ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE INGENIERÍA

"DISEÑO Y CONSTRUCCION DE UN SISTEMA DIDACTICO DE COMUNICACIÓN INDUSTRIAL BAJO EL PROTOCOLO MODBUS"

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y CONTROL

WILLIAM ANDRES CEVALLOS GUZMAN
MIGUEL FERNANDO MEJIA RIVERA

DIRECTOR: DR. LUIS CORRALES

Quito, Noviembre 2007

DECLARACIÓN

Nosotros, Miguel Fernando Mejía Rivera y William Andrés Cevallos Guzmán, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedemos nuestros derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

_____ _____

William Andrés Cevallos Guzmán Miguel Fernando Mejía Rivera

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Miguel Fernando Mejía Rivera y William Andrés Cevallos Guzmán , bajo mi supervisión.

_____

Dr. Luis Corrales

DIRECTOR DE PROYECTO

AGRADECIMIENTO

Agradecemos a todas las personas que nos han colaborado en la elaboración de este Proyecto, en especial al Dr. Luis Corrales y todas las personas que conforman Ingeniería y Diseño Electrónico por la ayuda prestada.

DEDICATORIA

A Dios, mis Padres, mi Hermana y a mis Amigos por darme la confianza necesaria para realizar todos los objetivos que me he propuesto, uno de ellos el de culminar con éxito en esta prestigiosa institución .

Miguel

A mis Padres, familia y a la Música.

Andrés

CONTENIDO

RESUMEN

En el presente proyecto se realiza un estudio del protocolo de Comunicación MODBUS, en este estudio se ve la utilización de los dos modos de comunicación ASCII y RTU; la manera que cada uno de ellos construye la trama, las diferentes funciones que se puede realizar, los diferentes campos que constituyen la trama y el respectivo chequeo de errores que cada modo (ASCII o RTU) elabora.

Además, se realiza la construcción de un módulo, el módulo consta de un Dispositivo Master que se encarga de la recepción de la trama bajo la norma RS-232 desde la HMI desarrollada en la computadora, y el envió de la trama (trama de petición MODBUS) bajo la norma RS-485 hacia los Dispositivos Esclavos en cualquiera de los modos de comunicación que se encuentre (ASCII o RTU).

Aparte del Dispositivo Master se realizó la construcción de 3 Dispositivos Esclavos, en los cuales se adaptaron 2 periféricos de entrada (un análogo y un digital) y de igual manera, 2 periféricos de salida (un análogo y un digital).

Cada Dispositivo Esclavo al momento de recibir la trama (trama de petición MODBUS) y comprobar bajo los parámetros que establece el protocolo MODBUS, elabora una trama de respuesta o caso contrario de existir errores en función, dirección o registros, el Dispositivo Esclavo elabora un mensaje de excepción. Sea un mensaje de excepción o una trama de repuesta será enviado bajo la norma RS-485 hacia el Dispositivo Master para que este Dispositivo envié la trama hacia la HMI del computador para que esta la procese.

Construido, tanto el Dispositivo Master como los Dispositivos Esclavos se realizarán las pruebas de funcionamiento con lo cual se verificaran los objetivos planteados.

PRESENTACIÓN

El uso que en la actualidad se da a las redes de comunicación industriales ha significado una optimización en recursos debido a que no se necesitan el cableado punto a punto que se realizaba para el manejo y supervisión de los diferentes elementos que intervienen en las instalaciones industriales.

Para el desarrollo de proyecto, el Capitulo 1 consta de un estudio acerca del protocolo de comunicación MODBUS en sus dos modos ASCII Y RTU.

En el capitulo 2, se realiza una explicación de los programas desarrollados en los microcontroladores que se encuentran en el Dispositivo master como también en los Dispositivos Esclavos, se describe e forma detallada el funcionamiento de cada uno de estos programas.

A continuación, en el Capitulo 3, se revisa los aspectos relacionados al diseño y construcción de hardware utilizado. Se describe el diseño de la tarjeta del Dispositivo Master, la tarjeta del Dispositivo esclavo y la tarjeta de periféricos acoplada al Dispositivo Esclavo.

Posteriormente en el Capitulo 4, se procede a realizar las pruebas de todo el sistema para obtener resultados que avalen el diseño realizado.

Finalizando en el Capitulo 5, se realiza las conclusiones y recomendaciones de la construcción del presente proyecto.

CAPÍTULO 1

CARACTERÍSTICAS GENERALES

CAPÍTULO 1

CARACTERÍSTICAS GENERALES

1.1 PROTOCOLO MODBUS[1]

1.1.1 INTRODUCCIÓN AL PROTOCOLO MODBUS

Modbus es un protocolo de comunicaciones en el nivel 7 (Aplicación) del Modelo OSI[2], basado en la arquitectura maestro/esclavo o cliente/servidor, diseñado en 1979 por Modicon para su gama de controladores lógicos programables (PLCs). Convertido en un protocolo de comunicaciones estándar de facto en la industria, es el que goza de mayor disponibilidad para la conexión de dispositivos electrónicos industriales.

Este protocolo define una estructura de mensaje que los controladores reconocerán y usarán, con independencia del tipo de redes sobre la que se comuniquen. Describe el proceso que usa un controlador para pedir acceso a otro dispositivo, cómo responderá a las peticiones desde otros dispositivos y cómo se detectarán y notificarán los errores. Establece un formato común para la disposición y contenido de los campos de mensaje.

Modbus permite el control de una red de dispositivos, por ejemplo un sistema de medida de temperatura y humedad, y comunicar los resultados a un computador. Modbus también se usa para la conexión de un ordenador de supervisión con una unidad remota (RTU) en sistemas de control supervisor de adquisición de datos (SCADA).

---

[1] Parte de esta sección fue tomada de la documentación en http://www.automatas.org/modbus

[2] El **modelo de referencia de Interconexión de Sistemas Abiertos** (OSI, Open System Interconection) lanzado en 1984 fue el modelo de red descriptivo creado por ISO. Proporcionó a los fabricantes un conjunto de estándares que aseguraron una mayor compatibilidad e interoperabilidad entre los distintos tipos de tecnología de red producidos por las empresas a nivel mundial.

Cada dispositivo de la red Modbus posee una dirección única. Cualquier dispositivo puede enviar órdenes Modbus, aunque lo habitual es permitirlo sólo a un dispositivo maestro. Cada comando Modbus contiene la dirección del dispositivo destinatario de la orden. Todos los dispositivos reciben la trama pero sólo el destinatario la ejecuta (salvo un modo especial denominado "Broadcast[3]"). Cada uno de los mensajes incluye información redundante que asegura su integridad en la recepción.

Existe gran cantidad de módems que aceptan el protocolo Modbus y algunos están específicamente diseñados para funcionar con este protocolo. Existen implementaciones para conexión por cable, Wireless, SMS o GPRS. La mayoría de problemas presentados hacen referencia a la latencia y a la sincronización.
Existen versiones del protocolo Modbus para puerto serie y Ethernet (Modbus/TCP).

1.1.1.1 VENTAJAS

Las razones por las cuales el uso de Modbus aventaja a otros protocolos de comunicaciones son:

1. es público
2. su implementación es fácil y requiere poco desarrollo

1.1.1.2 CODIFICACIÓN DE DATOS.

MODBUS usa una representación “big-endian” para direcciones y datos, formato en el cual el byte más significativo se encuentra primero. Esto significa que cuando una cantidad numérica más grande que un byte es transmitida, el byte más significante es enviado primero. Así, por ejemplo:

0x1234 será 0x12 0x34

[3] El modo ***Broadcast*** es un modo de difusión en el cual todos los esclavos conectados a la red reciben y ejecutan el mensaje

1.1.1.3 TRANSACCIONES SOBRE REDES MODBUS

Los puertos estándar Modbus en controladores Modicon utilizan un interface serie compatible RS-232C. La norma EIA RS-232C define las patillas del conector, cableado, niveles de señal, velocidades de transmisión y control de paridad.

Los controladores pueden ser conectados en red directamente o vía módems.

Los controladores comunican usando una técnica maestro – esclavo, en la cual sólo un dispositivo (el maestro) puede iniciar transacciones (llamadas 'peticiones' – 'queries'). Los otros dispositivos (los esclavos) responden suministrando al maestro el dato solicitado, o realizando la acción solicitada en la petición.

Entre los dispositivos maestros típicos se incluyen los procesadores centrales y los paneles de programación. Esclavos típicos son los PLCs (controladores programables). El maestro puede direccionar esclavos individualmente o puede generar un mensaje en modo de difusión a todos los esclavos. Los esclavos devuelven un mensaje (llamado 'respuesta') a las peticiones que les son direccionadas individualmente. No se devuelven respuestas a peticiones en modo difusión enviadas desde el maestro.

El protocolo Modbus establece el formato para la petición del maestro, colocando en ella la dirección del dispositivo esclavo (0 en caso de 'Broadcast'), un código de función que define la acción solicitada, cualquier dato que haya de enviarse y un campo de comprobación de error. El mensaje de respuesta del esclavo está también definido por el protocolo Modbus. Contiene campos confirmando la acción tomada, cualquier dato que haya de devolverse y un campo de comprobación de error. Si el mensaje recibido por el esclavo es defectuoso o el esclavo es incapaz de realizar la acción solicitada, construirá un mensaje de error y lo enviará como respuesta.

El código de función en la petición indica al dispositivo esclavo direccionado el tipo de acción a realizar. Los bytes de datos contienen cualquier información adicional que el esclavo necesitará para llevar a cabo la función. Por ejemplo el código de función 03 pedirá al esclavo que lea registros mantenidos (holding reg.) y responda con sus contenidos. El campo de datos debe contener la información que indique al esclavo en qué registro debe comenzar y cuántos ha de leer. El campo de comprobación de error proporciona un método para que el esclavo valide la integridad del contenido del mensaje recibido.

Si el esclavo elabora una respuesta normal, el código de función contenido en la respuesta es una réplica del código de función enviado en la petición. Los bytes de datos contienen los datos recolectados por el esclavo, tales como valores de registros o estados.

Si ocurre un error, el código de función contenido en la respuesta es diferente al código de función enviado en la petición, para indicar que la respuesta es una respuesta de error y los bytes de datos contienen un código que describe el error. El campo de comprobación de error permite al maestro confirmar que los contenidos del mensaje son válidos.

1.1.1.4 CARACTERÍSTICAS FÍSICAS DE LAS REDES MODBUS[4]

Las comunicaciones MODBUS pueden realizarse sobre enlaces punto a punto o multipunto, depende de si el mensaje se envía en modo *unicast* o *broadcast*.

En la Figura 1.1 se detalla el enlace físico, este puede ser cableado mediante RS-232, RS-422 o RS-485 o puede ser a través de enlaces radiales, típicamente del tipo *spread spectrum*[5], que tienen la ventaja de no requerir licencia para su instalación, o mediante comunicaciones telefónicas vía

[4] Tomado de “Control remoto del sistema B+G del Observatorio Astronómico del Montsec: monitorización de funciones” - Javier Clavero Quílez

[5] spread spectrum: Originalmente desarrollado por los militares, "spread spectrum radio transmission" esencialmente extiende (*spread*) una señal de radio sobre una frecuencia cuyo ancho de banda es grande para hacer difícil las interferencias y dificultar los atascamientos.

MODEM, que incluso contemplan la posibilidad de realizar un control a través de telefonía GSM.

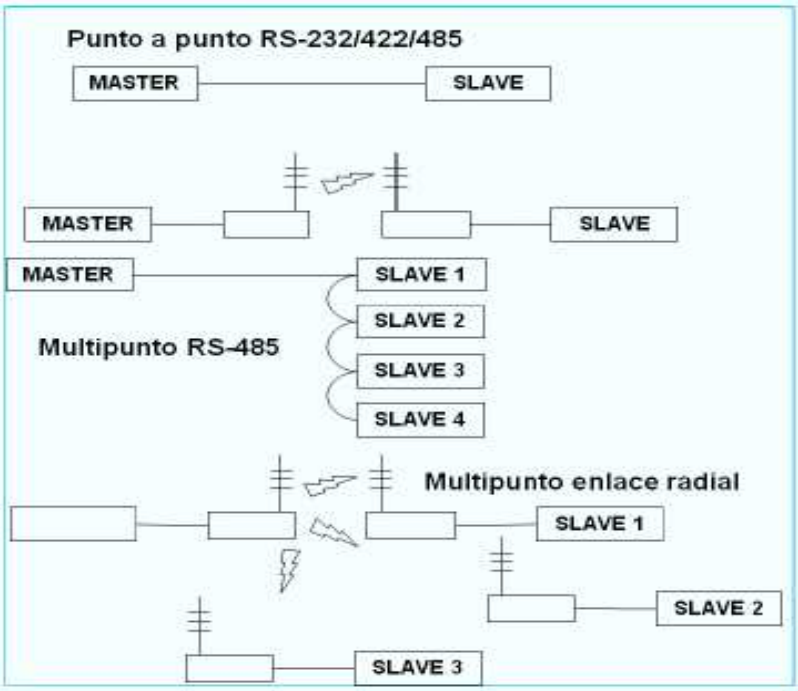

Figura 1.1 MODELOS DE CONEXIÓN FÍSICA DE REDES MODBUS

Los enlaces pueden ser de tipo *half duplex* o *full duplex* indistintamente, pero como se trata de un protocolo tipo *master-slave*, basta con un enlace *half duplex* para su implementación.

Los *baud rates*[6] soportados dependen de los equipos que se usen, pero típicamente van desde 1200 hasta 38400 bps. En este caso, MODBUS trabaja sobre una interfaz serie que puede variar entre una conexión RS232 o RS485.

La interfaz RS485 suele ser la más común, pero puede usarse la RS232 en caso de transmisiones a distancias cortas.

[6] baude rate.- la velocidad de transmisión

1.1.2 LOS DOS MODOS DE TRANSMICIÓN

Los controladores pueden ser configurados para comunicarse sobre redes standard Modbus utilizando cualquiera de los dos modos de transmisión: ASCII o RTU.

Los usuarios seleccionan el modo deseado, junto con los parámetros de comunicación del puerto serie (velocidad, paridad, etc), durante la configuración de cada controlador. El modo y los parámetros serie *deben ser los mismos para todos los dispositivos conectados a una red Modbus*.

La selección del modo ASCII o RTU tiene que ver únicamente con redes Modbus standard. Define los bits contenidos en los campos del mensaje transmitido en forma serie en esas redes. Determina cómo debe ser empaquetada y decodificada, la información en los campos del mensaje.

1.1.2.1 MODO ASCII

Cuando los controladores se configuran para comunicarse en una red Modbus según el modo ASCII (American Standard Code for Information Interchange), cada byte – 8 bits - en un mensaje se envía como dos caracteres ASCII. La principal ventaja de este modo es que permite intervalos de tiempo de hasta un segundo entre caracteres sin dar lugar a error.

El formato para cada byte en modo ASCII es:

- **Sistema de codificación:** Hexadecimal, caracteres ASCII 0-9, A-F.
  Un carácter hexadecimal contenido en cada carácter ASCII del mensaje.

- **Bits por byte:** 1 bit de arranque.
  7 bits de datos, el menos significativo se envía primero.

1 bit para paridad Par o Impar; ningún bit para No paridad.
1 bit de paro si se usa paridad; 2 bits si no se usa paridad.

- **Campo de Comprobación de error:** Comprobación Longitudinal Redundante (LRC).

1.1.2.2 MODO RTU

Cuando los controladores son configurados para comunicarse en una red Modbus usando el modo RTU (Remote Terminal Unit), cada byte de 8 bits en un mensaje contiene dos dígitos hexadecimales de 4 bits. La principal ventaja de este modo es que su mayor densidad de carácter permite mejor rendimiento que el modo ASCII, para la misma velocidad. Cada mensaje debe ser transmitido en un flujo continuo.

El formato para cada byte en modo RTU es:

- **Sistema de codificación:** Binario 8-bits, hexadecimal 0-9, A-F. Dos dígitos hexadecimales contenidos En cada campo de 8 bits del mensaje.

- **Bits por byte:** 1 bit de arranque.
  8 bits de datos, el menos significativo se envía primero.
  1 bit para paridad Par o Impar; ningún bit para No paridad.
  1 bit de paro si se usa paridad; 2 bits si no se usa paridad.

- **Campo de Comprobación de error:** Comprobación Cíclica Redundante (CRC).

Modbus RTU es una representación binaria compacta de los datos. Modbus ASCII es una representación legible del protocolo pero menos eficiente. Ambas implementaciones del protocolo son serie. El formato RTU finaliza la trama con un suma de control de redundancia cíclica (CRC), mientras que el formato ASCII utiliza una suma de control de redundancia longitudinal (LRC).

Los comandos básicos Modbus permiten controlar un dispositivo RTU para modificar el valor de alguno de sus registros o bien solicitar el contenido de dichos registros.

1.1.3 TRAMA DEL MENSAJE MODBUS

En cualquiera de los modos de transmisión serie (ASCII o RTU), un mensaje Modbus es situado por el dispositivo que transmite, en una trama que tiene un comienzo y un final conocidos. Esto permite a los dispositivos receptores comenzar en el arranque del mensaje, leer la parte de la dirección y determinar qué dispositivo es direccionado (o todos los dispositivos si es una difusión 'dirección = 0') y conocer cuándo se ha completado el mensaje.

Los mensajes parciales pueden ser detectados y establecer errores como resultado.

1.1.3.1 TRAMA ASCII

En modo ASCII, los mensajes comienzan con un carácter (:) 'dos puntos' (ASCII 3A hex) y terminan con un par de caracteres (CRLF) 'Retorno de Carro + Avance de Línea) (ASCII 0D hex y 0A hex).

Los caracteres a transmitir permitidos para todos los demás campos son 0-A, A-F hexadecimal. Los dispositivos conectados en red monitorizan el bus de red continuamente para detectar un carácter 'dos puntos'. Cuando se recibe, cada dispositivo decodifica el próximo campo (el campo de dirección) para enterarse si es el dispositivo direccionado. Pueden haber intervalos de hasta un segundo entre caracteres dentro del mensaje. Si transcurre más

tiempo entre caracteres, el dispositivo receptor asume que ha ocurrido un error.

A continuación se muestra una trama de mensaje típica.

INICIO	DIRECCIÓN	FUNCIÓN	DATOS	LRC	FINAL
**1 carácter : **	**2 caracteres**	**2 caracteres**	**N caracteres**	**2 caracteres**	**2 caracteres CRLF**

Figura 1.2 TRAMA DEL MENSAJE ASCII

1.1.3.2 TRAMA RTU

En modo RTU, los mensajes comienzan con un intervalo silencioso de al menos 3.5 tiempos de carácter. Esto es más fácilmente implementado como un múltiplo de tiempos de carácter a la velocidad de transmisión configurada en la red (mostrado como T1-T2-T3-T4 en la Figura 1.2).

El primer campo transmitido es entonces la dirección del dispositivo destinatario. Los caracteres a transmitir permitidos para todos los campos son 0-A, A-F hexadecimal.

Los dispositivos conectados en red monitorizan el bus de red continuamente incluso durante los intervalos ‘silencioso’. Cuando el primer campo (el campo de dirección) es recibido, cada dispositivo lo decodifica para enterarse si es el dispositivo direccionado. Siguiendo al último carácter transmitido, un intervalo de al menos 3.5 tiempos de carácter señala el final del mensaje. Un nuevo mensaje puede comenzar después de este intervalo.

La trama completa del mensaje debe ser transmitida como un flujo continuo. Si un intervalo silencioso de más de 1.5 tiempos de carácter tiene lugar antes de completar la trama, el dispositivo receptor desecha el mensaje incompleto y asume que el próximo byte será el campo de dirección de un nuevo mensaje.

De forma similar, si un nuevo mensaje comienza antes de que transcurran 3.5 tiempos de carácter después de un mensaje previo, el dispositivo receptor lo considerará una continuación del mensaje previo. Esto dará lugar a un error, ya que el valor en el campo final CRC no será válido para el mensaje combinado.

A continuación se muestra una trama de mensaje típica:

INICIO	DIRECCIÓN	FUNCIÓN	DATOS	CRC	FINAL
T1-T2-T3-T4	8 BITS	8 BITS	N * 8 BITS	16 BITS	T1-T2-T3-T4

Figura 1.3 TRAMA DEL MENSAJE RTU

1.1.4 MÉTODOS DE CHEQUEO DE ERRORES

Las redes series estándar Modbus utilizan dos tipos de comprobación de error. La comprobación de paridad (par o impar) puede ser aplicada opcionalmente a cada carácter. La comprobación de la trama (LRC o CRC) es aplicada al mensaje completo. Ambas comprobaciones, de carácter y de trama de mensaje son generadas en el dispositivo maestro y aplicadas a los contenidos del mensaje antes de la transmisión. El dispositivo esclavo comprueba cada carácter y la trama del mensaje completo durante la recepción.

El maestro es configurado por el usuario para aguardar durante un tiempo de espera predeterminado antes de abortar la transacción. Este intervalo es establecido para ser lo suficientemente largo para que cualquier esclavo responda normalmente. Si el esclavo detecta un error de transmisión, el mensaje no será tenido en cuenta. El esclavo no construirá una respuesta para el maestro. Así el tiempo de espera expirará y permite al programa del maestro tratar el error. Se puede observar que un mensaje direccionado a un dispositivo esclavo inexistente también causará un error de tiempo excedido - time out -.

