.Text & "'"
 Data1.RecordSource = MSQL
 Data1.Refresh
 Set MiRS = Data1.Recordset
 MSQL = "SELECT * FROM Consumos WHERE "
 MSQL = MSQL & "Telefono = '" & TxtTelefono.Text & "'"
 Data1.RecordSource = MSQL
 Data1.Refresh
 If MiRS.Fields("Numero") > 0 Then
  Data1.Recordset.MoveLast
  TxtLecAnt.Text = Str(Data1.Recordset.Fields("LecActu"))
 Else
  TxtLecAnt.Text = "0"
 End If
 'Se calcula el valor del consumo
 TxtLecAct.Text = Str(Lectura)
 If Lectura >= Val(TxtLecAnt.Text) Then
  TxtConsumo.Text = Str(Val(TxtLecAct.Text) - Val(TxtLecAnt.Text))
 Else
  TxtConsumo.Text = Str((99999.9 - Val(TxtLecAnt.Text)) + Lectura)
 End If
 'Si ocurrio un error de lectura en la comunicacion se cambia el titulo
del botón
 If ErrorDeLect Then BtnAceptar.Caption = "Cerrar"
' Captura de errores
Salir:
  Exit Sub
CapturaError:
  MsgBox Err.Description & Chr$(13) & Err
  Resume Salir
End Sub

Private Sub BtnCerrar_Click()
 Unload FormResAut
End Sub

Private Sub Form_Load()
Dim MSQL As String
Label1.Caption = "Estas lecturas pueden ser causa de errores como:" +
Chr(13) + "     No hay tono para marcar."
Label1.Caption = Label1.Caption + Chr(13) + "     No hay señal
portadora."
Label1.Caption = Label1.Caption + Chr(13) + "     La línea del usuario
esta ocupada."
```

```
Label1.Caption = Label1.Caption + Chr(13) + "     La Unidad de Lectura no
responde."
Label1.Caption = Label1.Caption + Chr(13) + "     Algún error en la
transmisión."
On Error GoTo CapturaErr
'Secrea una consulta temporal para almacenar los datos del reporte
 MSQL = "SELECT"
 MSQL = MSQL & " Fecha AS [FECHA_], Consumos.Telefono AS [TELEFONO_],"
 MSQL = MSQL & " Usuarios.Nombre AS [NOMBRES Y APELLIDOS],_
Usuarios.Direccion AS [DIRECCION_],"
 MSQL = MSQL & " LecActu AS [LECTURA ACTUAL], LecAnte AS [LECTURA_
ANTERIOR],"
 MSQL = MSQL & " Consumo AS [CONSUMO_] "
 MSQL = MSQL & " FROM Usuarios INNER JOIN Consumos ON Usuarios.Telefono_
   = Consumos.Telefono"
 MSQL = MSQL & " WHERE LecActu = 0"
 Data1.DatabaseName = App.Path + "\LACEE.MDB"
 Data1.RecordSource = MSQL
 Exit Sub
CapturaErr:
     Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
     Exit Sub
End Sub


Private Sub Form_Unload(Cancel As Integer)
On Error GoTo CapturaErr
 'Este código elimina de la base de datos las lecturas fallidas
 If Data1.Recordset.RecordCount > 0 Then
   Do Until Data1.Recordset.EOF
     Data1.Recordset.Delete
     Data1.Recordset.MoveNext
   Loop
 End If
 Exit Sub
CapturaErr:
     Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
     Exit Sub
End Sub
```

Crear, Insertar, Modificar y Eliminar

```
Private Sub BtnAceptar_Click()
 On Error GoTo CapturaErr
 Select Case OpcMnu
   Case 1, 2 'Crear o Insertar
     Data1.Recordset.AddNew
     TxtNombre = ""
     TxtDireccion = ""
     TxtTelefono = ""
     TxtNombre.SetFocus
   Case 3 'Modificar
     Dim mens3 As String
     mensage = "¿Está seguro de querer MODIFICAR este registro?"
     retcode = MsgBox(mensage, vbYesNo + vbQuestion, "Confirmación de_
```