1.1.4.1 CONTROL DE PARIDAD

Los usuarios pueden configurar los controladores para Control de paridad Par o Impar, o Sin Control de paridad. Esto determinará cómo será iniciado el bit de paridad en cada carácter.

Si se especifica cualquier control de paridad Par o Impar, se contabilizará la cantidad de bits que tienen valor 1 en la porción de datos de cada carácter (siete bits de datos para modo ACSII, u ocho para RTU). Al bit de paridad habrá de darse valor 0 o 1, para que se obtenga finalmente un número par o impar, respectivamente, de bits con valor 1.

Por ejemplo, estos 8 bits de datos forman parte de una trama de carácter RTU:

1100 0101

La cantidad de bits de valor 1 en el dato es cuatro. Si se utiliza Control de Paridad Par, el bit de paridad de la trama debe establecerse a valor 0, haciendo que la cantidad de bits de valor 1 siga siendo un número par (cuatro). Si se utiliza Control de Paridad Impar, el bit de paridad deberá tener valor 1, resultando una cantidad de bits de valor 1, impar (cinco).

Cuando el mensaje es transmitido, el bit de paridad es calculado y aplicado a la trama de cada carácter. El dispositivo receptor cuenta la cantidad de bits de valor 1 y establece un error si no coincide la paridad con la configurada para ese dispositivo (*todos los dispositivos en la red Modbus deben ser configurados para usar el mismo método de Control de paridad*).

Entonces se puede concluir para la comprobación de paridad que sólo detecta si un número ***impar*** de bits se han alterado en una trama de carácter durante la transmisión.

Por ejemplo, si se utiliza control de paridad Impar y dos bits de valor 1 de un carácter que tiene en origen 3 bits con valor 1, han quedado falseados (pasan a

valor 0) durante la transmisión, el resultado es todavía un cómputo impar de bits de valor 1 (y por lo tanto el error no es detectado por este método).

Si se especifica control No Paridad, no se transmite bit de paridad y no se hace comprobación de paridad. Se transmite un bit de paro adicional para rellenar la trama de carácter.

1.1.4.2 COMPROBACIÓN LRC

En modo ASCII, los mensajes incluyen un campo de comprobación de error que está basado en un método de ***Comprobación Longitudinal Redundante*** (LRC).

El campo LRC controla el contenido del mensaje, a excepción de los ':' del comienzo y el par CRLF. Es aplicado con independencia de cualquier método de control de paridad utilizado para los caracteres individuales del mensaje.

El campo LRC es un byte, conteniendo un valor binario de ocho bits. El valor LRC es calculado por el dispositivo que realiza la petición (comúnmente el dispositivo Master), añade el LRC al mensaje. El esclavo calcula el LRC durante la recepción del mensaje y compara el valor calculado con el valor recibido en el campo LRC. Si los dos valores no son iguales, resulta un error.

El valor LRC se calcula sumando entre sí los sucesivos bytes del mensaje, descartando cualquier acarreo y luego complementando a dos el valor resultante. Se realiza sobre el contenido del campo de mensaje ASCII excluyendo el carácter ':' de comienzo del mensaje y excluyendo el par CRLF de final de mensaje.

Como ejemplo se calculará el LRC de la siguiente Trama:

- Dirección del esclavo (hex) : 01
- Función (hex): 01

- Campo de dato 1 (hex): 00
- Campo de dato 2 (hex): 00
- Campo de dato 3 (hex): 00
- Campo de dato 4 (hex): 01

Entonces los pasos a seguir serán:

1. Sumar todos los bytes, descartando cualquier acarreo, así:
   01 + 01 + 00 + 00 + 00 + 01 = 03

2. Sacar el complemento a 1 del resultado de la suma, así:
   FF – 03 = FC

3. Sacar el complemento a 2 del resultado anterior, así:
   FC + 01 = FD

1.1.4.3 COMPROBACIÓN CRC

En modo RTU, los mensajes incluyen un campo de comprobación de error que está basado en un método Comprobación de Redundancia Cíclica (CRC). El campo CRC controla el contenido del mensaje completo. Se aplica con independencia de cualquier método de control de paridad utilizado para los caracteres individuales del mensaje.

El campo CRC es de dos bytes. El valor CRC es calculado por el dispositivo que realiza la petición (comúnmente el dispositivo Master), añade el CRC al mensaje. El esclavo calcula el CRC durante la recepción del mensaje y compara el valor calculado con el valor recibido en el campo CRC. Si los dos valores no son iguales, resulta un error.

Para calcular el valor CRC Modbus se precarga un registro de 16 bits, todos ellos a 1. Luego comienza un proceso que toma los sucesivos bytes del mensaje y los opera con el contenido del registro y actualiza éste con el resultado obtenido. Sólo los 8 bits de dato de cada carácter son utilizados

para generar el CRC. Los bits de arranque y paro y el bit de paridad, no se tienen en cuenta para el CRC.

Durante la generación del CRC, se efectúa una operación booleana OR exclusivo (XOR) a cada carácter de 8 bits con el contenido del registro. Entonces al resultado se le aplica un desplazamiento de bit en la dirección de bit menos significativo (LSB), rellenando la posición del bit más significativo (MSB) con un cero. El LSB es extraído y examinado. Si el LSB extraído fuese un 1, se realiza un XOR entre el registro y un valor fijo preestablecido[7].

Si el LSB fuese un 0, no se efectúa el XOR. Este proceso es repetido hasta haber cumplido 8 desplazamientos. Después del último desplazamiento (el octavo), el próximo byte es operado XOR con el valor actual del registro y el proceso se repite con ocho desplazamientos más, como se ha descrito mas arriba y así con todos los bytes del mensaje. El contenido final del registro, después de que todos los bytes del mensaje han sido procesados, es el valor del CRC.

Cuando el CRC es añadido al mensaje, primero se añade el byte de orden bajo seguido del byte de orden alto.

Como ejemplo se calculará el CRC de la siguiente Trama:

- Dirección del esclavo (hex) : 1
- Función (hex): 1
- Campo de dato 1 (hex): 0
- Campo de dato 2 (hex): 0
- Campo de dato 3 (hex): 0
- Campo de dato 4 (hex): 1

[7] El valor preestablecido es A001 hex, correspondiente al polinomio generador CRC16 'Inverso', que es el que se aplica al CRC Modbus.

Entonces los pasos a seguir serán:

1. Operación OR EXCLUSIVO entre el registro cargado con el valor FFFF y el campo de dirección, así:
   FFFF ***Xor*** 01 = FFFE (1111 1111 1111 1110)

2. Se extrae el bit menos significativo (LSB). Si el LSB extraído fuese un 1, se desplaza el byte y se realiza un XOR entre el registro y el valor A001. Si el LSB fuese un 0, se desplaza el byte. Este proceso es repetido hasta haber cumplido 8 desplazamientos.
   FFFE (1111 1111 1111 1110)

   Ier. Desplazamiento: 0111 1111 1111 1111
   2do. Desplazamiento y Xor: 1001 1111 1111 1110
   3ro. Desplazamiento: 0100 1111 1111 1111
   4to. Desplazamiento y Xor: 1000 0111 1111 1110
   5to. Desplazamiento: 0100 0011 1111 1111
   6to. Desplazamiento y xor: 1000 0001 1111 1110
   7mo. Desplazamiento: 0100 0000 1111 1111
   8vo. Desplazamiento: 1000 0000 0111 1110

3. El próximo byte es operado XOR con el valor actual del registro y el proceso se repite con ocho desplazamientos más, así:

   1ro: 1000 0000 0111 1110 (último valor cargado) **Xor** 01 (Campo de función)

   = 1000 0000 0111 1111

   Así hasta llegar al último desplazamiento con el valor:

   = 1110000011000001

4. El próximo byte es operado XOR con el valor actual del registro y el proceso se repite con ocho desplazamientos más, así:

1ro: 1110000011000001 (último valor cargado) **Xor** 00 (Campo de dato 01)

= 1110000011000001

Así hasta llegar al último desplazamiento con el valor:

= 1001000000100001

5. El próximo byte es operado XOR con el valor actual del registro y el proceso se repite con ocho desplazamientos más, así:

1ro: 1001000000100001 (último valor cargado) **Xor** 00 (Campo de dato 02)

= 1001000000100001

Así hasta llegar al último desplazamiento con el valor:

= 1100001010000

6. El próximo byte es operado XOR con el valor actual del registro y el proceso se repite con ocho desplazamientos más, así:

1ro: 1100001010000 (último valor cargado) **Xor** 00 (Campo de Dato 03)

= 1100001010000

Así hasta llegar al último desplazamiento con el valor:

= 11110000011000

7. El próximo byte es operado XOR con el valor actual del registro y el proceso se repite con ocho desplazamientos más, así:

1ro: 11110000011000 (último valor cargado) **Xor** 01 (Campo de Dato 04)

= 11110000011001

Así hasta llegar al último desplazamiento con el valor:

= 1100101011111101 equivalente a CAFD Hexadecimal

- Entonces el CRC será: Chequeo en Bajo: ***FD*** Chequeo en Alto: ***CA***

1.2 FUNCIONES DE CONTROL[8]

1.2.1 DIRECCIONES EN LOS MENSAJES MODBUS

Todas las direcciones en los mensajes Modbus son referenciadas a cero. La primera unidad de cada tipo de dato es direccionada como parte número cero. Por ejemplo: La bobina conocida como “bobina 1” en un controlador programable es direccionada como bobina 0000 en el campo de dirección de un mensaje Modbus. La bobina 127 decimal es direccionada como bobina 007E hex (126 decimal).

El registro mantenido 40001 es direccionado como registro 0000 en el campo de dirección de un mensaje Modbus. El campo código de función ya especifica una operación sobre un ’registro mantenido’. Por lo tanto la referencia ‘4XXXX’ está implícita. El registro mantenido 40108 es direccionado como registro 006B hex (107 decimal).

1.2.2 CAMPOS CONTENIDOS EN LOS MENSAJES MODBUS

La Figura 1.4 muestra un ejemplo de un mensaje de petición Modbus. La Figura 1.5 es un ejemplo de una respuesta normal. Ambos ejemplos

[8] Tomado de http://www.automatas.org/modbus/dcon7

muestran los campos contenidos en hexadecimal y también cómo estaría distribuido en la trama, un mensaje en modo ASCII o en modo RTU.

PETICION Nombre del Campo	Datos Ejemplo (Hex)	Caracteres ASCII	Campo 8-Bits RTU
Cabecera		:	Ninguno
Dirección del Esclavo	06	0 6	0000 0110
Función	03	0 3	0000 0011
Dirección Comienzo Alto	00	0 0	0000 0000
Dirección Comienzo Bajo	0B	0 B	0110 1011
No. de Registros Alto	00	0 0	0000 0000
No. De Registros Bajo	03	0 3	0000 0011
Comprobación de Error		LRC (2 caract)	CRC (16 bits)
Terminación		CR LF	Ninguno
		Total Bytes: 17	8

Figura 1.4 PETICIÓN DEL MAESTRO CON TRAMA ASCII/RTU

RESPUESTA Nombre del Campo	Datos Ejemplo (Hex)	Caracteres ASCII	Campo 8-Bits RTU
Cabecera		:	Ninguno
Dirección del Esclavo	06	0 6	0000 0110
Función	03	0 3	0000 0011
Computo de Bytes	06	0 6	0000 0110
Dato Alto	02	0 2	0000 0010
Dato Bajo	2B	2 B	0010 1011
Dato Alto	00	0 0	0000 0000
Dato Bajo	00	0 0	0000 0000
Dato Alto	00	0 0	0000 0000
Dato Bajo	63	6 3	0110 0011
Comprobación de Error		LRC (2caract)	CRC (16 bits)
Terminación		CR LF	Ninguno
		Total Byts: 23	11

Figura 1.5 RESPUESTA DE UN ESCLAVO CON TRAMA ASCII/RTU

La petición del maestro es una solicitud de Lectura de Registros Mantenidos, al dispositivo esclavo con dirección 06. El mensaje solicita datos numéricos de tres registros mantenidos, 40108 al 40110. Observe que el mensaje especifica la dirección de comienzo como 0107 (006B hex.).

La respuesta del esclavo replica el código de función, indicando que esto es una respuesta normal. El campo 'Cómputo de Bytes' especifica cuántas unidades de datos de 8 bits se devuelven. Muestra la cantidad de bytes de datos que vienen a continuación, bien ASCII o RTU.

En el modo ASCII, este valor representa la mitad del cómputo real de caracteres ASCII en el dato, ya que en este modo, cada dígito hexadecimal de 4 bits requiere un carácter ASCII y por lo tanto, deben haber dos caracteres ASCII para contener cada unidad de dato de 8 bits. Por ejemplo, el dato: 63 hex se envía como un byte – ocho bits - en modo RTU (0110 0011). El mismo valor, enviado en modo ASCII requiere dos caracteres ASCII, el ASCII '6' (011 0110) y el ASCII '3' (011 0011). El campo 'Cómputo de bytes' contabiliza este dato como una sola de dato de 8 bits, con independencia del método de trama de carácter (ASCII o RTU).

1.2.3 LISTADO DE CODIGOS DE FUNCIONES

La lista que se muestra a continuación detalla los códigos soportados por los controladores Modicon, en decimal.

Código	Nombre	384	484	584	884	M84	984
01	Leer Estados de Bobinas	Y[9]	Y	Y	Y	Y	Y
02	Leer Estados de Entradas	Y	Y	Y	Y	Y	Y
03	Leer Registros Mantenidos	Y	Y	Y	Y	Y	Y
04	Leer Registros de Entradas	Y	Y	Y	Y	Y	Y
05	Forzar una única Bobina	Y	Y	Y	Y	Y	Y
06	Preestablecer un único Registro	Y	Y	Y	Y	Y	Y

[9] 'Y' indica que la función está soportada.

'N' indica que no está soportada.

07		Leer Status de Excepción	Y	Y	Y	Y	Y
Y							
08		Diagnósticos					
09		Programar 484	N	Y	N	N	N
N							
10		Selección 484	N	Y	N	N	N
N							
11		Buscar Contador de Eventos de Comunic.	Y	N	Y	N	N
Y							
12		Buscar Anotac. de Eventos de Comunic.	Y	N	Y	N	N
Y							
13		Programar Controlador	Y	N	Y	N	N
Y							
14		Selección Controlador	Y	N	Y	N	N
Y							
15		Forzar Múltiples Bobinas	Y	Y	Y	Y	Y
Y							
16		Preestablecer Múltiples Registros	Y	Y	Y	Y	Y
Y							
17		Reportar Identificación de Esclavo	Y	Y	Y	Y	Y
Y							
18		Programar 884/M84	N	N	N	Y	Y
N							
19		Resetear Enlace de Comunicaciones	N	N	N	Y	Y
N							
20		Leer Referencia General	N	N	Y	N	N
Y							
21		Escribir Referencia General	N	N	Y	N	N
Y							
22		Escribir con máscara en Registros 4X	N	N	N	N	N
(1)[10]							
23		Leer/Escribir Registros 4X	N	N	N	N	N
(1)							
24		Leer Cola FIFO	N	N	N	N	N
(1)							

Tabla 1.1 LISTADO DE CÓDOGO DE FUNCIONES

1.3 MAPA DE DIRECCIONES MODBUS[11]

El protocolo Modbus, en su versión original, soporta 4 tipos de datos:

- **Salidas digitales[12] (direcciones 00001-09999):** son salidas físicas discretas. Requieren un bit que puede tomar los valores 0 o 1 y permiten acceso de escritura.

[10] (1) Función sólo soportada por 984-785

[11] Tomado de Tomado de "Control remoto del sistema B+G del Observatorio Astronómico del Montsec: monitorización de funciones" - Javier Clavero Quílez

[12] En Modbus se utiliza el indiferentemente el término bobina o salida digital

- **Entradas digitales (direcciones 10001-19999):** son entradas físicas discretas. Requieren un bit que puede tomar los valores 0 o 1 y permiten acceso de escritura/lectura.

- **Entradas analógicas (direcciones tipo 30001-39999):** son entradas físicas analógicas que funcionan con registros de 16 bits y que permiten acceso de escritura.

- **Salidas analógicas (direcciones 40001-49999):** se trata de salidas físicas analógicas o registros internos del equipo. También conocidas con el nombre de holding registers. Registros de 16 bits que permiten acceso de lectura y de escritura.

En la Tabla 1.2 se pueden ver estos mismos datos pero de forma más clara:

Tipo de Datos	Tipo de Acceso	Rango de Memoria
Salidas Digitales	Escritura	00001-09999
Entradas Digitales	Escritura / Lectura	10001-19999
Entradas Analógicas	Escritura	30001-39999
Registros de memoria	Escritura / Lectura	40001-49999

Tabla 1.2 ESTRUCTURA DE LOS DATOS EN MODBUS.

1.4 INTERFACES[13]

1.4.1 INTERFACE RS-232.

Los niveles de voltaje que se usa en el estándar RS-232 se muestran en la Tabla 1.3 siguiente:

[13] Parte de esta sección fue tomado de http://es.wikipedia.org/wiki/Modbus

Voltaje	Lógico	Control	Terminología
+3[v] a +25[v]	0	Activo	Espacio
-3[v] a - 25[v]	1	Inactivo	Marca

Tabla 1.3 NIVELES DE VOLTAJE PARA RS-232

Se observa que un 1 lógico equivale a un voltaje negativo (-3v a -25 v), y un 0 lógico equivale a un voltaje positivo (+3v a +25v). Ha esta característica se la llama "*lógica invertida*". Un voltaje que está entre +3[v] y -3[v] es tomado como indeterminado.

La velocidad a la que se envían datos en forma serial a través de una línea de comunicación, se denomina velocidad de transmisión en baudios. La velocidad de baudios es expresada en unidades de bits (para este caso) por segundo. Una conexión RS-232 conectado a 1200 baudios tiene la capacidad de enviar 1200 bits de datos en 1 segundo.

Si se pueden enviar 1200 bits en un segundo, como máximo, el inverso de 1200 da un resultado el tiempo de bit (periodo de un bit).

$$\frac{1}{(Velocidad\ de\ bauduios)} = tiempo\ de\ bit = \frac{1}{1200} = 833\,\mu s$$

Si un receptor y transmisor se conectan a 1200 baudios, el transmisor enviara bits de datos cada 833us, y el receptor tomara lectura de los bits de datos cada 833us.

Si la línea de transmisión es inactiva entonces el estado de marca (1 lógico). La transmisión de cada carácter en una línea de comunicación asíncrona va precedida de un bit de inicio. El bit de inicio es un espacio (0 lógico) con duración igual al tiempo de bit. En el receptor, cuando la línea cambia de marca a espacio se indica la presencia de un bit de inicio y después se envían los bits de datos con un tiempo de bit igual a 833us, si la transmisión es a 1200 baudios.

Después de que el último bit de datos ha sido enviado, el transmisor pasa al nivel de marca durante un tiempo de bit. Este bit es llamado bit de paro. El bit de paro indica que todos los bits de datos han sido enviados y la transmisión del carácter se ha completado. Si el receptor detecta un bit de inicio y el apropiado número de bits, pero no detecta el nivel de marca al final, esto indica un error el la transmisión.

Parameter	Conditions	Min	Max	Units
Driver Output Voltage Open Circuit			25	V
Driver Output Voltage Loaded	$3k\Omega < R_L < 7\ k\Omega$	±5	±15	V
Driver Output Resistance Power Off	$-2V<V_o<2V$		300	Ω
Slew Rate		4	30	V/µS
Maximum Load Capacitance			2500	pF
Receiver Input Resistance		3	7	kΩ
Receiver Input Threshold:				V
Output = Mark		-3		
Output = Space			3	

Tabla 1.4 ESPECIFICACIONES ELÉCTRICAS PARA RS-232

1.4.1.1 LIMITACIONES DE LA RS-232

La RS-232 C tiene una limitación de distancia máxima de 15 metros. Si bien no es una desventaja considerable cuando los equipos a conectar se encuentran cerca, sí es un inconveniente cuando la RS-232 se utiliza para conectar directamente terminales que puedan estar lejanas.

Cuando una señal cambia de una condición a otra, la especificación limita el tiempo que puede permanecer en la región de transición. Este requerimiento determina el máximo de capacidad distribuida admisible en el cable, porque la capacidad limita el tiempo de transición de la señal. La norma RS-232 especifica que la capacidad en la línea no debe superar los 2.500 picofaradios. Los cables que se suelen utilizar tienen una capacidad de 120 a 150 picofaradios por metro de longitud, por lo que la RS-232 tiene como límite de 15 m de distancia, como se vio anteriormente.

Una segunda limitación de la RS-232 es su método de toma de tierra o retorno común. Este método, llamado transmisión no balanceada, funciona bien la mayor parte del tiempo. Sin embargo, si hay diferencia de potencial entre los dos extremos del cable (lo cual es bastante probable en recorridos largos), se reduce la región de transición entre marca y espacio. Cuando ocurre esto, existe la posibilidad que no se interpreten bien los distintos estados de la señal.

Otra dificultad es su máximo de 20 KB/s para la velocidad de transmisión. Si bien en el momento de aparición del estándar era suficiente, en la actualidad, comparando con las velocidades alcanzadas por las redes de área local, 10 y 100 MB/s y las exigencias de ancho de banda que las aplicaciones requieren, la RS-232 C en algunos casos está disminuyendo su aplicación.

A partir de la RS-232 se desarrollaron nuevas interfaces que pretenden transmitir a mayor velocidad alcanzando mayor distancia. Estas nuevas interfaces como la RS-422 y la RS-423 eliminan algunas de las restricciones de la RS-232, por ejemplo, la de poseer un retorno común para todas las señales.

1.4.2 INTERFACE RS-485.

Está definido como un sistema en bus de transmisión multipunto diferencial, es ideal para transmitir a altas velocidades sobre largas distancias (35 Mbps hasta 10 metros y 100 Kbps en 1.200 metros) y a través de canales ruidosos, ya que reduce los ruidos que aparecen en los voltajes producidos en la línea de transmisión. El medio físico de transmisión es un par entrelazado que admite hasta 32 estaciones en 1 solo hilo, con una longitud máxima de 1.200 metros operando entre 300 y 19200 bps y la comunicación half-duplex (semiduplex). Soporta 32 transmisores y 32 receptores. La transmisión diferencial permite múltiples drivers dando la posibilidad de una configuración multipunto.

Parameter	Conditions	Min	Max	Units
Driver Output Voltage Open Circuit		1.5 -1.5	6 -6	V V
Driver Output Voltage Loaded	R_L= 100Ω	1.5 -1.5	5 -5	V V
Driver Output Short Circuit Current	Per output to common		±250	mA
Driver Output Rise Time	R_L = 54Ω C_L = 50 pF		30	% of bit width
Driver Common-Mode Voltage	R_L = 54Ω		±3	V
Receiver Sensitivity	-7V ≤V_{CM}≤ 12 V		±200	mV
Receiver Common-Mode Voltage Range		-7	12	V
Receiver Input Resistance		12		kΩ

Tabla 1.6 ESPECIFICACIONES ELÉCTRICAS PARA RS-485

En el siguiente capitulo se encuentra el desarrollo de Hardware del sistema.

CAPÍTULO 2

DESARROLLO DEL SOFTWARE DEL SISTEMA

CAPÍTULO 2

DESARROLLO DEL SOFTWARE DEL SISTEMA

Este capítulo corresponde al diseño y desarrollo del software requerido para el presente proyecto.

Se han desarrollado programas, utilizando lenguaje Assembler, para cada uno de los diferentes microcontroladores utilizados, y adicionalmente, se realizó una HMI (Interfaz Hombre Máquina) para el comando de los Dispositivos Esclavos.

En resumen se han realizado cinco programas, los cuales se enlistan a continuación:

- Software de recepción de la trama MODBUS al Dispositivo Master desde el computador.
- Software de transmisión desde el Dispositivo Master hacia los Dispositivos Esclavos.
- Software de configuración de los Dispositivos Esclavos para que operen en modo ASCII y RTU.
- Software de recepción de la trama MODBUS al Dispositivo Master desde los Dispositivos Esclavos.
- Software de transmisión de la trama MODBUS del Dispositivo Master hacia el computador.

2.1 DESARROLLO DEL SOFTWARE DEL DISPOSITIVO MASTER

El software de este Dispositivo permitirá la recepción de la trama MODBUS desde el computador, como también la transmisión dirigida a los Dispositivos Esclavos. Además, receptará la trama MODBUS desde los Dispositivos Esclavos y enviará la trama MODBUS de regreso hacia el computador para que la HMI desarrollada procese la información.

2.1.1 DESARROLLO DEL SOFTWARE DE RECEPCIÓN DE LA TRAMA MODBUS EN EL DISPOSITIVO MASTER DESDE EL COMPUTADOR Y TRANSMISION HACIA LOS DISPOSITIVOS ESCLAVOS

El objetivo de este programa es la recepción en el Dispositivo Master de la trama MODBUS desde el computador en los dos modos : ASCII o RTU. Según los parámetros que configure el usuario en la HMI se construirá la trama en cualquiera de los dos modos ASCII o RTU, con su respectivo chequeo de errores, para enviarla al Dispositivo Master.

La recepción en el Dispositivo Master comienza cuando se recibe una interrupción. El esclavo recibe la trama, ya sea en modo (ASCII o RTU), en recepción continua por lo que se arranca un timer para controlar el tiempo entre caracteres. Cuando no exista un carácter en el buffer de recepción y el timer haya vencido, continuara con el siguiente paso del programa.

Para capturar la trama MODBUS se procede a distinguirla por el primer carácter
en ASCII el primer carácter es ":", 3A hexadecimal, finalizando la trama con los dos últimos caracteres correspondientes (CRLF)"Retorno de carro + Avance de Línea" ,0D y 0A hexadecimal. Si se captura una trama MODBUS valida se la dirigirá a una subrutina de envió, donde cada carácter será enviado como el protocolo MODBUS define: en un tiempo menor a 1 segundo, pero, de conformidad al proyecto, se enviará en un tiempo de 200 ms entre carácter y carácter. Si el primer carácter es diferente a ":", 3A hexadecimal, se le dirigirá a una subrutina correspondiente al envió en modo RTU, la cual enviará la trama MODBUS según establece el protocolo MODBUS en modo RTU que se describe a continuación.

Para definir el tiempo de carácter en modo RTU se emplea un cristal de 4 MHZ, utilizando en Modo de alta velocidad (BRGH=1). Se carga en el registro SPBRG del microcontrolador el valor de 103 con lo cual se tiene una velocidad

de transmisión de 2400 bps. El cálculo de tiempos de carácter se realiza de la siguiente manera:

Velocidad de transmisión:	2400 bps
Bit de Inicio:	1
Bits de Datos:	8
Bit de Paridad:	0
Bits de Parada:	2
Total bits:	11

Un carácter está compuesto por 11 bits; Por tanto :

$$TIEMPOS\ DE\ CARACTER = \frac{11}{2400} = 4.58ms$$

Para el tiempo de transmisión entre carácter y carácter, el protocolo MODBUS en modo RTU establece que debe ser menor o igual a 1.5 tiempos de carácter que corresponde a 6.87ms. Para el proyecto se utiliza un tiempo de 6.63ms. Mientras se envía la trama en cualquiera de los dos métodos, se inhabilita en el Dispositivo Master el funcionamiento de los microcontroladores asociados a la recepción de tramas MODBUS desde los Dispositivos Esclavos, colocando los MAX-485 asociados en un estado de alta impedancia.

Se realizó un control de fallas en todos los microcontroladores, tanto en el Dispositivo Master como en los Dispositivos Esclavos mediante un detector de pulso faltante, cuyo funcionamiento es el de reiniciar al microcontrolador en caso de que se cuelgue. El control se realiza enviando por un pin del microcontrolador un tren de pulsos, si se envía este tren de pulsos el microcontrolador funcionara sin ningún problema; caso contrario no se envía este tren de pulsos inmediatamente se reiniciara el microcontrolador.

En la Figura 2.1 se muestra el flujograma correspondiente al programa de recepción de la trama MODBUS en el Dispositivo Master desde el computador y la transmisión de la trama MODBUS hacia los Dispositivos Esclavos.

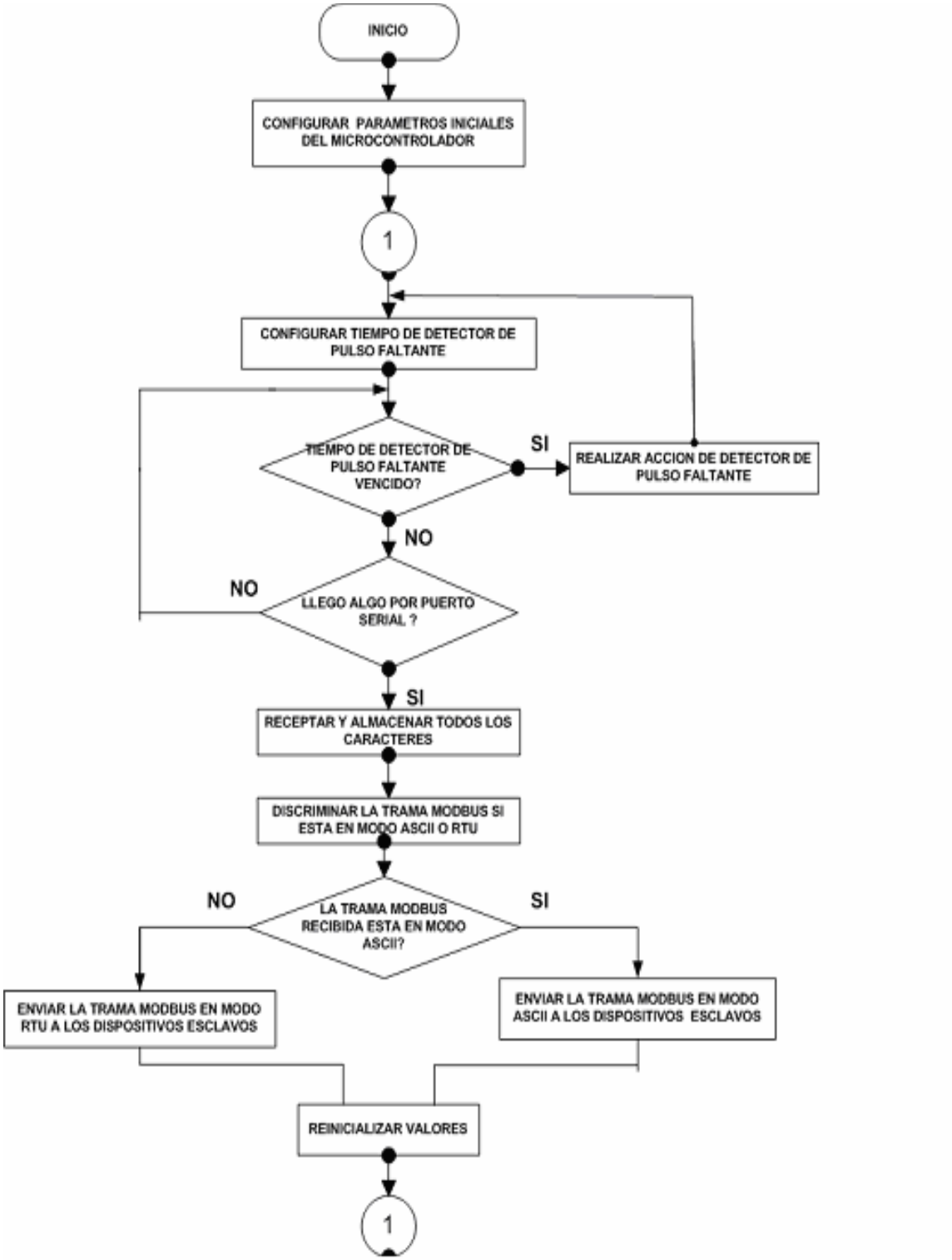

Figura 2.1 FLUJOGRAMA DE RECEPCIÓN DE LA TRAMA MODBUS EN EL DISPOSITIVO MASTER DESDE EL COMPUTADOR Y TRANSMISIÓN HACIA LOS DISPOSITIVOS ESCLAVOS

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Configurar parámetros iniciales del microcontrolador

Declarar Variables (Etiquetas).
Configurar Puertos.
Configurar Timer.
Configurar Puerto Serial : 2400 bps, 8 bits.
Vectorizar Interrupciones.
Cargar Valores en Registros.

Configurar tiempo del Detector de pulso faltante

Fijar tiempo para enviar uno lógico al detector de pulso faltante.

Realizar acción de Detector de pulso faltante

Enviar cero lógico al detector de pulso faltante.

Receptar y almacenar todos los caracteres

Receptar todos los caracteres que ingresen al puerto serial.
Almacenar caracteres.

Discriminar la trama MODBUS si esta en modo ASCII o RTU

Primer carácter es igual a ":" , 3A hexadecimal?
Si
Enviar trama en modo ASCII.
No
Enviar trama en modo RTU.

Enviar la trama MODBUS en modo ASCII a los dispositivos esclavos

Enviar los caracteres con un intervalo entre carácter y carácter cada 200ms.
Colocar a los MAX-485 en estado de alta impedancia.

Enviar la trama MODBUS en modo RTU a los dispositivos esclavos

Enviar los caracteres con un intervalo entre carácter y carácter menor o igual 6.63ms.
Colocar a los MAX-485 en estado de alta impedancia.

Reinicializar Valores

Vectorizar Interrupciones.
Cargar Valores en Registros.

2.1.2 DESARROLLO DEL SOFTWARE DE RECEPCIÓN EN EL DISPOSITIVO MASTER DE LA TRAMA MODBUS DESDE LOS DISPOSITIVOS ESCLAVOS Y LA TRANSMISION DE LA TRAMA HACIA EL COMPUTADOR

En los microcontroladores encargados de la recepción de la trama MODBUS se desarrollaron programas de acuerdo a la configuración que el usuario seleccione, mediante un conmutador de dos posiciones, entre el modo ASCII o RTU. Después de recibir la trama MODBUS desde los Dispositivos Esclavos se procederá a enviarla al HMI del computador para que esta la procese.

2.1.2.1 DESARROLLO DEL SOFTWARE EN MODO ASCII

La recepción de tramas MODBUS en modo ASCII se realiza con un intervalo de tiempo entre caracteres menor o igual a 1 segundo y con la verificación de los caracteres de inicio ":", 3A Hexadecimal, y CRLF, 0D y 0A Hexadecimales al final.

En la Figura 2.2 se muestra el flujograma del programa de recepción de la trama MODBUS en modo ASCII desde los dispositivos esclavos y transmisión de la trama MODBUS en modo ASCII hacia el computador.

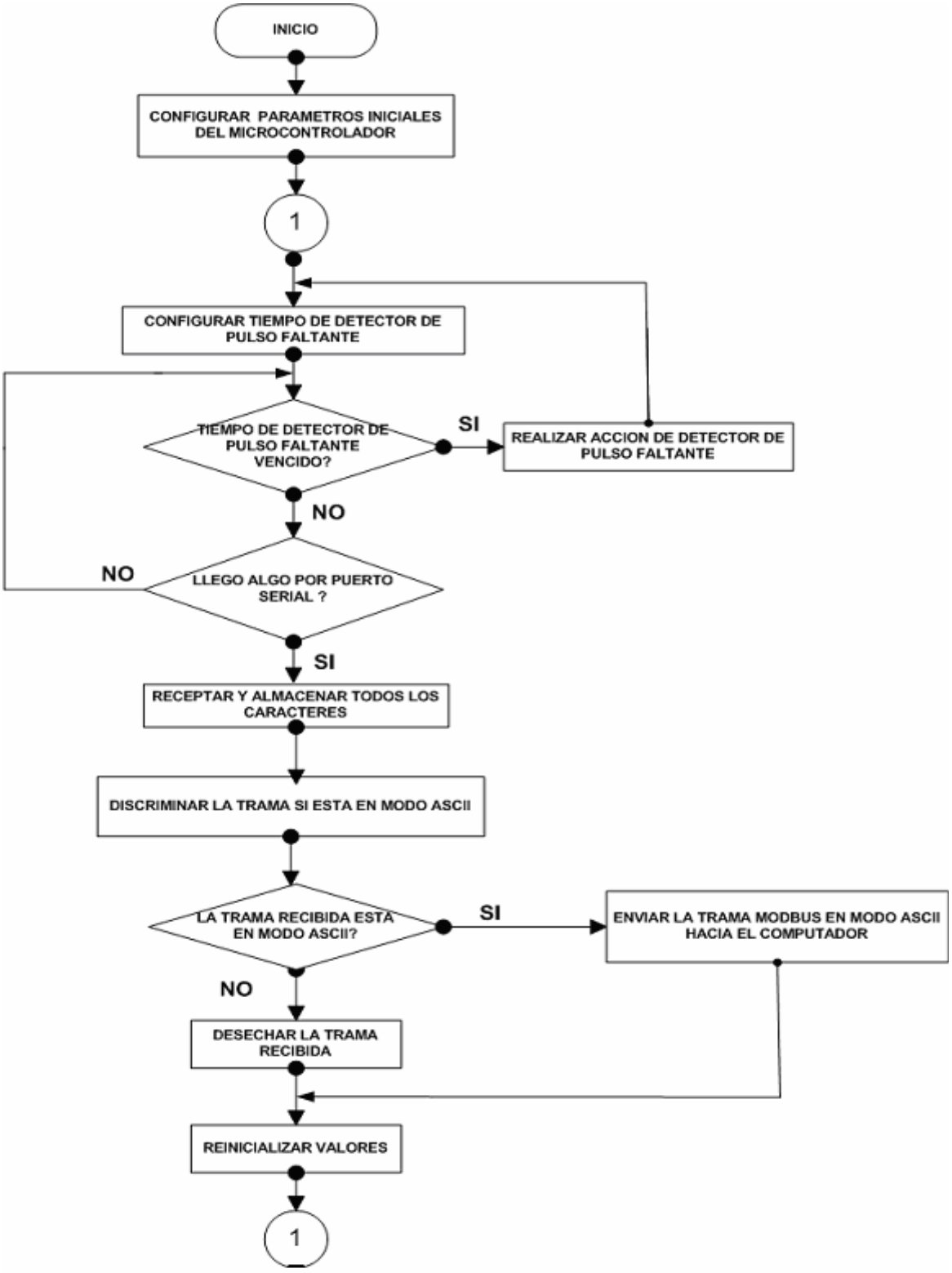

Figura 2.2 FLUJOGRAMA DE RECEPCIÓN DE LA TRAMA EN MODO ASCII DESDE LOS DISPOSITIVOS ESCLAVOS Y ENVIÓ DE TRAMA HACIA EL COMPUTADOR

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Configurar parámetros iniciales del microcontrolador

Declarar Variables (Etiquetas).
Configurar Puertos.
Configurar Timer.
Configurar Puerto Serial : 2400 bps, 8 bits.
Vectorizar Interrupciones.
Cargar Valores en Registros.

Configurar tiempo del Detector de pulso faltante

Fijar tiempo para enviar uno lógico al detector de pulso faltante.

Realizar acción de Detector de pulso faltante

Enviar cero lógico al detector de pulso faltante.

Receptar y almacenar todos los caracteres

Receptar todos los caracteres que ingresen al puerto serial .
Almacenar caracteres.

Discriminar la trama MODBUS si esta en modo ASCII

Comprobar que el primer carácter sea igual a ":", 3A Hexadecimal y comprobar que los últimos caracteres sean CRLF, 0D y 0A hexadecimales.

Enviar la trama MODBUS en modo ASCII al computador

Enviar los caracteres recibidos con un intervalo de tiempo entre carácter y carácter cada 525µs.

Desechar la Trama recibida

Si el tiempo entre carácter y carácter es mayor a 1 segundo desechar la trama.
Si los últimos caracteres no son CRLF, 0D y 0A hexadecimales desechar la trama.

Reinicializar Valores

Vectorizar Interrupciones.
Cargar Valores en Registros.

2.1.2.2 DESARROLLO DEL SOFTWARE EN MODO RTU

El software para que microcontrolador reciba la trama en modo RTU debe cumplirse con la norme que dice que cuando se recibe el primer carácter, después de 3.5 tiempos de carácter en silencio, se debe esperar un tiempo igual a 1.5 tiempos de carácter para recibir el dato siguiente. Si luego de transcurrido este tiempo ingresa un carácter, la trama MODBUS será invalidada.

Cuando el tiempo en silencio sea mayor a 3.5 tiempos de carácter (que para este proyecto es de16.575 ms), se dará por finalizada la recepción y se procederá a enviar la trama MODBUS hacia el computador; pero si llega un carácter en este intervalo de tiempo el microcontrolador recibirá este carácter como si fuera parte de la trama recibida.

Concluido el envió de la trama hacia el computador, regresará al estado de espera por otra trama.

En la Figura 2.3 se muestra el flujograma del programa de recepción de la trama MODBUS en modo RTU desde los dispositivos esclavos y la transmisión de la misma hacia el computador.