```
      Modificación")
     'Modifica el registro si escoge la opción "SI"
     If retcode = vbYes Then
       Data1.Recordset.Edit
       Data1.UpdateRecord
     Else
       Data1.UpdateControls
     End If
   Case 4 'Eliminar
     Dim mens4 As String
     mensage = "¿Está seguro de querer ELIMINAR este registro?"
     retcode = MsgBox(mensage, vbYesNo + vbQuestion, "Confirmación de_
     Eliminación")
     'Elimina el registro actual si escoge la opción "SI"
     If retcode = vbYes Then Data1.Recordset.Delete
     Data1.Refresh
   Case 5 'Consultar
 End Select
 Exit Sub
CapturaErr:
  Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
  Err = 0
 Exit Sub
End Sub

Private Sub BtnBuscar_Click()
 Data1.BOFAction = 1
 Set ControldeDato = Data1.Recordset
 FormBuscar.Show 1
 Data1.BOFAction = 0
End Sub

Private Sub BtnCancelar_Click()
 Unload FormOpcMnuBD
End Sub

Private Sub Form_Load()
 On Error GoTo CapturaErr
 'El siguiente código asigna la base de datos al el Control de Datos
 Data1.DatabaseName = App.Path + "\LACEE.MDB"
 Data1.RecordSource = "Usuarios"
 Data1.RecordsetType = vbRSTypeDynaset
 Data1.Recordset.AddNew
 'Se asignan los campos correspondientes a cada uno de los textos
 TxtNombre.DataField = "Nombre"
 TxtDireccion.DataField = "Direccion"
 TxtTelefono.DataField = "Telefono"

 'Segun la opción escogida en el menú
 Select Case OpcMnu
   Case 1 'Crear
     FormOpcMnuBD.Caption = "Crear"
     Label4.Caption = "Digite en cada campo los datos a CREAR y luego_
      haga clic en el botón ACEPTAR."
     BtnAceptar.Caption = "C&rear"
     BtnBuscar.Visible = False
     'Permite editar y añadir registros en la cuadricula
```

```
      GridBD.AllowAddNew = True
      GridBD.AllowUpdate = True
    Case 2 'Insertar
      FormOpcMnuBD.Caption = "Insertar"
      Label4.Caption = "Digite en cada campo los datos a INSERTAR y luego_
       haga clic en el botón ACEPTAR."
      BtnAceptar.Caption = "&Insertar"
      BtnBuscar.Visible = False
      'Permite editar y añadir registros en la cuadricula
      GridBD.AllowAddNew = True
      GridBD.AllowUpdate = True
    Case 3 'Modificar
     FormOpcMnuBD.Caption = "Modificar"
     Label4.Caption = "Digite en cada campo los datos a MODIFICAR y luego_
     haga clic en el botón ACEPTAR."
     BtnAceptar.Caption = "&Modificar"
     BtnBuscar.Visible = True
     'Mueve el puntero al primer registro de la lista
     Data1.Recordset.MoveFirst
     'Permite editar y Modificar registros en la cuadricula
     GridBD.AllowUpdate = True
    Case 4 'Eliminar
      FormOpcMnuBD.Caption = "Eliminar"
      Label4.Caption = "Digite en cada campo los datos a ELIMINAR y luego_
      haga clic en el botón ACEPTAR."
      BtnAceptar.Caption = "&Eliminar"
      BtnBuscar.Visible = True
      'Inabilita las cajas de texto
      TxtNombre.Locked = True
      TxtDireccion.Locked = True
      TxtTelefono.Locked = True
      'Mueve el puntero al primer registro de la lista
      Data1.Recordset.MoveFirst
    Case 5 'Consultar
     FormOpcMnuBD.Caption = "Consultar"
     Label4.Caption = "Digite en cada campo los datos a CONSULTAR y luego_
     haga clic en el botón ACEPTAR."
      BtnAceptar.Caption = "C&onsultar"
      BtnBuscar.Visible = True
     'Inabilita las cajas de texto
      TxtNombre.Locked = True
      TxtDireccion.Locked = True
      TxtTelefono.Locked = True
      'Mueve el puntero al primer registro de la lista
      Data1.Recordset.MoveFirst
 End Select
 Exit Sub
CapturaErr:
  Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
 Exit Sub
End Sub

Private Sub TxtDireccion_KeyPress(KeyAscii As Integer)
  If KeyAscii = 13 Then TxtTelefono.SetFocus
End Sub