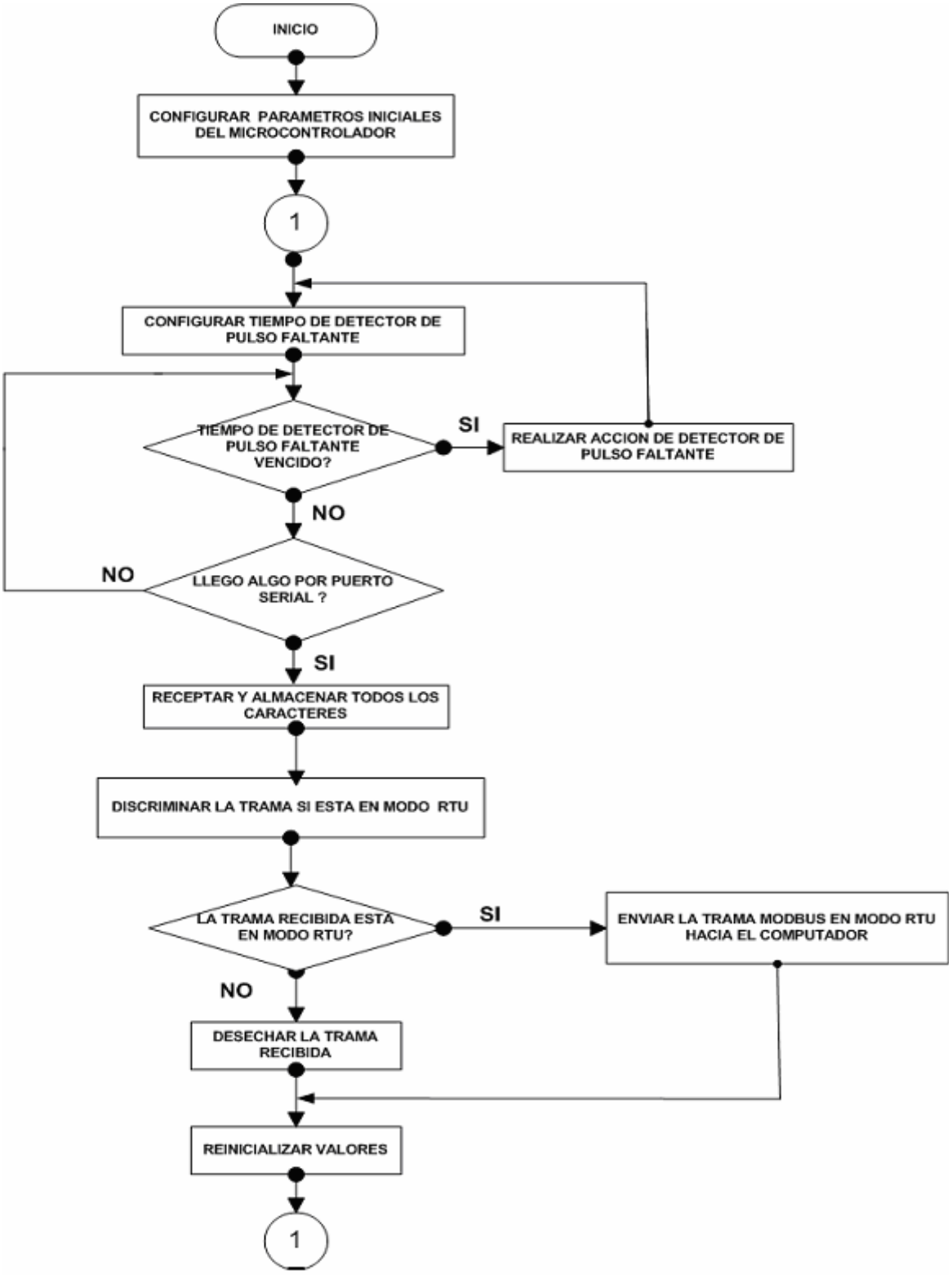

Figura 2.3 FLUJOGRAMA DE RECEPCIÓN DE LA TRAMA EN MODO RTU DESDE LOS DISPOSITIVOS ESCLAVOS Y ENVIÓ DE TRAMA HACIA EL COMPUTADOR

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Configurar parámetros iniciales del microcontrolador

Declarar Variables (Etiquetas).
Configurar Puertos.
Configurar Timer.
Configurar Puerto Serial : 2400 bps, 8 bits.
Vectorizar Interrupciones.
Cargar Valores en Registros.

Configurar tiempo del Detector de pulso faltante

Fijar tiempo para enviar uno lógico al detector de pulso faltante.

Realizar acción de Detector de pulso faltante

Enviar cero lógico al detector de pulso faltante.

Receptar y almacenar todos los caracteres

Receptar todos los caracteres que ingresen al puerto serial.
Almacenar caracteres.

Discriminar la trama MODBUS si está en modo RTU

Verifica si la trama cumple con el Protocolo MODBUS en modo RTU.

Enviar la trama MODBUS en modo RTU al computador

Enviará los caracteres recibidos con un intervalo entre carácter y carácter cada 525 µs.

Desechar la Trama recibida

Si el tiempo entre carácter y carácter es mayor al 1.5 tiempos de carácter desechar la trama.

Reinicializar Valores

Vectorizar Interrupciones.
Cargar Valores en Registros.

2.2.- DESARROLLO DEL SOFTWARE DE LOS DISPOSITIVOS ESCLAVOS

El software desarrollado para los Dispositivos Esclavos para que trabajen en los dos modos, ASCII o RTU, tienen configuraciones similares tanto en el manejo de puertos, interrupciones, recuperación y almacenaje de datos hacia la memoria EEPROM, manejo de la visualización y el comando del detector de pulso faltante acoplado al sistema; lo que difiere es la manera en que cada uno de los modos de comunicación recibe la trama MODBUS, procesa la información y envía la trama, en el formato apropiado, al Dispositivo Master.

2.2.1 DESARROLLO DEL SOFTWARE PARA OPERACIÓN EN MODO ASCII

La funciones de este programa son: la recepción desde el Dispositivo Master de la trama MODBUS en modo ASCII (trama MODBUS de petición), procesamiento de la trama MODBUS de petición y construcción y envió de la trama MODBUS de respuesta hacia el Dispositivo Master.

En primer lugar se realiza la configuración inicial de los puertos del microcontrolador, las interrupciones y la comunicación serial. El paso siguiente es leer los registro almacenados en la EEPRON y configurar al LCD; a continuación se dirigirá al programa principal, el cual está encargado del control del detector de pulso faltante, carga de registros y lo concerniente a actualizar valores de periféricos y mostrarlos en el LCD, y lo principal, es la espera de comunicación.

El algoritmo para la recepción en el Dispositivo Esclavo de la trama MODBUS comienza con la espera del carácter de inicio ":", 3A hexadecimal, mediante interrupción serial. A continuación, se revisa los 2 caracteres siguientes que corresponden a la dirección del Dispositivo Esclavo, Si corresponden con la dirección del Dispositivo Esclavo, o si es una trama BROADCAST, continuará con la recepción de caracteres caso contrario, el Esclavo se dirigirá al programa principal a la espera de una nueva trama MODBUS. Si

correspondieron la dirección de la trama con la del Esclavo se receptará toda la trama hasta que los dos últimos caracteres sean CRLF, (0D y 0A hexadecimales) llegada la trama se procederá a cambiar los caracteres ASCII de la trama MODBUS de petición por caracteres hexadecimales. El paso siguiente es calcular el contenido del campo de comprobación de errores, que en el modo ASCII emplea el algoritmo de Comprobación Longitudinal de Errores (LRC). Luego se compara el LRC de la trama MODBUS con el LRC calculado por el Dispositivo Esclavo. Si los resultados no son iguales, la trama MODBUS sufrió daños, será desechada y el algoritmo seguirá al programa principal en espera de una nueva Trama MODBUS. Si los resultados de la comparación del LRC son iguales, la trama MODBUS de petición no sufrió daños y el algoritmo se dirigirá a la subrutina de funciones, la cual elaborará una trama MODBUS de respuesta.

Enviada la trama MODBUS de respuesta al Dispositivo Master con intervalo de tiempo de 200 ms entre caracteres se procederá a reiniciar el estado de los registros e interrupciones y regresar al programa de espera.

En la Figura 2.4 se muestra el flujograma que correspondiente al programa del dispositivo esclavo trabajando en modo ASCII.

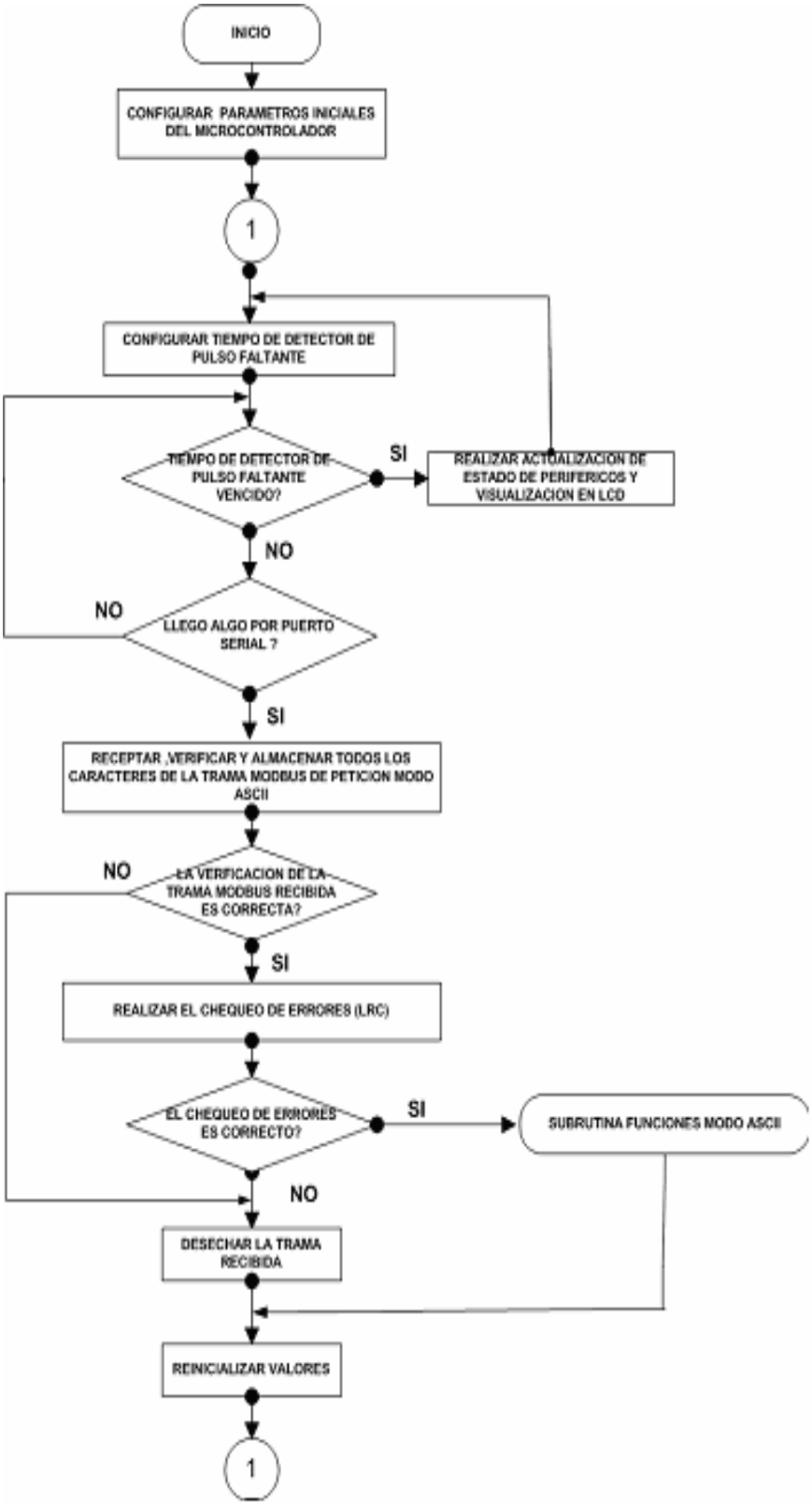

Figura 2.4 FLUJOGRAMA DEL PROGRAMA DEL DISPOSITIVO ESCLAVO EN MODO ASCII

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Configurar parámetros iniciales del microcontrolador

Declarar Variables (Etiquetas).
Configurar Puertos.
Configurar Timer.
Configurar Puerto Serial : 2400 bps, 8 bits.
Leer memoria EEPROM del microcontrolador.
Inicializar LCD.
Vectorizar Interrupciones.
Cargar Valores en Registros.

Configurar tiempo del Detector de pulso faltante

Fijar tiempo para enviar uno lógico al detector de pulso faltante.

Realizar actualización de estado de periféricos y mostrar datos en LCD

Enviar cero lógico al detector de pulso faltante.
Actualizar los valores de los periféricos de entrada y de salida.
Mostar datos de periféricos en LCD.

Receptar ,verificar y almacenar todos los caracteres de la trama modbus de peticion
modo ASCII

Receptar todos los caracteres que ingresen al puerto serial.
Verificar que la trama MODBUS esté en modo ASCII.
Almacenar caracteres.

Realizar el chequeo de errores (LRC)

Realizar la comparación del LRC de la trama MODBUS de petición con la calculada por el Dispositivo Esclavo.

Desechar la Trama recibida

Si el tiempo entre carácter y carácter sobrepasa 1 segundo desechar la trama.
Si los caracteres de finalización de trama CRLF no llegan desechar la trama.
Si la dirección del dispositivo esclavo es incorrecta desechar la trama.
Si el LRC comparado no es igual desechar la trama.

Reinicializar Valores

Vectorizar Interrupciones.
Cargar Valores en Registros.

2.2.1.1.- SUBRUTINA FUNCIONES EN MODO ASCII

En esta subrutina se verificará a qué función corresponde la trama de petición MODBUS en modo ASCII, dirigiéndola luego a las diferentes subrutinas de funciones específicas. En la Figura 2.5 se ilustra este programa.

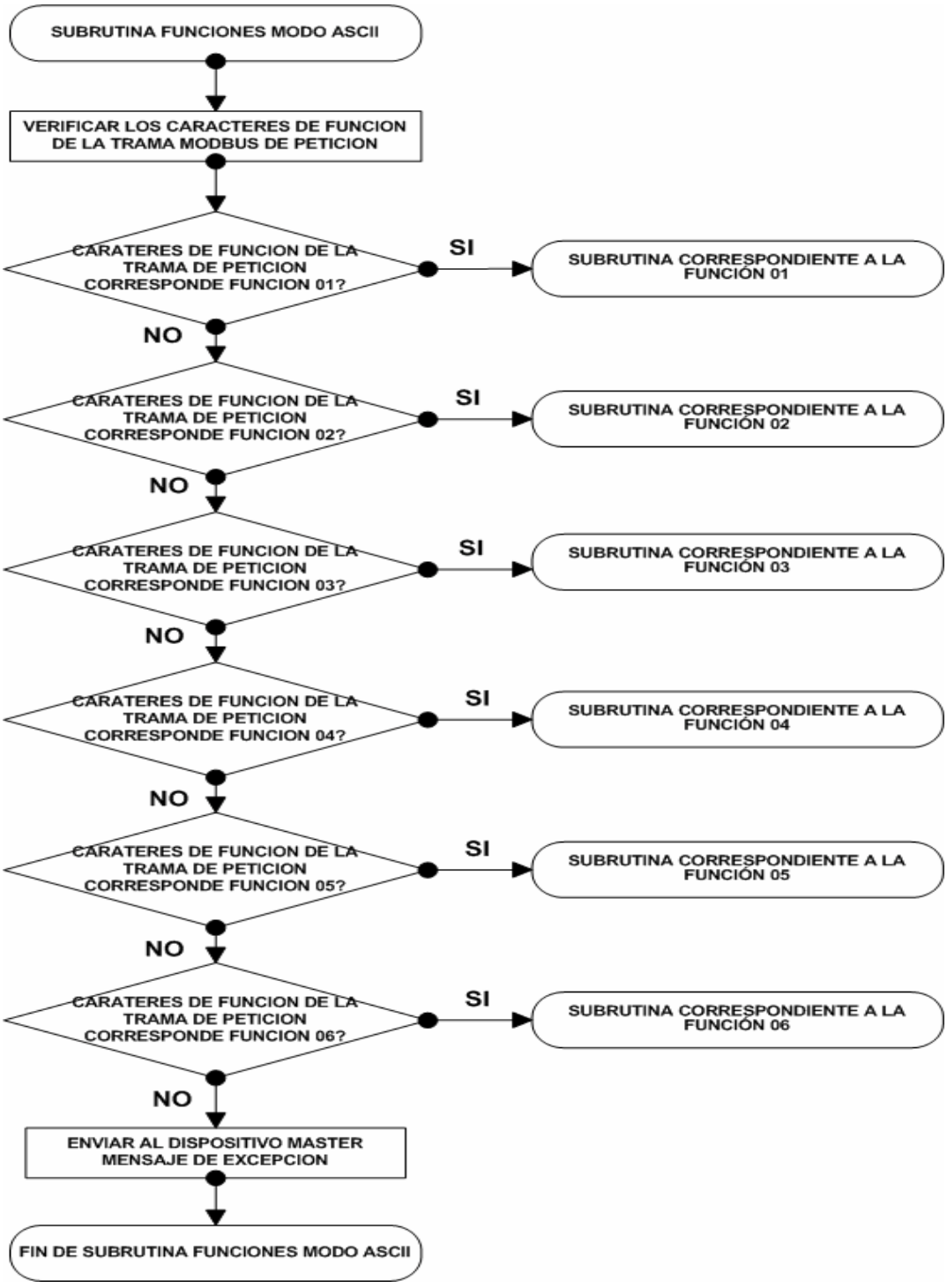

Figura 2.5 FLUJOGRAMA DEL PROGRAMA DE FUNCIONES EN MODO ASCII

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Verificar los caracteres de función de la trama MODBUS de petición

Verificar los 2 caracteres correspondientes a la función en la trama MODBUS de petición y se dirige a las subrutinas especificas de funciones.

Enviar al Dispositivo Master mensaje de excepción

Elaborar un mensaje de excepción con el código función ilegal.
Calcular el LRC del mensaje de excepción.
Cambiar los caracteres de hexadecimales por caracteres ascii.
Enviar el mensaje de excepción con un tiempo de 200 ms entre caracteres al Dispositivo Master.

2.2.1.2.- SUBRUTINA CORRESPONDIENTE A LA FUNCIÓN 01

Esta función tiene como tarea leer el estado de las salidas digitales. En el caso de los Dispositivos Esclavos solamente existe una salida digital, a la cual se le asignó la dirección 1 correspondiente a 00001. De ingresar a esta subrutina el primer paso es la revisión nuevamente de los 2 caracteres correspondientes al campo de dirección del Dispositivo Esclavo en la Trama MODBUS de petición en modo ASCII ya que esta función no soporta BROADCAST. Si es una trama BROADCAST no ejecutará acción alguna y saldrá al programa principal; Caso contrario, se construirá un trama MODBUS de respuesta con el estado de la salida digital y se enviará al Dispositivo Master; después regresará al programa principal.

En la Figura 2.6 se muestra el flujograma del programa correspondiente a la subrutina de la Función 01.

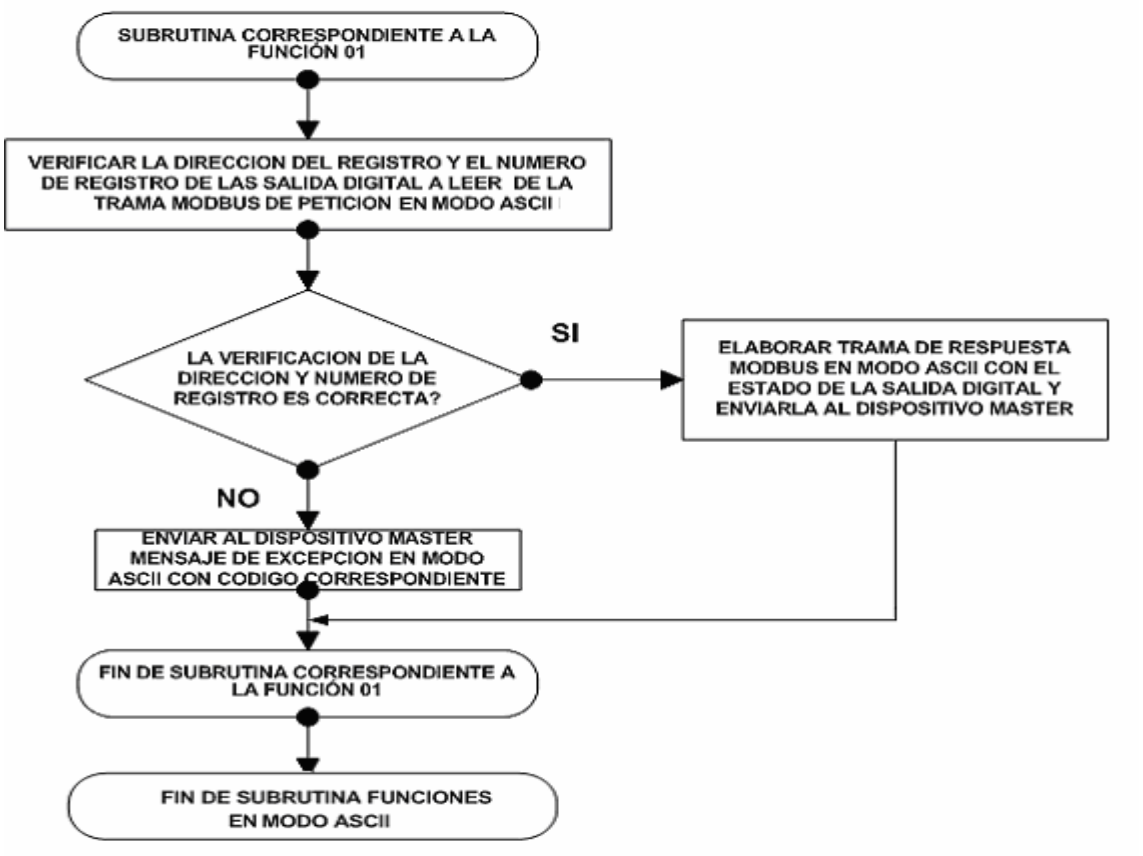

Figura 2.6 FLUJOGRAMA SUBRUTINA DE LA FUNCIÓN 01

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Elaborar trama de respuesta MODBUS en modo ASCII con el estado de la salida digital y enviarla al dispositivo master

Elaborar una Trama MODBUS con el estado actual de la salida digital.
Calcular el LRC de la trama MODBUS.
Cambiar los caracteres de hexadecimales por caracteres ascii.
Enviar la trama MODBUS de respuesta en modo ASCII con un tiempo de 200 ms entre caracteres al dispositivo master.

Enviar al Dispositivo Master mensaje de excepción en modo ASCII con código correspondiente

Elaborar un mensaje de excepción si la dirección de la salida digital a leer es incorrecta.
Elaborar un mensaje de excepción si el número de la salida digital a leer es incorrecto.
Calcular el LRC del mensaje de excepción.
Cambiar los caracteres de hexadecimales por caracteres ascii.
Enviar el mensaje de excepción con un tiempo de 200 ms entre caracteres al Dispositivo Master.

2.2.1.3.- SUBRUTINA CORRESPONDIENTE A LA FUNCIÓN 02

Esta función tiene como tarea leer el estado de la entradas digitales. En el caso de los Dispositivos Esclavos solamente existe un entrada digital, a la cual se le asignó la dirección 1 que correspondiente a 10001. De ingresar a esta subrutina, el primer paso es la revisión nuevamente de los 2 caracteres correspondientes al campo de dirección del Dispositivo Esclavo en la Trama MODBUS de petición en modo ASCII ya que esta función no soporta BROADCAST. Si es una trama BROADCAST no ejecutará acción alguna y saldrá al programa principal; Caso contrario, se construirá un trama MODBUS de respuesta con el estado de la entrada digital y se enviará al Dispositivo Master; después regresará al programa principal.

En la Figura 2.7 se muestra el flujograma del programa correspondiente a la subrutina de la Función 02.

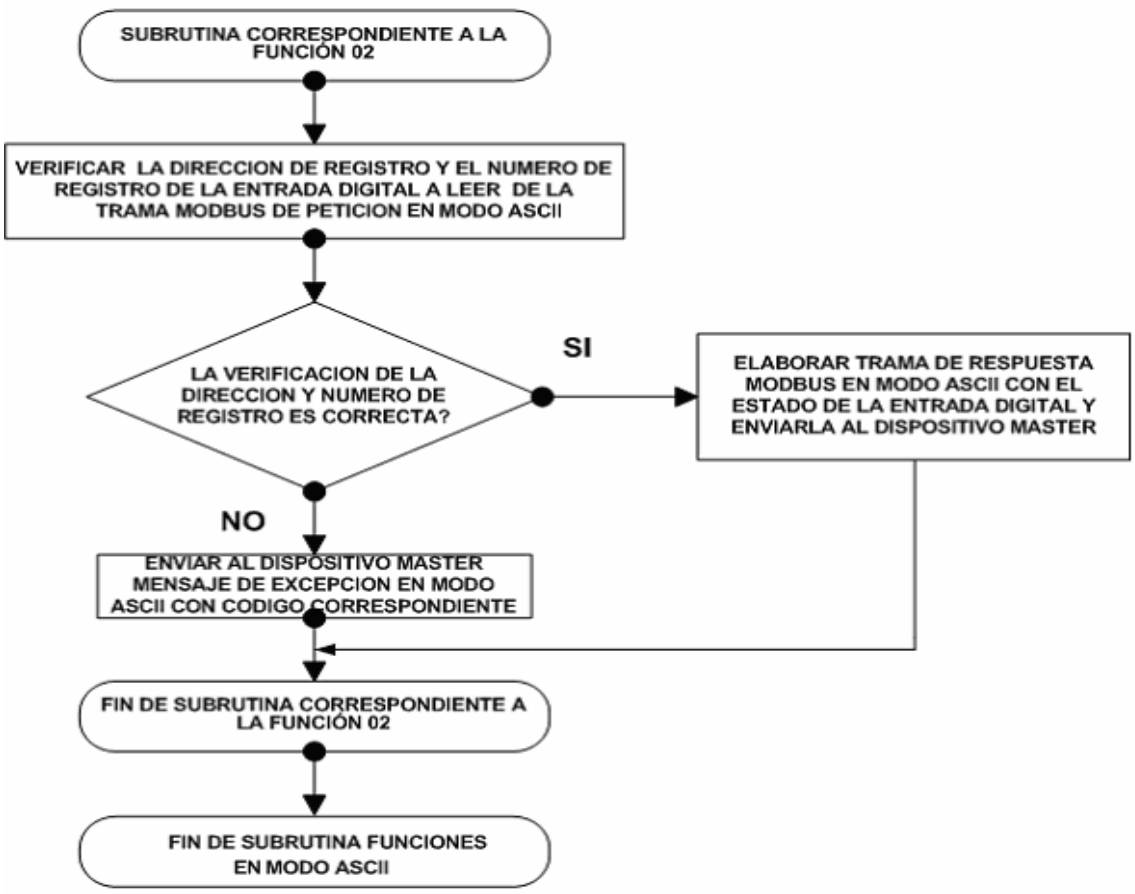

Figura 2.7 FLUJOGRAMA SUBRUTINA DE LA FUNCIÓN 02

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Elaborar trama de respuesta MODBUS en modo ASCII con el estado de la entrada digital y enviarla al dispositivo master

Elaborar una Trama MODBUS con el estado actual de la entrada digital.
Calcular el LRC de la trama MODBUS.
Cambiar los caracteres de hexadecimales por caracteres ascii.
Enviar la trama MODBUS de respuesta en modo ASCII con un tiempo de 200 ms entre caracteres al dispositivo master.

Enviar al Dispositivo Master mensaje de excepción en modo ASCII con código correspondiente

Elaborar un mensaje de excepción si la dirección de la entrada digital a leer es incorrecta.
Elaborar un mensaje de excepción si el número de la entrada digital a leer es incorrecto.
Calcular el LRC del mensaje de excepción.
Cambiar los caracteres de hexadecimales por caracteres ascii.
Enviar el mensaje de excepción con un tiempo de 200 ms entre caracteres al Dispositivo Master.

2.2.1.4.- SUBRUTINA CORRESPONDIENTE A LA FUNCIÓN 03

Esta función tiene como tarea leer registros de memoria. En el caso de los Dispositivos Esclavos solamente existe un registro de memoria, a la cual se le asigno la dirección 1 que corresponde a 40001. De ingresar a esta subrutina, el primer paso es la revisión nuevamente de los 2 caracteres correspondientes al campo de dirección del Dispositivo Esclavo en la Trama MODBUS de petición en modo ASCII ya que esta función no soporta BROADCAST. Si es una trama BROADCAST no ejecutará acción alguna y saldrá al programa principal; Caso contrario, se construirá un trama MODBUS de respuesta con el valor del registro de memoria y se enviará al Dispositivo Master; después regresará al programa principal.

En la Figura 2.8 se muestra el flujograma del programa correspondiente a la subrutina de la Función 03.

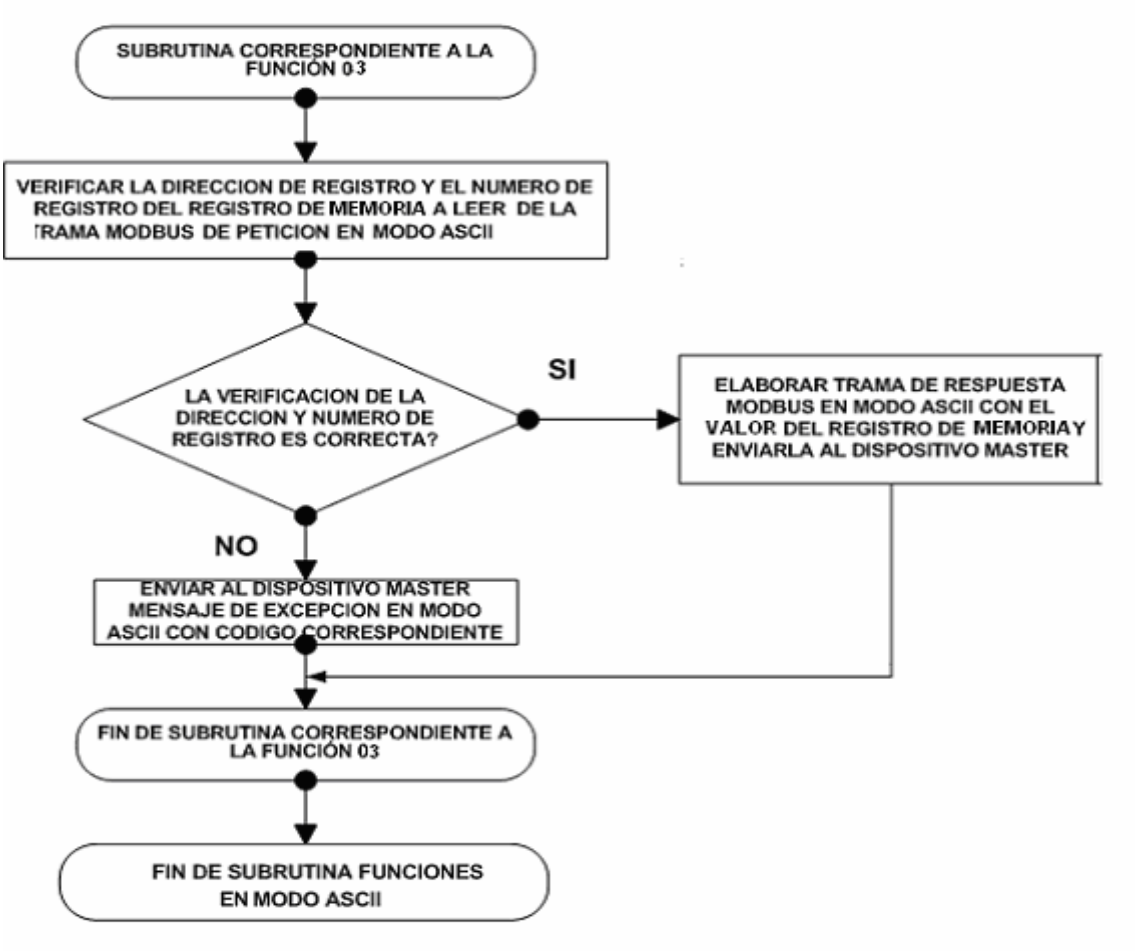

Figura 2.8 FLUJOGRAMA SUBRUTINA DE LA FUNCIÓN 03

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Elaborar trama de respuesta MODBUS en modo ASCII con el valor del registro de memoria y enviarla al dispositivo master

Elaborar una Trama MODBUS con el valor del registro de memoria.
Calcular el LRC de la trama MODBUS
Cambiar los caracteres hexadecimales por caracteres ascii.
Enviar la trama MODBUS de respuesta en modo ASCII con un tiempo de 200 ms entre caracteres al dispositivo master.

Enviar al Dispositivo Master mensaje de excepción en modo ASCII con código correspondiente

Elaborar un mensaje de excepción si la dirección del registro de memoria a leer es incorrecta.
Elaborar un mensaje de excepción si el número del registro de memoria a leer es incorrecto.
Calcular el LRC del mensaje de excepción.
Cambiar los caracteres de hexadecimales por caracteres ascii.
Enviar el mensaje de excepción con un tiempo de 200 ms entre caracteres al Dispositivo Master.

2.2.1.5.- SUBRUTINA CORRESPONDIENTE A LA FUNCIÓN 04

Esta función tiene como tarea leer registros de entrada. En el caso de los Dispositivos Esclavos solamente existe un registro de entrada, a la cual se le asignó la dirección 1 que corresponde a 30001. De ingresar a esta subrutina, el primer paso es la revisión nuevamente de los 2 caracteres correspondientes al campo de dirección de Dispositivo Esclavo en la Trama MODBUS de petición en modo ASCII ya que esta función no soporta BROADCAST. Si es una trama BROADCAST no ejecutará acción alguna y saldrá al programa principal; Caso contrario, se construirá un trama MODBUS de respuesta con el valor del registro de entrada y se enviará al Dispositivo Master; después regresará al programa principal.

En la Figura 2.9 se muestra el flujograma del programa correspondiente a la subrutina de la Función 04.

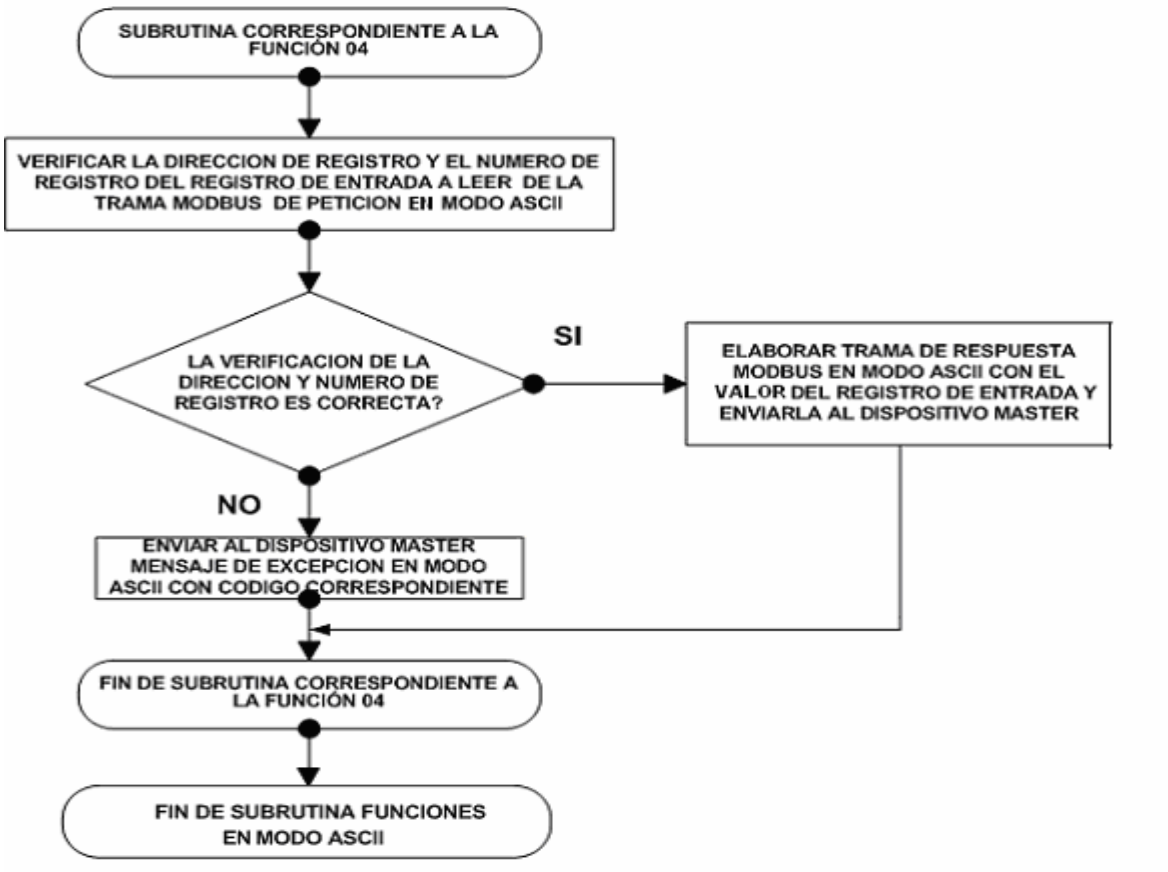

Figura 2.9 FLUJOGRAMA SUBRUTINA DE LA FUNCIÓN 04

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Elaborar trama de respuesta MODBUS en modo ASCII con el valor del registro de entrada y enviarla al dispositivo master

Elaborar una Trama MODBUS con el valor del registro de entrada.
Calcular el LRC de la trama MODBUS.
Cambiar los caracteres hexadecimales por caracteres ascii.
Enviar la trama MODBUS de respuesta en modo ASCII con un tiempo de 200 ms entre caracteres al dispositivo master.

Enviar al Dispositivo Master mensaje de excepción en modo ASCII con código correspondiente

Elaborar un mensaje de excepción si la dirección del registro de entrada a leer es incorrecta.
Elaborar un mensaje de excepción si el número del registro de entrada a leer es incorrecto.
Calcular el LRC del mensaje de excepción.
Cambiar los caracteres de hexadecimales por caracteres ascii.
Enviar el mensaje de excepción con un tiempo de 200 ms entre caracteres al Dispositivo Master.

2.2.1.6.- SUBRUTINA CORRESPONDIENTE A LA FUNCIÓN 05

Esta función tiene como tarea forzar una salida digital, En el caso de los Dispositivos Esclavos se asignó la dirección 1 que corresponde a 00001. En primer lugar esta subrutina procederá a comparar los 4 caracteres correspondientes a la dirección alta y baja de la salida digital que se quiera forzar de la trama MODBUS de petición en modo ASCII, si estos valores son validados por el Dispositivo Esclavo, revisará los 4 caracteres correspondientes al campo de datos alto y bajo de la trama MODBUS de petición, si estos caracteres son FF00 procederá a forzar a 1 lógico la salida digital y si es 0000 procederá a forzar a 0 lógico la salida digital, realizado este paso procederá a revisar los 2 caracteres correspondientes a la dirección del Esclavo si es BROADCAST saldrá al programa principal; caso contrario el Dispositivo Esclavo enviará la misma trama de petición como trama de respuesta.

En la Figura 2.10 se muestra un flujograma del programa correspondiente a la subrutina de la Función 05.

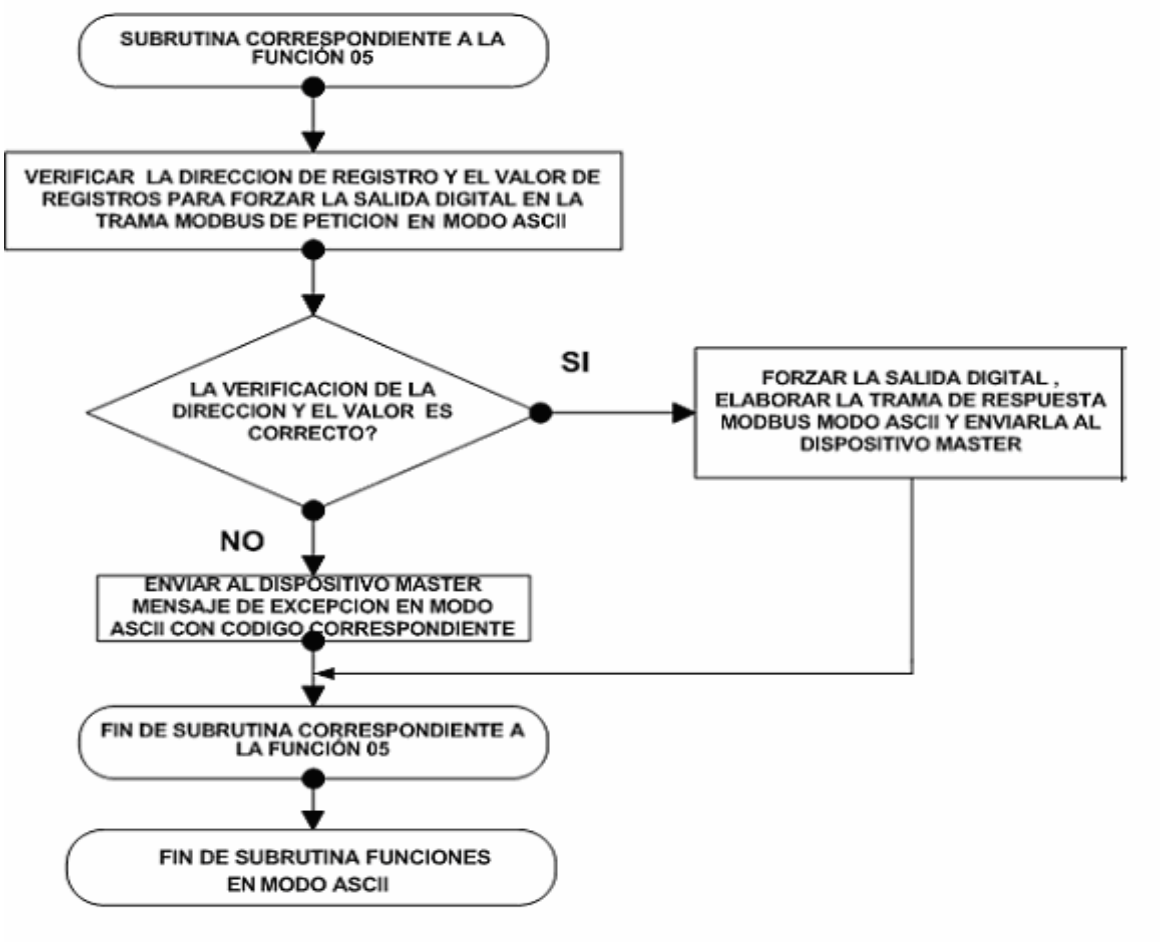

Figura 2.10 FLUJOGRAMA SUBRUTINA DE LA FUNCIÓN 05

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Forzar la salida digital ,elaborar trama de respuesta MODBUS ASCII y enviarla al dispositivo master

Forzar uno lógico o cero lógico las salida digital.
Elaborar una Trama MODBUS de respuesta igual a la trama MODBUS de petición.
Cambiar los caracteres hexadecimales por caracteres ascii.
Enviar la trama MODBUS de respuesta en modo ASCII con un tiempo de 200 ms entre caracteres al dispositivo master.