Private Sub TxtNombre_KeyPress(KeyAscii As Integer)
```

```
  If KeyAscii = 13 Then TxtDireccion.SetFocus
End Sub

Private Sub TxtTelefono_KeyPress(KeyAscii As Integer)
  If KeyAscii = 13 Then
    Dim Numero As Long
    Numero = IsNumeric(TxtTelefono.Text)
    If Not Numero Then
      MSG = "El teléfono no puede estar en blanco," + Chr(13) + "ni_
      puede contener letras o espacios."
      MSG = MsgBox(MSG, vbExclamation, "")
      Exit Sub
    Else
     BtnAceptar.SetFocus
    End If
  End If
End Sub
```

Buscar

```
Private Sub BtnBusSig_Click()
 Static SW As Variant
 If SW = 0 Then
  ControldeDato.FindFirst (CmbCampo.Text & " LIKE " & "'*" &_
      TxtBuscar.Text & "*'")
  SW = 1
 Else
  ControldeDato.FindNext (CmbCampo.Text & " LIKE " & "'*" &_
      TxtBuscar.Text & "*'")
  If ControldeDato.NoMatch Then
    MSG = ("Se ha llegado al fin al de la tabla." & Chr(13) & "¿Desea_
      continuar?")
    MSG = MsgBox(MSG, vbYesNo + vbQuestion, "")
    If MSG = vbYes Then
      ControldeDato.MoveFirst
      SW = 0
      BtnBusSig_Click
    Else
      Exit Sub
    End If
  End If
 End If
  TxtEncontro.Text = ControldeDato(CmbCampo.Text)
End Sub

Private Sub BtnCerrar_Click()
 SW = 0
 Unload FormBuscar
End Sub

Private Sub Form_Load()
  CmbCampo.Text = CmbCampo.List(0)
End Sub

Private Sub TxtBuscar_KeyPress(KeyAscii As Integer)
 If KeyAscii = 13 Then BtnBusSig.SetFocus
End Sub
```

Configurar Puerto de Comunicaciones

```
'Este codigo se ejecuta cuando se hace clic en el botón <Aceptar>
Private Sub BtnAceptar_Click()
 Dim X As Integer
 Data2.Recordset.Edit
 For X = 0 To 3
   If PuertoCom(X).Value Then Exit For
 Next X
 Data2.Recordset.Fields("Puerto").Value = X + 1
 Data2.UpdateRecord
 Puerto = X + 1
 Unload FormConfig  'Descarga el formulario de la memoria
End Sub



'Este codigo se ejecuta cuando se hace clic en el botón <Aceptar>
Private Sub BtnCancelar_Click()
 Unload FormConfig
End Sub

'Este codigo se ejecuta cuando se carca el formulario en memoria
Private Sub Form_Load()
 Dim BD As Database, WS As Workspace
 Dim Tabla1 As TableDef
 Dim Cam1 As Field
 'Actualiza los valores de la Configuracion anterior
 On Error GoTo CapturaErr
 'Abre la tabla de Configuraciones y se la asigna al control de datos
 Data2.DatabaseName = App.Path + "\CONFIG.MAC"
 Data2.RecordSource = "Configuracion"
 Data2.RecordsetType = vbRSTypeDynaset
 Data2.Recordset.AddNew
 If Data2.Recordset.RecordCount = 0 Then Data2.UpdateRecord
 Data2.Recordset.MoveFirst
 'Almacena el valor del puerto que esta almacenodo el archivo CONFIG.MAC
 Puerto = Data2.Recordset.Fields("Puerto").Value
 If Puerto = 0 Then
  Puerto = 2 'FormProgreso.MSComm1.CommPort 'MDIFormPpal.MSComm1.CommPort
 End If
 Configuracion = "1200,N,8,1"
 PuertoCom(Puerto - 1).Value = True
Exit Sub
CapturaErr:
 Select Case Err
  Case 3024 ' La base de datos Config.Mac no existe
    Dim captura As Variant
    captura = Err
    'Se crea la base de Datos
    Set WS = DBEngine.Workspaces(0)
    NomBD = App.Path + "\CONFIG.MAC"
    Set BD = WS.CreateDatabase(NomBD, dbLangGeneral)
    Set Tabla1 = BD.CreateTableDef("Configuracion")
    Set Cam1 = Tabla1.CreateField("Puerto", dbInteger, 3)
```