Enviar al Dispositivo Master mensaje de excepción en modo ASCII con código correspondiente

Elaborar un mensaje de excepción si la dirección de la salida digital a forzar es incorrecta.
Elaborar un mensaje de excepción si el valor de la acción que se desee realizar en la salida digital es incorrecta.
Calcular el LRC del mensaje de excepción.
Cambiar los caracteres de hexadecimales por caracteres ascii.
Enviar el mensaje de excepción con un tiempo de 200 ms entre caracteres al Dispositivo Master.

2.2.1.7.- SUBRUTINA CORRESPONDIENTE A LA FUNCIÓN 06

Esta función tiene como tarea la de programar un registro de memoria. En el caso de los Dispositivos Esclavos, se asignó la dirección 1 que corresponde a 30001. En primer lugar esta subrutina procederá a comparar los 4 caracteres correspondientes a la dirección alta y baja del registro de memoria que se quiera programar de la trama MODBUS de petición en modo ASCII, si estos valores son validados por el Dispositivo Esclavo, revisará los 4 caracteres correspondientes al campo de datos alto y bajo de la trama MODBUS de petición, si estos caracteres son permitidos se procederá a programar el Dispositivo Esclavo. A continuación se revisará los 2 caracteres correspondientes a la dirección del Dispositivo Esclavo si es BROADCAST saldrá al programa principal; caso contrario el Dispositivo Esclavo enviará la misma trama de petición como trama de respuesta al Dispositivo Master, finalizado el envió regresará al programa principal.

En la Figura 2.11 se muestra el flujograma del programa correspondiente a la subrutina de la Función 06.

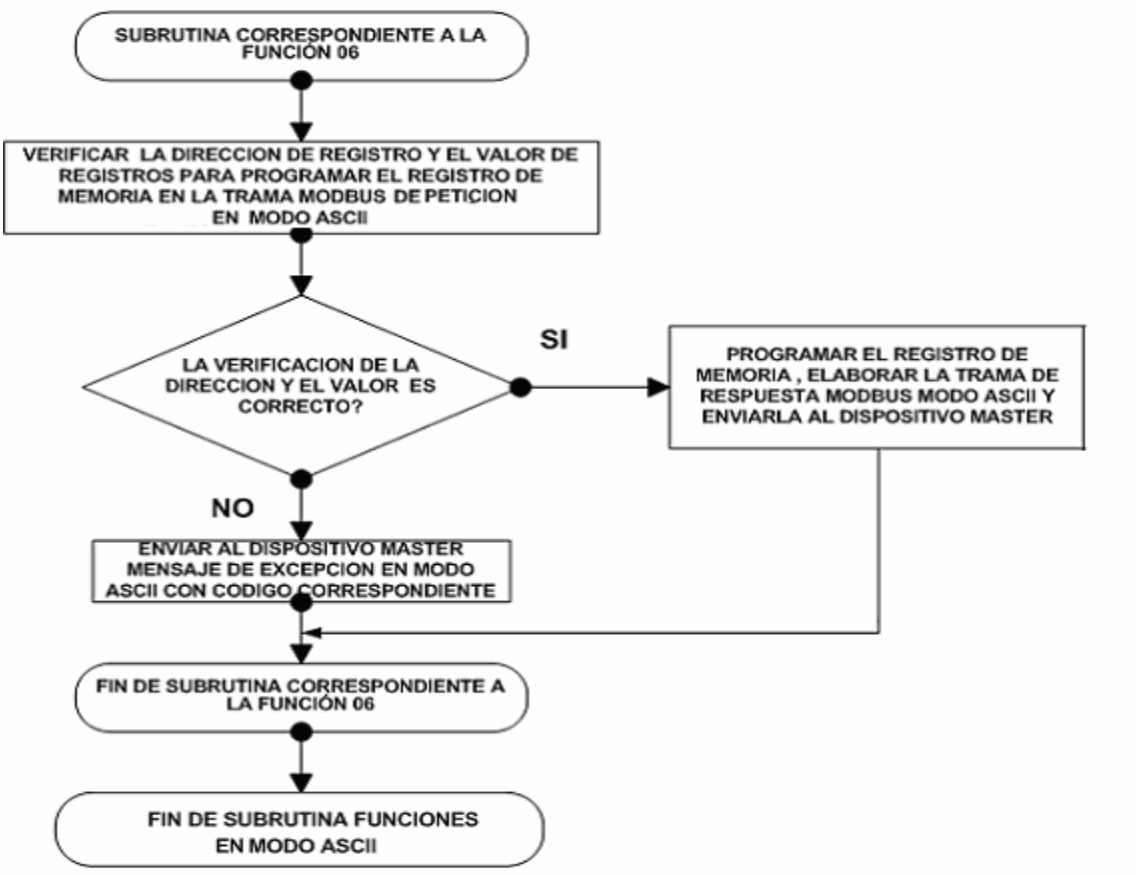

Figura 2.11 FLUJOGRAMA DE LA SUBRUTINA DE LA FUNCIÓN 06

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Programar el registro de memoria, elaborar la trama de respuesta MODBUS modo ASCII y enviarla al dispositivo master

Programar el registro de memoria.
Elaborar una Trama MODBUS de respuesta igual a la trama MODBUS de petición.
Cambiar los caracteres de hexadecimales por caracteres ascii.
Enviar la trama MODBUS de respuesta en modo ASCII con un tiempo de 200 ms entre caracteres al dispositivo master .

Enviar al Dispositivo Master mensaje de excepción en modo ASCII con código correspondiente

Elaborar un mensaje de excepción si la dirección del registro de memoria a programar es incorrecta.
Elaborar un mensaje de excepción si el valor que se desee programar en el registro de memoria es incorrecta.
Calcular el LRC del mensaje de excepción.
Cambiar los caracteres de hexadecimales por caracteres ascii.
Enviar el mensaje de excepción con un tiempo de 200 ms entre caracteres al Dispositivo Master.

2.2.2 DESRROLLO DEL SOFTWARE PARA OPERACIÓN EN MODO RTU

La funciones de este programa son: la recepción desde el Dispositivo Master de la trama MODBUS en modo RTU (trama MODBUS de petición), procesamiento de la trama MODBUS de petición, y construcción y envió de la trama MODBUS de respuesta hacia el Dispositivo Master.

En primer lugar, se realiza la configuración inicial de los puertos del microcontrolador, las interrupciones y la comunicación serial. El paso siguiente es leer los registros almacenados en la EEPRON y configurar al LCD. A continuación se dirigirá al programa principal, el cual está encargado del control del detector de pulso faltante, carga de registros y lo concerniente a actualizar valores de periféricos y mostrarlos en el LCD, y lo principal, la espera de comunicación.

El algoritmo para la recepción en el Dispositivo Esclavo de la trama MODBUS, comienza con la espera de 3.5 tiempos de carácter en silencio, transcurrido este tiempo el algoritmo espera la llegada del primer carácter. Al momento que llegue el primer carácter, mediante interrupción serial, se revisará este carácter que corresponde a la Dirección del Esclavo. Si corresponde a la dirección de Dispositivo Esclavo o si es una trama BROADCAST, continuará con la recepción de caracteres; caso contrario, el Esclavo se dirigirá al programa principal.

Si correspondieron la dirección de la trama con la del Esclavo se esperará un tiempo de 1.5 tiempos de carácter para la recepción continua de caracteres, en caso de que un carácter ingrese a un tiempo mayor que dicho tiempo y la trama no se haya completado, la trama será invalidada, si esto no ocurre y existe un tiempo en silencio de 3.5 tiempos de carácter la trama MODBUS será validada y saldrá de la interrupción. El siguiente paso es de calcular el contenido del campo de comprobación de errores, que en el modo RTU emplea el algoritmo de Comprobación de Redundancia Cíclica (CRC). Luego se compara el CRC de la trama MODBUS con el CRC calculado por el Dispositivo

Esclavo. Si los resultados no son iguales, la trama MODBUS sufrió daños, será desechada y el algoritmo seguirá al programa principal en espera de una nueva trama MODBUS. Si los resultados de la comparación del CRC son iguales, la trama MODBUS de petición no sufrió daños y el algoritmo se dirigirá a la subrutina de funciones, la cual elaborar una trama MODBUS de respuesta.

Enviada la trama MODBUS de respuesta al Dispositivo Master con un intervalo de tiempo de 1.5 tiempos de carácter entre caracteres, se dirigirá al programa principal para la espera de una nueva trama MODBUS.

En la Figura 2.12 se muestra el flujograma correspondiente al programa del dispositivo esclavo trabajando en modo RTU.

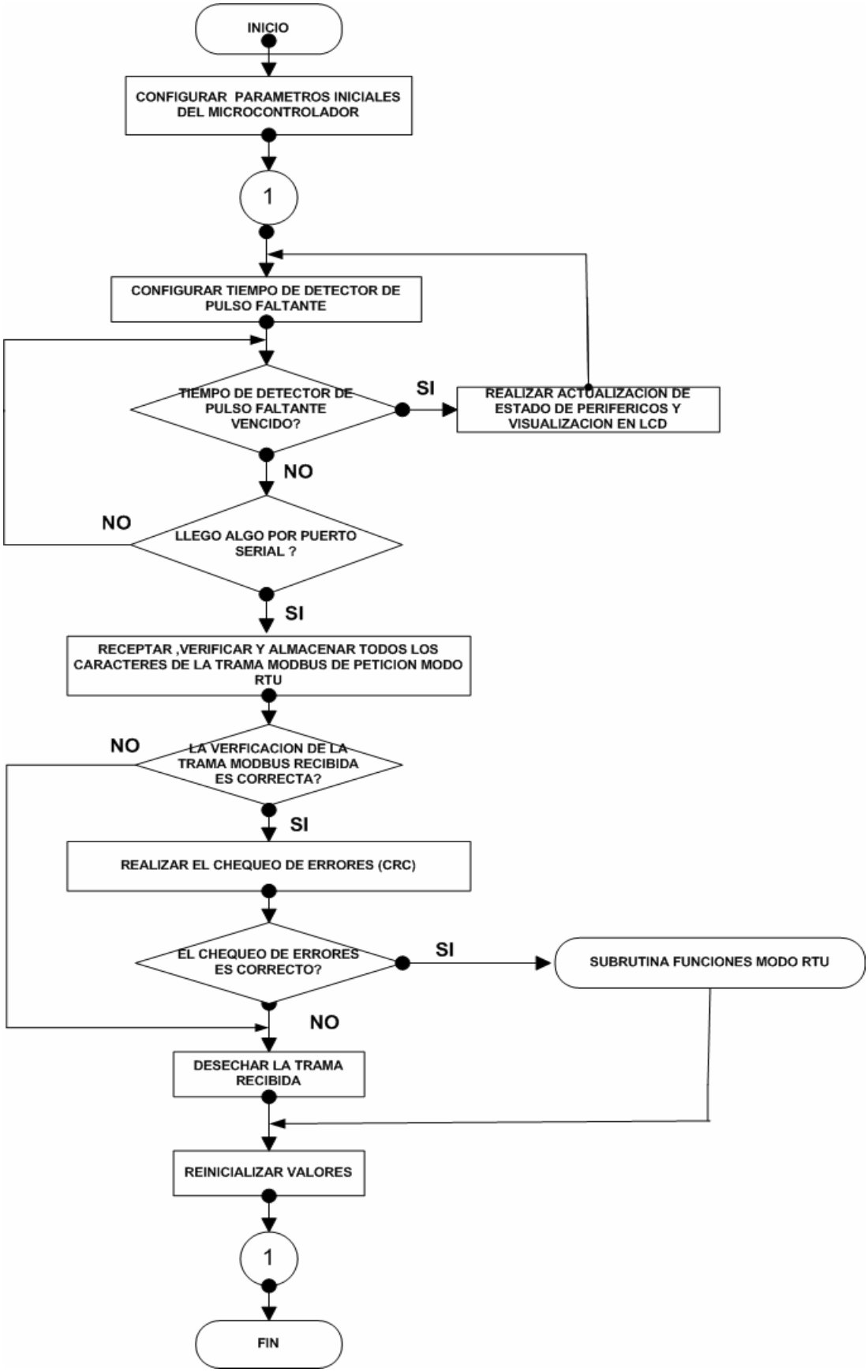

Figura 2.12 FLUJOGRAMA DEL PROGRAMA DEL DISPOSITIVO ESCLAVO EN MODO RTU

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Configurar parámetros iniciales del microcontrolador

Declarar Variables (Etiquetas).
Configurar Puertos.
Configurar Timer.
Configurar Puerto Serial : 2400 bps, 8 bits.
Leer memoria EEPROM del microcontrolador.
Inicializar LCD.
Vectorizar Interrupciones.
Cargar Valores en Registros.

Configurar tiempo del Detector de pulso faltante

Fijar tiempo para enviar uno lógico al detector de pulso faltante.

Realizar actualización de estado de periféricos y mostrar datos en LCD

Enviar cero lógico al detector de pulso faltante.
Actualizar los valores de los periféricos de entrada y de salida.
Mostar datos de periféricos en LCD.

Receptar ,verificar y almacenar todos los caracteres de la trama MODBUS de petición en modo RTU

Receptar todos los caracteres que ingresen al puerto serial.
Verificar que la trama MODBUS esté en modo RTU.
Almacenar caracteres.

Realizar el chequeo de errores (CRC)

Realizar la comparación del CRC de la trama MODBUS de petición con el valor calculado por el Dispositivo Esclavo.

Desechar la Trama recibida

Si el tiempo entre carácter y carácter sobrepasa 1.5 tiempos de carácter y no se haya completado la trama, desechar la trama .
Si la dirección del dispositivo esclavo es incorrecta desechar la trama.
Si la comparación del CRC es incorrecta desechar la trama.

Reinicializar Valores

Vectorizar Interrupciones.
Cargar Valores en Registros.

2.2.2.1.- SUBRUTINA FUNCIONES EN MODO RTU

En esta subrutina se verificará a qué función corresponde la trama de petición MODBUS en modo RTU, dirigiéndola luego a las diferentes subrutinas de funciones específicas. En la Figura 2.13 se ilustra este programa.

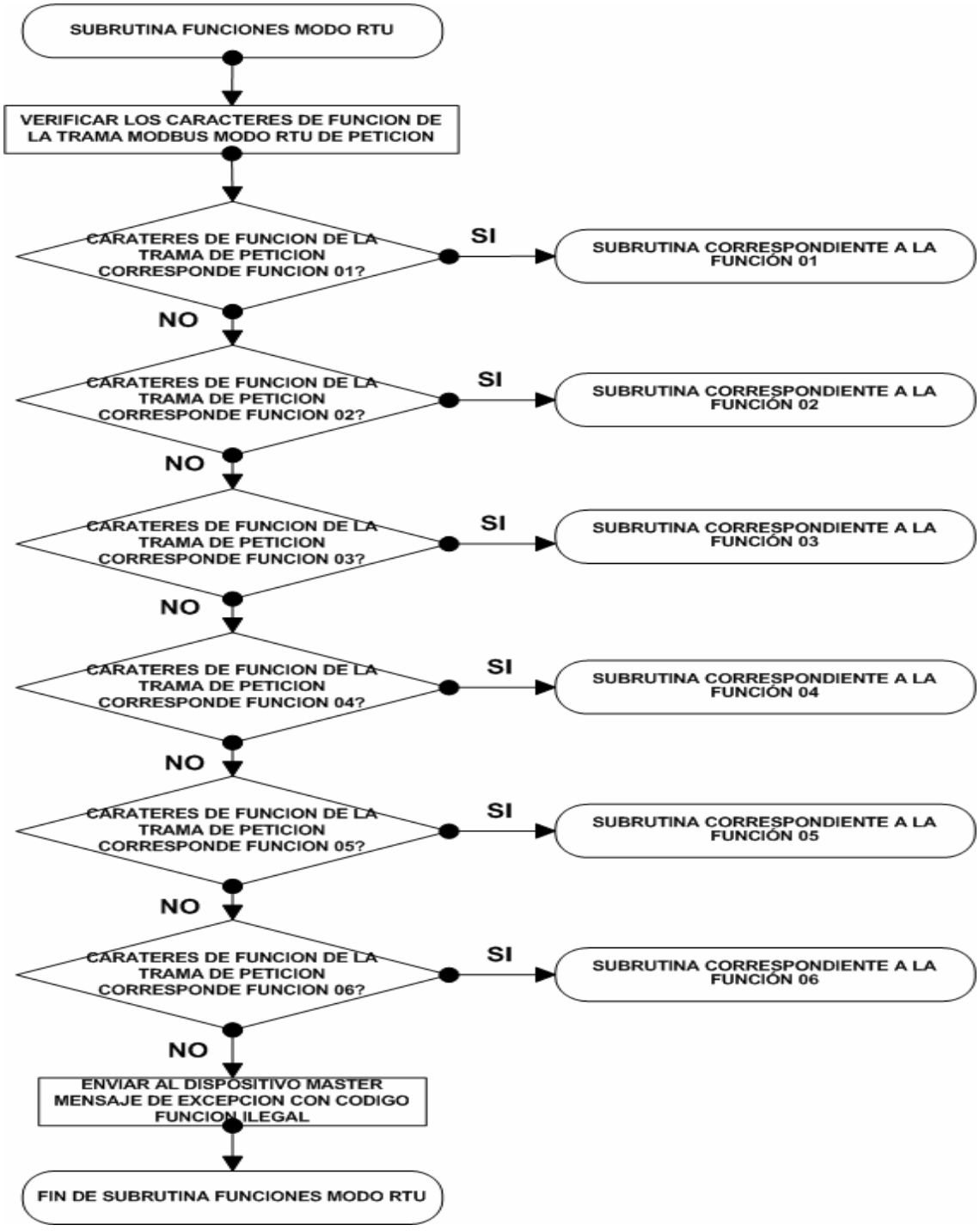

Figura 2.13 FLUJOGRAMA DEL PROGRAMA DE FUNCIONES MODO RTU

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Verificar los caracteres de función de la trama MODBUS de petición

Verificar el carácter correspondientes a la función en la trama MODBUS de Petición y se dirige a las subrutinas especificas de funciones.

Enviar al Dispositivo Master mensaje de excepción con código función ilegal

Elaborar un mensaje de excepción con el código función ilegal.
Calcular el CRC del mensaje de excepción.
Enviar el mensaje de excepción con un tiempo de 6.63ms entre carácter y carácter al Dispositivo Master

2.2.2.2.- SUBRUTINA CORRESPONDIENTE A LA FUNCIÓN 01

Esta función tiene como tarea leer el estado de las salidas digitales. En el caso de los Dispositivos Esclavos solamente existe una salida digital, a la cual se le asignó la dirección 1 correspondiente a 00001. De ingresar a esta subrutina, el primer paso es la revisión nuevamente del carácter correspondiente al campo de dirección del Dispositivo Esclavo en la Trama MODBUS de petición en modo RTU ya que esta función no soporta BROADCAST. Si es una trama BROADCAST no ejecutará acción alguna y saldrá al programa principal; Caso contrario, se construirá un trama MODBUS de respuesta con el estado de la salida digital y se enviará al Dispositivo Master; después regresará al programa principal.

En la Figura 2.14, se muestra el flujograma del programa correspondiente a la subrutina de la Función 01.