```
    Cam1.DefaultValue = 0
    Tabla1.Fields.Append Cam1
    BD.TableDefs.Append Tabla1
    BD.Close
    Resume Next
  Case Else
    Resume Next
  End Select
End Sub
```

Reportes 6 Primeros / Ultimos Consumos

```
Private Sub BtnBuscar_Click()
 Data1.BOFAction = 1
 Set ControldeDato = Data1.Recordset
 FormBuscar.Show 1
 Data1.BOFAction = 0
End Sub

Private Sub BtnCancelar_Click()
 Unload FormReportesPC
End Sub

Private Sub BtnGenerar_Click()
Dim Telef, MSQL, Encabezado As String
On Error GoTo CapturaErr
'Este codigo genera el reporte
 Telef = Data1.Recordset.Fields("Telefono").Value
'Secrea una consulta temporal para almacenar los datos del reporte
 MSQL = "SELECT TOP 6"
 MSQL = MSQL & " Consumos.Numero AS [Nro], Fecha AS [FECHA_],_
      Consumos.Telefono AS [TELEFONO_], Hora AS [HORA_],"
 MSQL = MSQL & " Usuarios.Nombre AS [NOMBRES Y APELLIDOS],_
      Usuarios.Direccion AS [DIRECCION_],"
 MSQL = MSQL & " LecActu AS [LECTURA ACTUAL], LecAnte AS [LECTURA_
      ANTERIOR],"
 MSQL = MSQL & " Consumo AS [CONSUMO_] "
 MSQL = MSQL & " FROM Usuarios INNER JOIN Consumos ON Usuarios.Telefono_
      = Consumos.Telefono"
 MSQL = MSQL & " WHERE Consumos.Telefono = '" & Telef & "'"
 'Segun la opción tomada se ejecuta un tipo de consulta
 Select Case OpcMnu
   Case 1 '6 Primeros Consumos
     MSQL = MSQL & " ORDER BY Consumos.Numero ASC"
     Reporte.BoundReportHeading = "6 PRIMEROS CONSUMOS"
   Case 2 '6 Ultimos Consumos
     MSQL = MSQL & " ORDER BY Consumos.Numero DESC"
     Reporte.BoundReportHeading = "6 ULTIMOS CONSUMOS"
 End Select

 Data2.DatabaseName = App.Path + "\LACEE.MDB"
 Data2.RecordSource = MSQL
'SE CAMBIAN ALGUNAS PROPIEDADES DEL REPORTE
 Reporte.ReportFileName = App.Path + "\PRIMEROS6CONS.RPT"
 Reporte.RetrieveDataFiles
 Exit Sub
```

```
CapturaErr:
     Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
     Exit Sub
End Sub

Private Sub Form_Load()
On Error GoTo CapturaErr
 'El siguiente código asigna la base de datos al el Control de Datos
 Data1.DatabaseName = App.Path + "\LACEE.MDB"
 Data1.RecordSource = "Usuarios"
 Data1.RecordsetType = vbRSTypeDynaset
 Data1.Refresh
 Data1.Recordset.AddNew
 Data1.Recordset.MoveFirst
 'Segun la opción escogida en el menú
 Select Case OpcMnu
   Case 1 '6 Primeros Consumos
     FormReportesPC.Caption = "Reportes 6 Primeros Consumos"
   Case 2 '6 Ultimos Consumos
     FormReportesPC.Caption = "Reportes 6 Ultimos Consumos"
 End Select
  Exit Sub
CapturaErr:
  Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
 Exit Sub
End Sub

Private Sub Option1_Click(Index As Integer)
 Reporte.Destination = Index
End Sub
```

Reporte Consumo Promedio

```
Public MododeReporte As Integer ' Variable a nivel global a nievel de
                                 ' modulo utilizada para determinar el
                                 'modo del reporte (Grafico o Listado)
Public Destinacion As Integer  ' Variable global a nivel de modulo
                               'utilizada para determinar el lugar del
                               'reporte
Public TipoReporte As Integer  ' Variable global a nivel de modulo
                               'utilizada para determinar que tipo de
                               'grafico se mostrará ( Barras, Líneas o
                               'Sectores)

Private Sub BtnBuscar_Click()
 Data1.BOFAction = 1
 Set ControldeDato = Data1.Recordset
 FormBuscar.Show 1
 Data1.BOFAction = 0
End Sub

Private Sub BtnCancelar_Click()
 Unload FormConProm
End Sub

Private Sub BtnGenerar_Click()
```

```
Dim Telef, MSQL As String
Dim Promedio, Suma As Double ' Variables utilizadas para calcular el
consumo promedio
Dim ParteEntera, ParteDecimal As Variant
On Error GoTo CapturaErr
'Este codigo genera el reporte de Consumo Promedio
 Telef = Data1.Recordset.Fields("Telefono").Value

'Secrea una consulta temporal para almacenar los datos del reporte
 MSQL = "SELECT"
 MSQL = MSQL & " Fecha AS [FECHA_], Hora AS [HORA_], Nombre, Direccion,
 MSQL = MSQL & " LecActu AS [LECTURA ACTUAL], LecAnte AS [LECTURA
 MSQL = MSQL & " Consumo AS [CONSUMO_] "
 MSQL = MSQL & " FROM Usuarios INNER JOIN Consumos ON Usuarios.Telefono =
 MSQL = MSQL & " WHERE Consumos.Telefono = '" & Telef & "'"
 'Se actualiza el control de datos
 Data2.DatabaseName = App.Path + "\LACEE.MDB"
 Data2.RecordSource = MSQL
 Select Case MododeReporte
   Case 0 ' Reporte modo gráfico
    'Se modifican algunas propiedades del grafico segun el tipo
    'seleccionado
     If TipoReporte = 2 Then
        FormGrafico.Grafico.DataReset = 3
        FormGrafico.Grafico.BottomTitle = ""
        FormGrafico.Grafico.ExtraData = 1
        Data2.Recordset.MoveLast
        Data2.Recordset.MoveFirst
        Do Until Data2.Recordset.EOF
        FormGrafico.Grafico.LegendText =_
            (Data2.Recordset.Fields("FECHA_"))
          Data2.Recordset.MoveNext
        Loop
     Else
        FormGrafico.Grafico.DataReset = 5
        FormGrafico.Grafico.BottomTitle = "FECHA"
     End If
     FormGrafico.Grafico.GraphType = TipoReporte
     FormGrafico.Grafico.DrawMode = Destinacion
     'Se calcula el consumo promedio
     Data2.Recordset.MoveLast
     Data2.Recordset.MoveFirst
     FormGrafico.Grafico.ExtraData = Data2.Recordset.RecordCount
     FormGrafico.Grafico.NumPoints = Data2.Recordset.RecordCount
     FormGrafico.Grafico.NumSets = 1
     FormGrafico.Grafico.GridStyle = gphBoth
     Select Case TipoReporte
       Case 2, 4
         X = 1
       Case 6
         X = 0
     End Select
     FormGrafico.Text1 = ""
     FormGrafico.Text1.Text = "LISTADO DE FECHAS POR CONSUMO" & vbCrLf
     FormGrafico.Text2(0).Text = Data2.Recordset.Fields("Telefono")
     FormGrafico.Text2(1).Text = Data2.Recordset.Fields("Nombre")
     FormGrafico.Text2(2).Text = Data2.Recordset.Fields("Direccion")
```

```
     Do Until Data2.Recordset.EOF
       Suma = Suma + Data2.Recordset.Fields("CONSUMO_")
       FormGrafico.Grafico.GraphData =_
            Int(Data2.Recordset.Fields("CONSUMO_"))
       FormGrafico.Text1 = FormGrafico.Text1 + Str(X) + "          " +_
            (Data2.Recordset.Fields("FECHA_")) + vbCrLf
       Data2.Recordset.MoveNext
       X = X + 1
     Loop

     If Data2.Recordset.RecordCount > 0 Then
       ParteEntera = (Suma \ Data2.Recordset.RecordCount)
       ParteDecimal = (Suma Mod Data2.Recordset.RecordCount)
       Promedio = ParteEntera + (ParteDecimal / 10)
     End If
     'Se genera el reporte grafico
     FormGrafico.Grafico.GraphTitle = "CONSUMO PROMEDIO = " &_
            Str(Promedio) & "(KWH)"
     FormGrafico.Grafico.LeftTitle = "Consumo (KWH)"
     FormGrafico.Grafico.BottomTitle = "Fecha"
     FormGrafico.Show 1