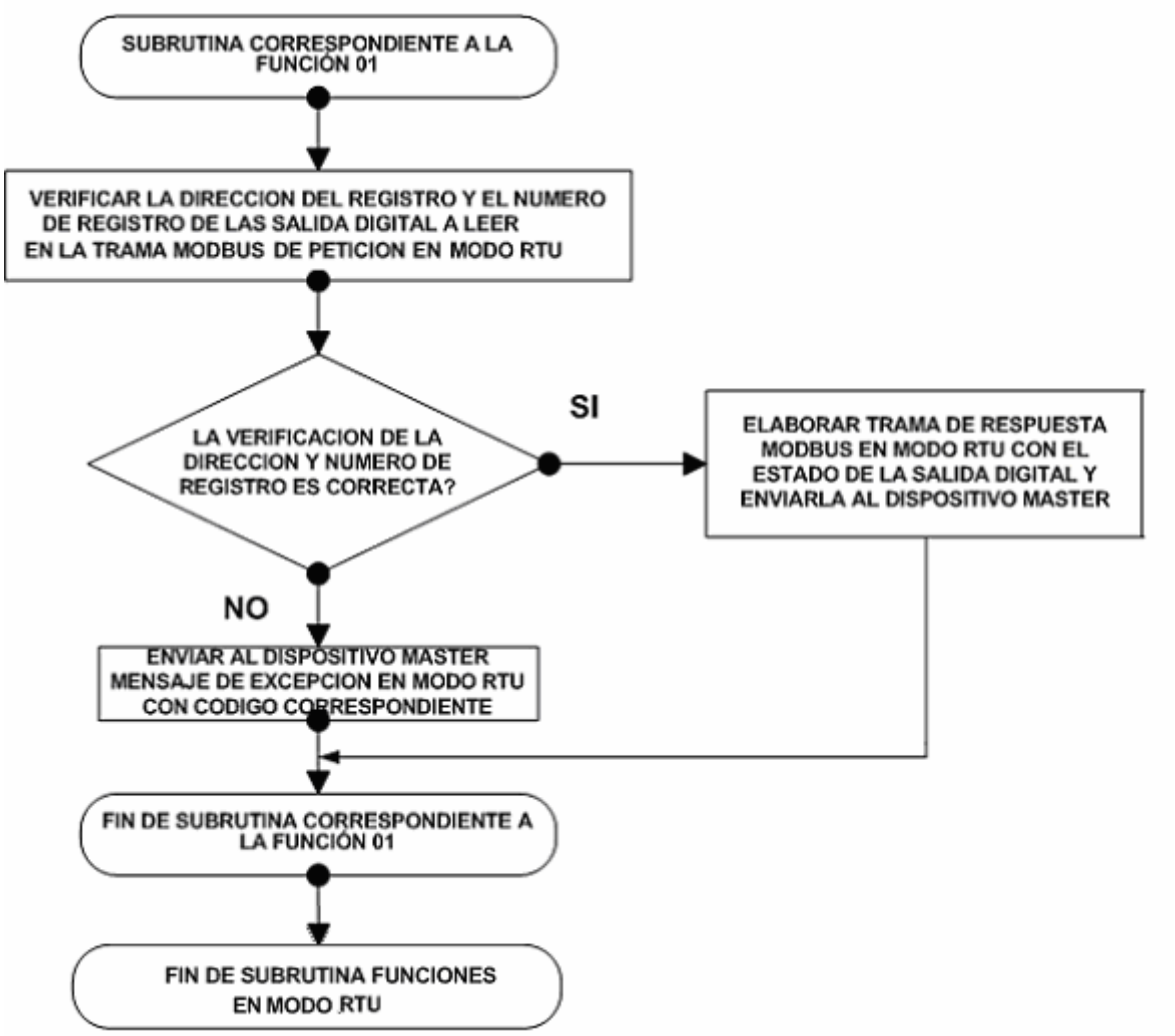

Figura 2.14 FLUJOGRAMA DE LA SUBRUTINA DE LA FUNCIÓN 01

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Elaborar trama de respuesta MODBUS en modo RTU con el estado de la salida digital y enviarla al dispositivo master

Elaborar una Trama MODBUS con el estado actual de la salida digital.
Calcular el CRC de la trama MODBUS.
Enviar la trama MODBUS de respuesta con un tiempo de 6.63ms entre carácter y carácter al Dispositivo Master.

Enviar al Dispositivo Master mensaje de excepción en modo RTU con código correspondiente

Elaborar un mensaje de excepción si la dirección de la salida digital a leer es incorrecta.
Elaborar un mensaje de excepción si el número de la salida digital a leer es incorrecta.
Calcular el CRC del mensaje de excepción.
Enviar el mensaje de excepción con un tiempo de 6.63ms entre carácter y carácter al Dispositivo Master

2.2.2.3.- SUBRUTINA CORRESPONDIENTE A LA FUNCIÓN 02

Esta función tiene como tarea leer el estado de las entradas digitales. En el caso de los Dispositivos Esclavos solamente existe una entrada digital, a la cual se le asignó la dirección 1 que correspondiente a 10001. De ingresar a esta subrutina, el primer paso es la revisión nuevamente del carácter correspondiente al campo de dirección del Dispositivo Esclavo en la Trama MODBUS de petición en modo RTU ya que esta función no soporta BROADCAST. Si es una trama BROADCAST no ejecutará acción alguna y saldrá al programa principal; Caso contrario se construirá un trama MODBUS de respuesta con el estado de la entrada digital y se enviará al Dispositivo Master; después regresará al programa principal.

En la Figura 2.15 se muestra el flujograma del programa correspondiente a la subrutina de la Función 02.

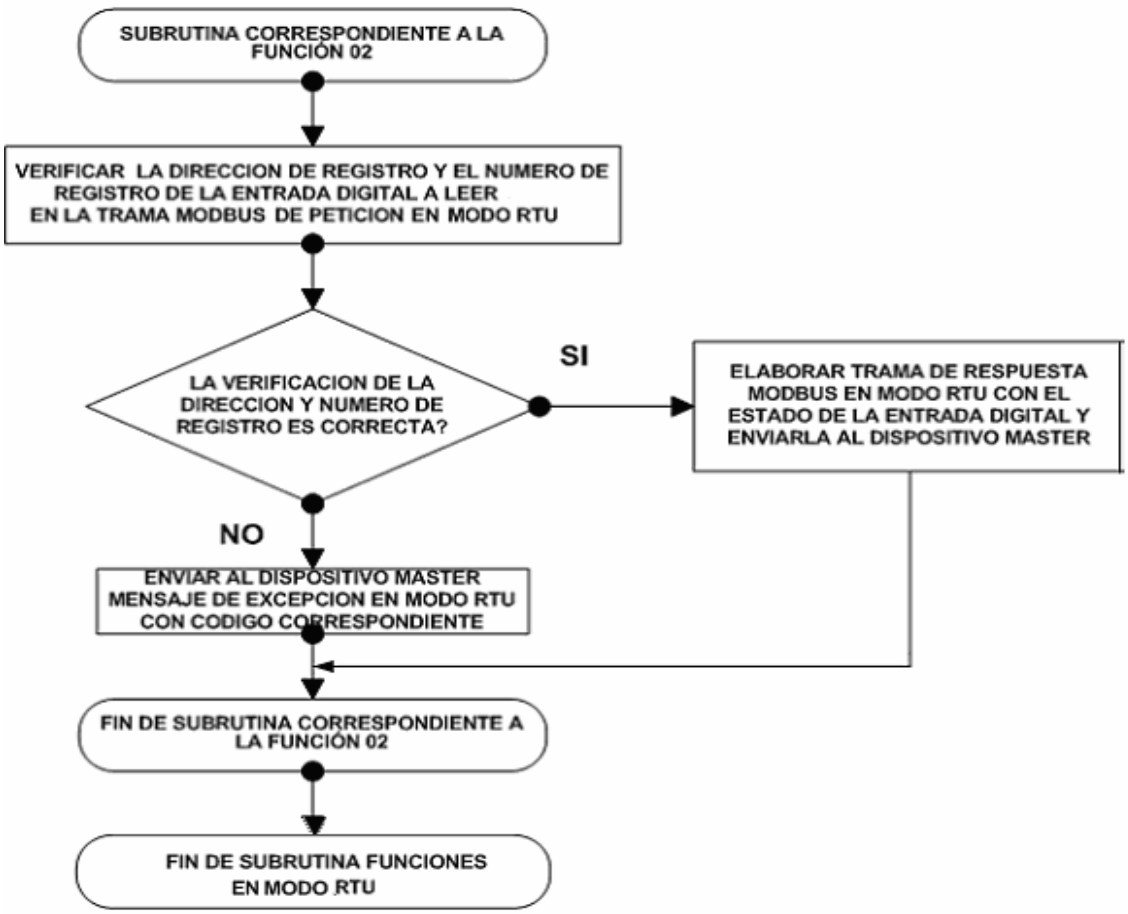

Figura 2.15 FLUJOGRAMA DE LA SUBRUTINA DE LA FUNCIÓN 02

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Elaborar trama de respuesta MODBUS en modo RTU con el estado de la entrada digital y enviarla al dispositivo master

Elaborar una Trama MODBUS con el estado actual de la entrada digital.
Calcular el CRC de la trama MODBUS.
Enviar la trama MODBUS de respuesta con un tiempo de 6.63ms entre carácter y carácter al Dispositivo Master.

Enviar al Dispositivo Master mensaje de excepción en modo RTU con código correspondiente

Elaborar un mensaje de excepción si la dirección de la entrada digital a leer es incorrecta.
Elaborar un mensaje de excepción si el número de la entrada digital a leer es incorrecta.
Calcular el CRC del mensaje de excepción.
Enviar el mensaje de excepción con un tiempo de 6.63ms entre carácter y carácter al Dispositivo Master.

2.2.2.4.- SUBRUTINA CORRESPONDIENTE A LA FUNCIÓN 03

Esta función tiene como tarea leer registros de memoria. En el caso de los Dispositivos Esclavos solamente existe un registro de memoria, a la cual se le asignó la dirección 1 que corresponde a 40001. De ingresar a esta subrutina, el primer paso es la revisión nuevamente de los caracteres correspondientes al campo de dirección del Dispositivo Esclavo en la Trama MODBUS de petición en modo RTU ya que esta función no soporta BROADCAST. Si es una trama BROADCAST no ejecutará acción alguna y saldrá al programa principal; Caso contrario, se construirá un trama MODBUS de respuesta con el valor del registro de memoria y se enviará al dispositivo Master; después regresará al programa principal.

En la Figura 2.16 se muestra el flujograma del programa correspondiente a la subrutina de la Función 03.

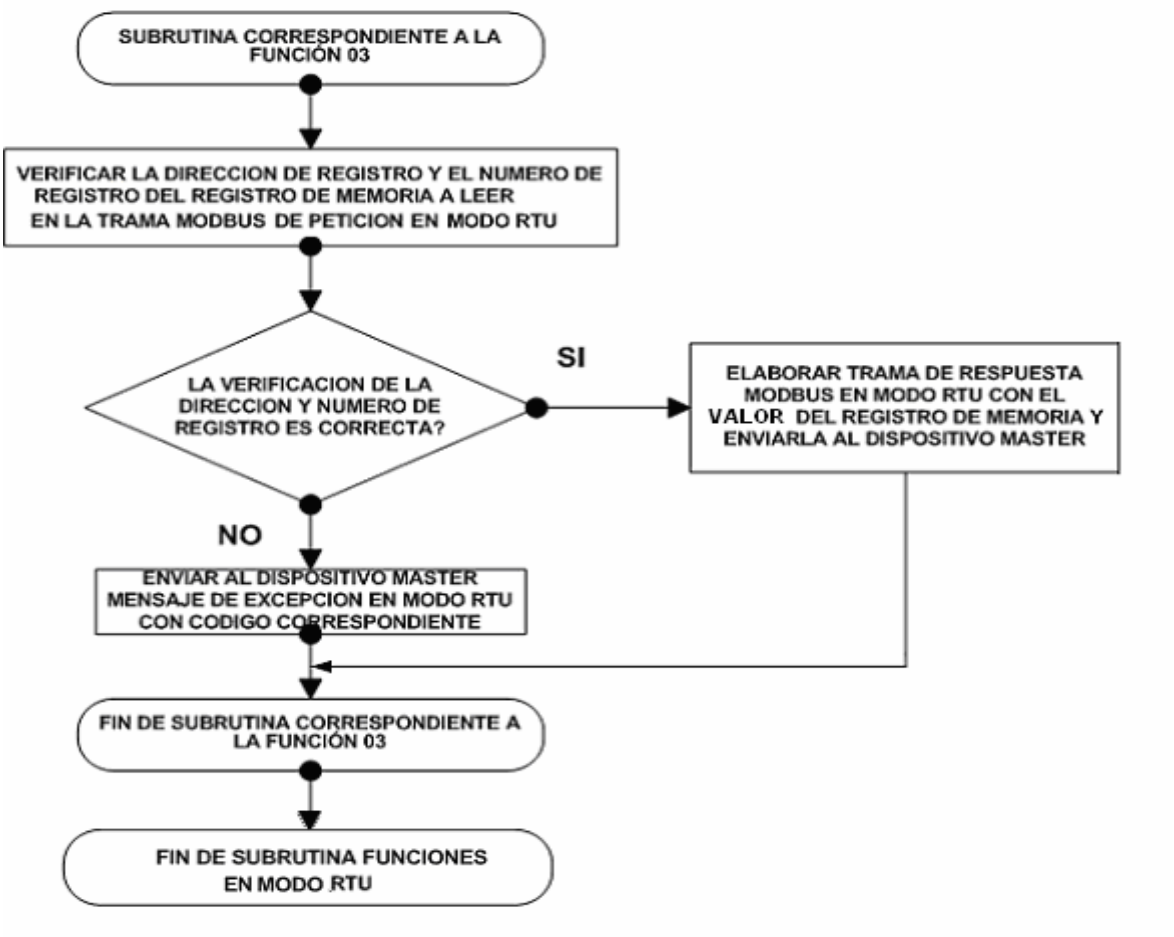

Figura 2.16 FLUJOGRAMA DE LA SUBRUTINA DE LA FUNCIÓN 03

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Elaborar trama de respuesta MODBUS en modo RTU con el valor del registro de memoria y enviarla al dispositivo master

Elaborar una Trama MODBUS con el valor del registro de memoria.
Calcular el CRC de la trama MODBUS.
Enviar la trama MODBUS de respuesta con un tiempo de 6.63ms entre carácter y carácter al Dispositivo Master.

Enviar al Dispositivo Master mensaje de excepción en modo RTU con código correspondiente

Elaborar un mensaje de excepción si la dirección del registro de memoria a leer es incorrecta.
Elaborar un mensaje de excepción si el número del registro de memoria a leer es incorrecta.
Calcular el CRC del mensaje de excepción.
Enviar el mensaje de excepción con un tiempo de 6.63ms entre carácter y carácter al Dispositivo Master.

2.2.2.5.- SUBRUTINA CORRESPONDIENTE A LA FUNCIÓN 04

Esta función tiene como tarea leer registros de entrada. En el caso de los Dispositivos Esclavos solamente existe un registro de memoria, a la cual se le asignó la dirección 1 que corresponde a 30001. De ingresar a esta subrutina, el primer paso es la revisión nuevamente de los caracteres correspondientes al campo de dirección de Dispositivo Esclavo en la Trama MODBUS de petición en modo RTU ya que esta función no soporta BROADCAST. Si es una trama BROADCAST no ejecutará acción alguna y saldrá al programa principal; Caso contrario, se construirá un trama MODBUS de respuesta con el valor del registro de entrada y se enviará al dispositivo Master; después regresará al programa principal.

En la Figura 2.17 se muestra el flujograma del programa correspondiente a la subrutina de la Función 04.

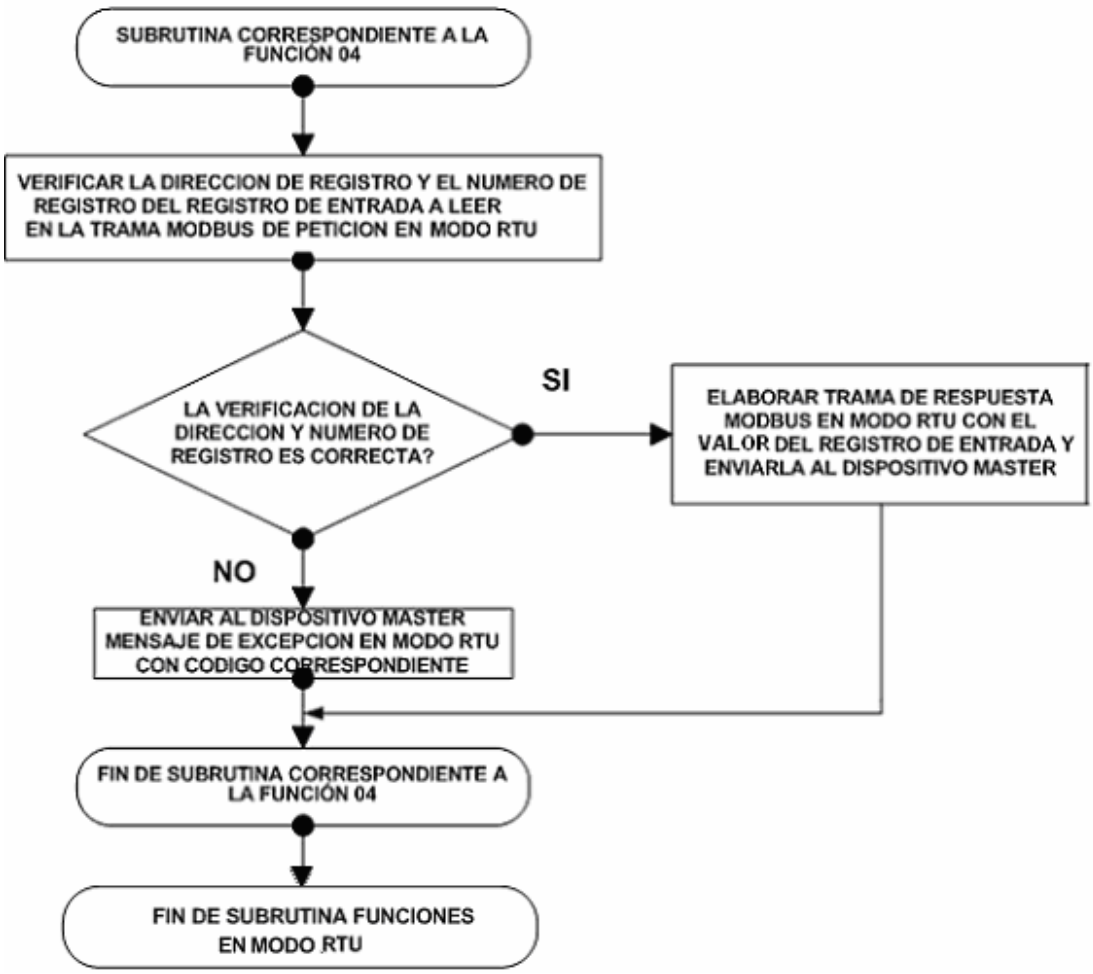

Figura 2.17 FLUJOGRAMA DE LA SUBRUTINA DE LA FUNCIÓN 04

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Elaborar trama de respuesta MODBUS en modo RTU con el valor del registro de entrada y enviarla al dispositivo master

Elaborar una Trama MODBUS con el valor del registro de entrada.
Calcular el CRC de la trama MODBUS.
Enviar la trama MODBUS de respuesta con un tiempo de 6.63ms entre carácter y carácter al Dispositivo Master.

Enviar al Dispositivo Master mensaje de excepción en modo RTU con código correspondiente

Elaborar un mensaje de excepción si la dirección del registro de entrada a leer es incorrecta.
Elaborar un mensaje de excepción si el número del registro de entrada a leer es incorrecto.
Calcular el CRC del mensaje de excepción.
Enviar el mensaje de excepción con un tiempo de 6.63ms entre carácter y carácter al Dispositivo Master.

2.2.2.6.- SUBRUTINA CORRESPONDIENTE A LA FUNCIÓN 05

Esta función tiene como tarea forzar una salida digital, En el caso de los Dispositivos Esclavos se asignó la dirección 1 que corresponde a 00001. En primer lugar esta subrutina, procederá a comparar los 2 caracteres correspondientes a la dirección alta y baja de la salida digital que se quiera forzar de la trama MODBUS de petición en modo RTU, si estos valores son validos por el Dispositivo Esclavo, revisará los 2 caracteres correspondientes al campo de datos alto y bajo de la trama MODBUS de petición, si estos caracteres son FF00 procederá a forzar en 1 lógico la salida digital y si es 0000 procederá a forzar en 0 lógico la salida digital, realizado este paso procederá a revisar el carácter correspondientes a la dirección del Dispositivo Esclavo, Si es una trama BROADCAST saldrá al programa principal; caso contrario el Dispositivo Esclavo enviará la misma trama de petición como trama de respuesta al Dispositivo Master; después regresará al programa principal.

En la Figura 2.18 se muestra el flujograma del programa correspondiente a la subrutina de la Función 05.