   Case 1 ' Reporte modo listado
    'SE CAMBIAN ALGUNAS PROPIEDADES DEL REPORTE
    Reporte.ReportFileName = App.Path + "\CONSUMOPROMEDIO.RPT"
    Reporte.RetrieveDataFiles
    'Se ejecuta el control de reportes
    Reporte.Action = 1
 End Select
 Exit Sub
CapturaErr:
     Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
     Exit Sub
End Sub

Private Sub CmbGrafico_Click()
 Select Case CmbGrafico.Text
  Case "Líneas"
    TipoReporte = (6)
  Case "Barras"
    TipoReporte = (4)
  Case "Sectores"
    TipoReporte = (2)
 End Select
End Sub

Private Sub Form_Load()
On Error GoTo CapturaErr
 'El siguiente código asigna la base de datos al el Control de Datos
 Data1.DatabaseName = App.Path + "\LACEE.MDB"
 Data1.RecordSource = "Usuarios"
 Data1.RecordsetType = vbRSTypeDynaset
 Data1.Refresh
 Data1.Recordset.AddNew
 Data1.Recordset.MoveFirst
 '********************
 MododeReporte = 1
```

```
 TipoReporte = 6
 CmbGrafico.Text = CmbGrafico.List(1)
 Destinacion = 2
 Exit Sub
CapturaErr:
  Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
 Exit Sub
End Sub


Private Sub Option1_Click(Index As Integer)
   MododeReporte = Index  ' Se captura el modo del reporte
                          ' (0-Gráfico  1-Listado)
End Sub

Private Sub Option2_Click(Index As Integer)
  Reporte.Destination = Index
  Select Case Index
    Case 0
      Destinacion = 2
    Case 1
      Destinacion = 5
  End Select
End Sub
```

Reporte Especifico

```
Private Sub BtnCancelar_Click()
 Unload FormConEsp
End Sub

Private Sub BtnGenerar_Click()
Dim MSQL, Encabezado As String
On Error GoTo CapturaErr
'Este codigo genera el reporte
'Secrea una consulta temporal para almacenar los datos del reporte
 MSQL = MSQL & "SELECT Fecha AS [FECHA_], Hora AS [HORA_],_
      Consumos.Telefono AS [TELEFONO_],"
 MSQL = MSQL & " Usuarios.Nombre AS [NOMBRES Y APELLIDOS],_
      Usuarios.Direccion AS [DIRECCION_],"
 MSQL = MSQL & " LecActu AS [LECTURA ACTUAL], LecAnte AS [LECTURA_
      ANTERIOR],"
 MSQL = MSQL & " Consumo AS [CONSUMO_] "
 MSQL = MSQL & " FROM Usuarios INNER JOIN Consumos ON Usuarios.Telefono_
      = Consumos.Telefono"
 MSQL = MSQL & " WHERE Consumos.Fecha = '" & (TxtFecha.Text) & "'"
 'Se actualiza la consulta en el control de datos
 Data1.DatabaseName = App.Path + "\LACEE.MDB"
 Data1.RecordSource = MSQL
 Data1.Refresh
 'SE CAMBIAN ALGUNAS PROPIEDADES DEL REPORTE
 Reporte.ReportFileName = App.Path + "\CONSUMOESPECIFICO.RPT"
 Reporte.RetrieveDataFiles
 'Se ejecuta el control de reportes
 Reporte.Action = 1

 Exit Sub
```

```
CapturaErr:
     Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
     Exit Sub
End Sub


Private Sub Form_Load()
On Error GoTo CapturaErr
 'El siguiente código asigna la base de datos al el Control de Datos
 Data1.DatabaseName = App.Path + "\LACEE.MDB"
 Data1.Refresh
 Exit Sub
CapturaErr:
  Err = MsgBox(Err.Description + Chr(13) + Str(Err), vbCritical, "")
 Exit Sub
End Sub

Private Sub Option1_Click(Index As Integer)
 Reporte.Destination = Index
End Sub
```