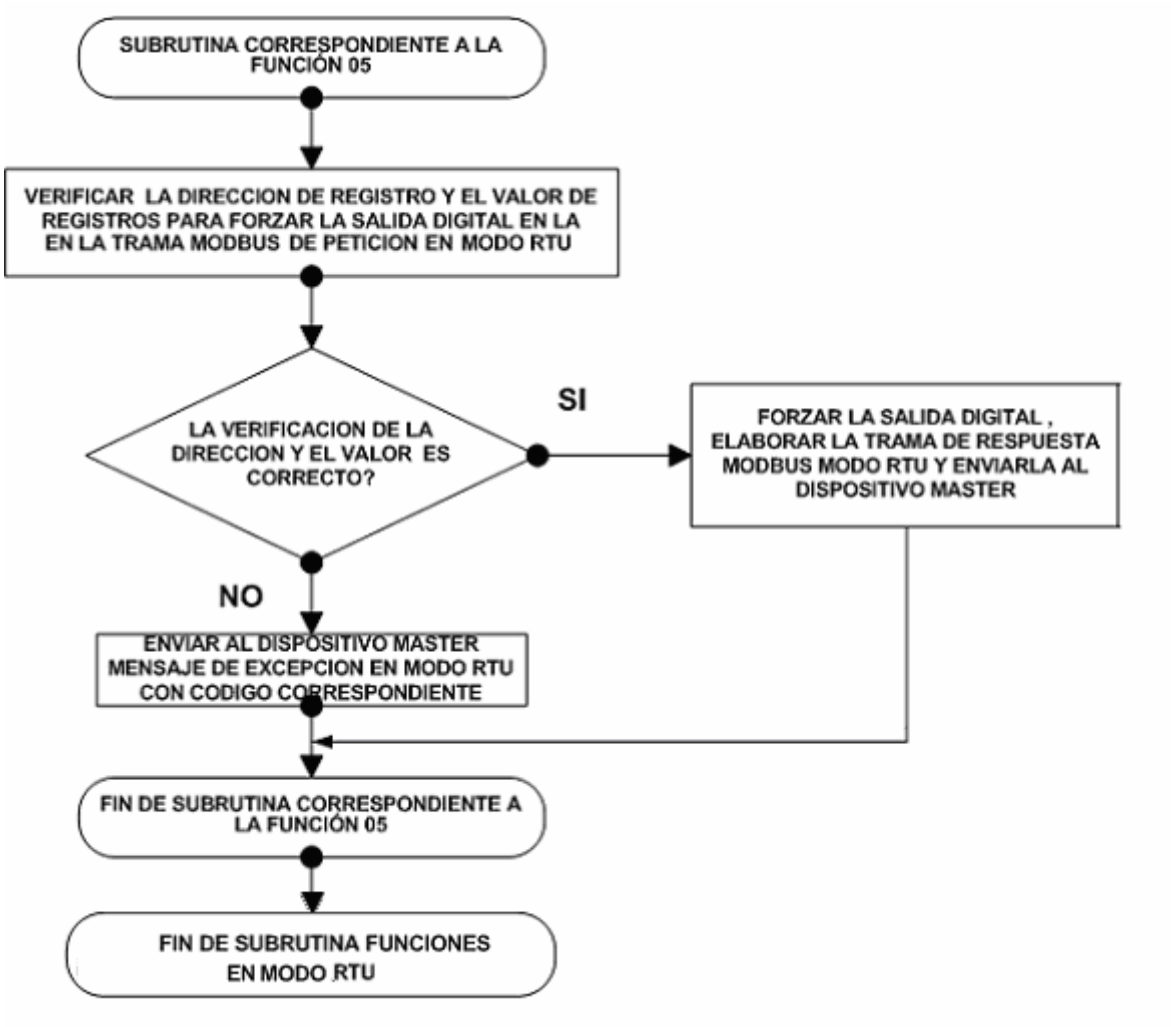

Figura 2.18 FLUJOGRAMA DE LA SUBRUTINA DE LA FUNCIÓN 05

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Forzar la salida digital ,elaborar trama de respuesta MODBUS RTU y enviarla al dispositivo master

Forzar uno lógico o cero lógico las salida digital.
Elaborar una Trama MODBUS de respuesta igual a la trama MODBUS de petición.
Enviar la trama MODBUS de respuesta con un tiempo de 6.63ms entre carácter y carácter al Dispositivo Master.

Enviar al Dispositivo Master mensaje de excepción en modo RTU con código correspondiente

Elaborar un mensaje de excepción si la dirección de la salida digital a forzar es incorrecta.
Elaborar un mensaje de excepción si el valor de la acción que se desee realizar en la salida digital es incorrecta.
Calcular el CRC del mensaje de excepción.
Enviar el mensaje de excepción con un tiempo de 6.63ms entre carácter y carácter al Dispositivo Master.

2.2.2.7.- SUBRUTINA CORRESPONDIENTE A LA FUNCIÓN 06

Esta función tiene como tarea programar un registro de memoria. En el caso de los Dispositivos Esclavos, se asignó la dirección 1 que corresponde a 30001.

En primer lugar esta subrutina, procederá a comparar los 2 caracteres correspondientes a la dirección alta y baja del registro de memoria que se quiera programar de la trama MODBUS de petición en modo RTU, si estos valores son validados por el Dispositivo Esclavo, revisará los 2 caracteres correspondientes al campo de datos alto y bajo de la trama MODBUS de petición, si estos caracteres son permitidos se procederá a programar el Dispositivo Esclavo. A continuación se revisará el carácter correspondiente a la dirección del Dispositivo Esclavo. Si es una trama BROADCAST saldrá al programa principal; caso contrario el Dispositivo Esclavo enviará la misma trama de petición como trama de respuesta al Dispositivo Master; después regresará al programa principal.

En la Figura 2.19 Se muestra el flujograma del programa correspondiente a la subrutina de la Función 06.

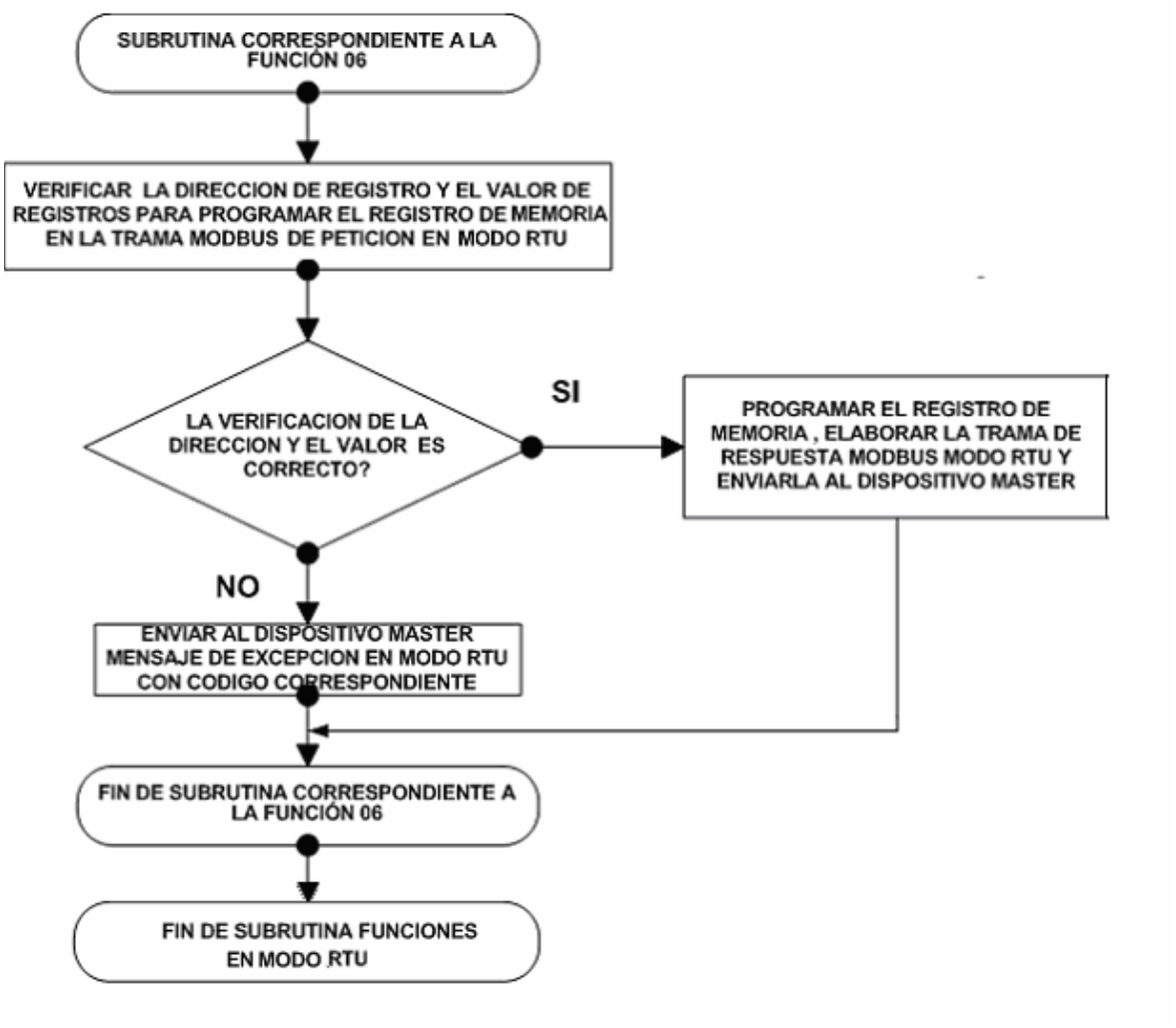

Figura 2.19 FLUJOGRAMA SUBRUTINA DE LA FUNCIÓN 06

Las subrutinas se explican en forma ampliada en lenguaje estructurado a continuación.

Programar el registro de memoria, elaborar la trama de respuesta MODBUS modo RTU y enviarla al dispositivo master

Programar el registro de memoria.
Elaborar una Trama MODBUS de respuesta igual a la trama MODBUS de petición.
Enviar la trama MODBUS de respuesta con un tiempo de 6.63ms entre carácter y carácter al Dispositivo Master.

Enviar al Dispositivo Master mensaje de excepción en modo ASCII con código correspondiente

Elaborar un mensaje de excepción si la dirección del registro de memoria a programar es incorrecta.
Elaborar un mensaje de excepción si el valor que se desee programar en el registro de memoria es incorrecto.
Calcular el CRC del mensaje de excepción.
Enviar el mensaje de excepción con un tiempo de 6.63ms entre carácter y carácter al Dispositivo Master.

2.3 IMPLEMENTACIÓN DEL HMI

2.3.1 INTRODUCCIÓN

1. Cuando los seres humanos y los computadores interactúan lo hacen a través de un medio o interfaz hombre – máquina, que definimos como HMI.

2.3.2 INTERFAZ HOMBRE-MÁQUINA.

A continuación se describe el Funcionamiento de la interfaz creada y los criterios utilizados en el mismo:

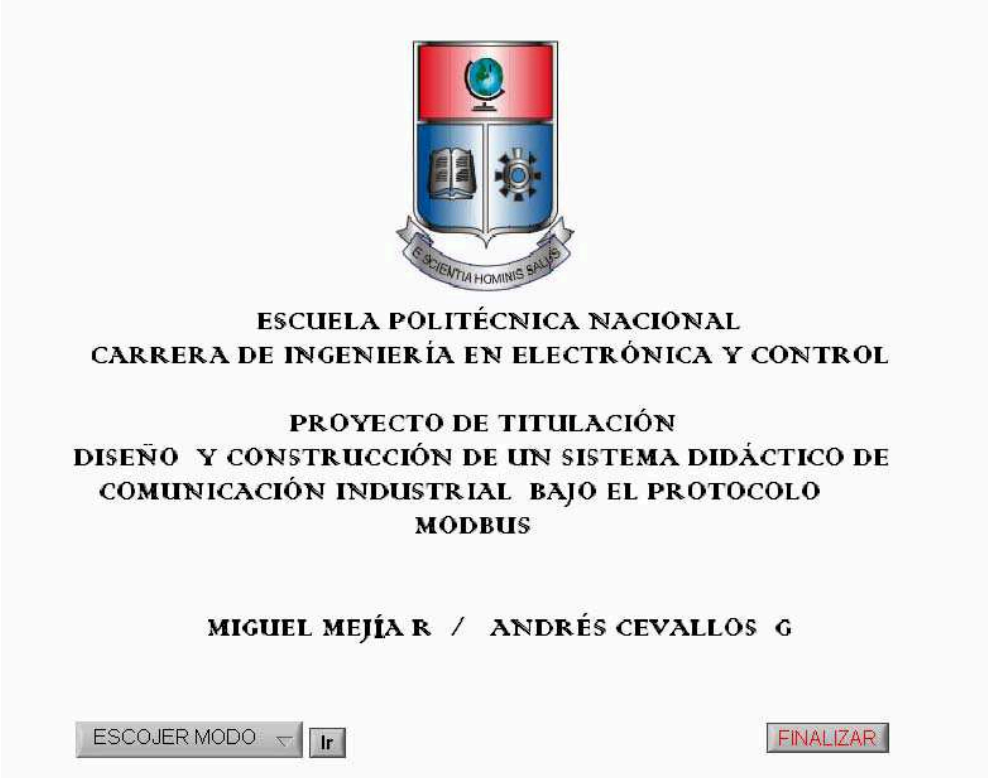

Figura 2.20 VENTANA DE INICIO

Al ejecutar el programa "HMI SISTEMA DIDÁCTICO DE COMUNICACIÓN INDUSTRIAL BAJO EL PROTOCOLO MODBUS" se visualiza la pantalla inicio Figura 2.20, donde se puede escoger en el menú los modos de comunicación además del tipo de visualización. Así podemos escoger entre:

- MODO NORMAL ASCII
- MODO NORMAL RTU

- MODO DIDÁCTICO ASCII
- MODO DIDÁCTICO RTU

2.3.2.1 INTERFAZ MODO NORMAL

El HMI puede trabajar en modo normal que es un similar al que se usa en las plantas industriales. La idea principal de este modo es de familiarizar al estudiante con interfaces comunes en puestos de trabajo.

Para este caso vamos a analizar la ventana en modo ASCII, para el otro modo la pantalla es exactamente igual diferenciándose en que el DIAGRAMA[14] programado utiliza la *comunicación serial RTU* con sus respectivos tiempos y datos.

2.3.2.1.1 VIZUALIZACIÓN "MODO NORMAL"

Figura 2.21 VENTANA VIRTUAL NORMAL MODO ASCII

[14] LABVIEW utiliza lenguaje G, este es un lenguaje de programación gráfico basado en diagramas.

En esta ventana podemos ver las seis funciones implementadas para cada Esclavo MODBUS:

- Leer estado salida digital
- Leer estado entrada digital
- Leer registros de memoria
- Leer registros de entrada
- Forzar una salida digital
- Programar un registro

Como se puede apreciar en la Figura 2.21 tenemos además las dos últimas Funciones en modo de difusión BROADCAST.

Para cualquiera de las Funciones mencionadas (Incluido BROADCAST) contamos con un Display de texto, el cual nos indica cuando la trama está siendo procesada. Esto para que el usuario no trate de enviar una trama mientras exista comunicación establecida, y la PC esté esperando datos en el puerto serial.

Figura 2.22 DISPLAY VIRTUAL "MENSAJE PROCESANDO TRAMA"

La programación de esta pantalla se lo realiza por funciones. Como se muestra en la Figura 2.23 siguiente, al presionar el botón asociado a la función, el lazo "*case"* cambiará a estado "TRUE", en este correrá el programa desarrollado para la comunicación serial cargando los datos predeterminados de la función para, realizando así la petición en caso de lectura y acción en caso de escritura.

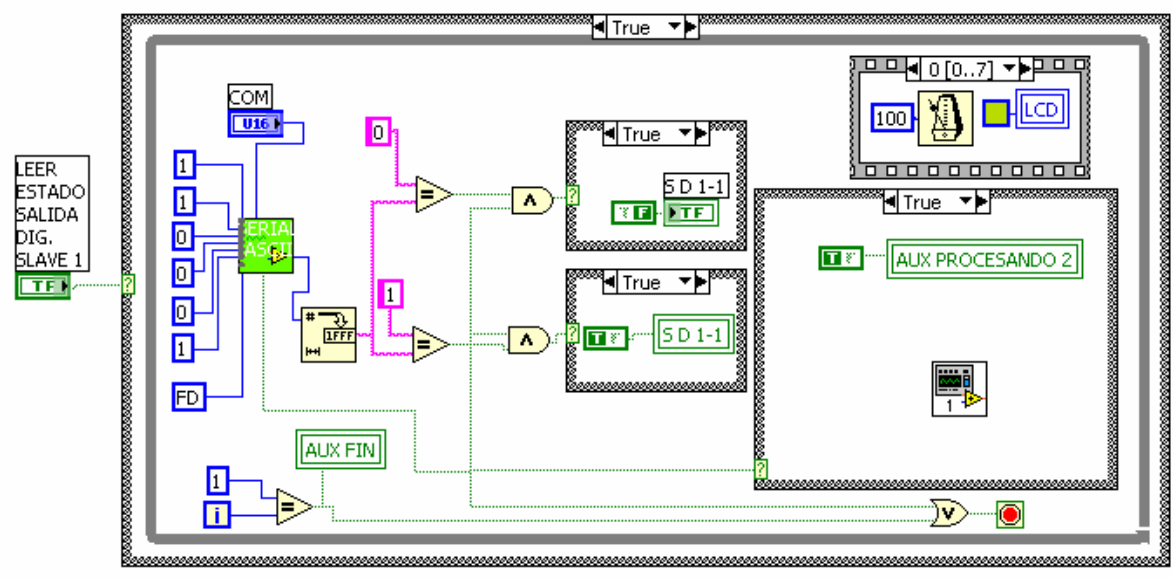

Figura 2.23 DIAGRAMA EJEMPLO EJECUCIÓN DE UNA FUNCIÓN

La Estructura Case mostrada en la Figura 2.23 (parte inferior derecha) sirve para desplegar una de las pantallas que nos indicaran si se estableció la comunicación o si hubo un error en la misma. Para esto se cuenta con un indicador en el software SERIAL ASCII que nos permite saber cuando se ha establecido la comunicación y si la trama es verdadera.

Para el caso de no tener enlace de la PC al Dispositivo MASTER se tendrá el mensaje de la Figura 2.24:

Figura 2.24 VENTANA DE INFORMACIÓN "ERROR DE COMUNICACIÓN"

Si se ha establecido la comunicación y la trama es aceptada por el HMI, se tendrá el mensaje de la Figura 2.25 siguiente:

Figura 2.25 VENTANA DE INFORMACIÓN "TRAMA RECIBIDA"

2.3.2.2 INTERFAZ MODO DIDÁCTICO

El HMI puede trabajar en un Modo Didáctico, en este se muestran al usuario las tramas MODBUS que se envíen y que llegan. La idea principal de este modo es utilizar este sistema como herramienta de estudio de este protocolo.

En este modo, el usuario podrá ver todos los campos de la trama MODBUS, la manera de formar una, como se envían y como llegan las mismas a un Dispositivo; así también la manera como se calcula el Campo de Chequeo de Errores

Para este caso vamos a analizar la ventana en modo RTU, para el otro modo la pantalla es similar diferenciándose en que el DIAGRAMA programado utiliza la comunicación serial diferente con sus respectivos tiempos y datos como se indicó anteriormente.

2.3.2.2.1 VIZUALIZACIÓN “MODO DIDÁCTICO"

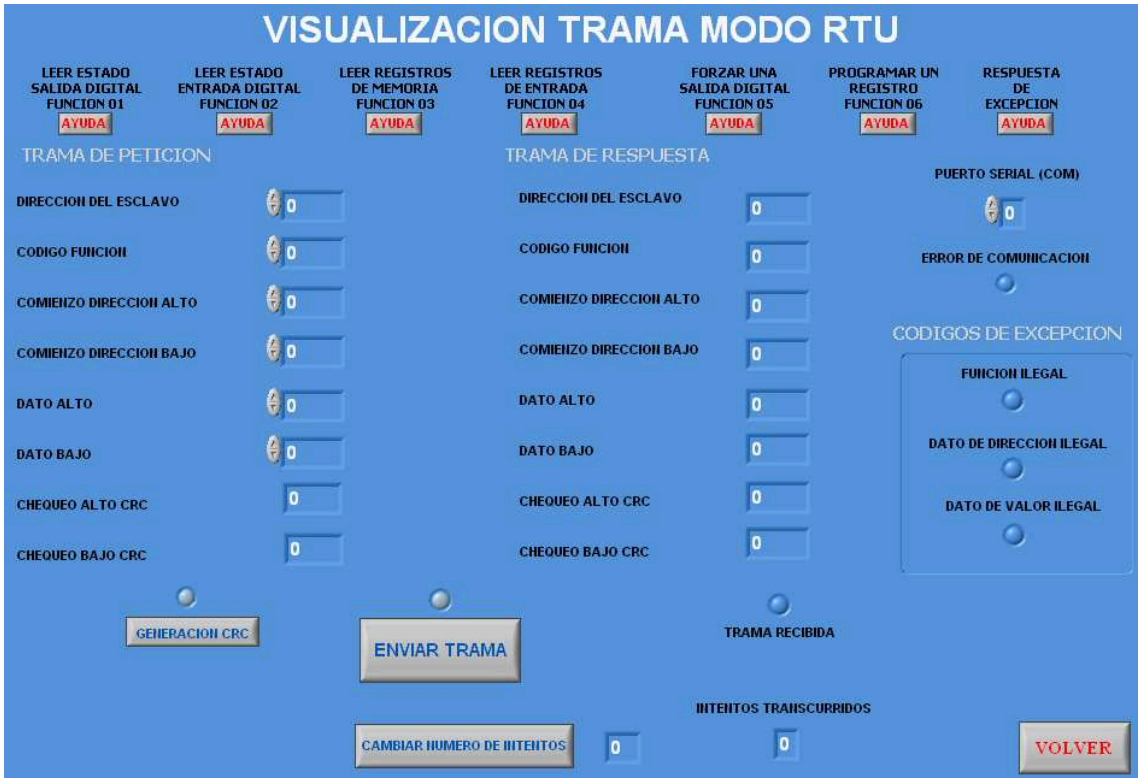

Figura 2.26 VENTANA VIRTUAL “DIDÁCTICO MODO RTU”

En esta ventana podemos ver todos los campos de las Tramas MODBUS de Petición y de Respuesta, ingresando de forma manual valores numéricos a dichos campos a excepción del Campo de Chequeo de Errores que se calcula automáticamente presionando el botón “*GENERACIÓN CRC*” para RTU y “*GENERACIÓN LRC*” para ASCII.[15]

Como se puede apreciar en la Figura 2.26, se tienen indicadores LEDs para saber si la trama se ha recibido y de igual manera si han ocurrido errores.

El usuario puede ingresar el número de intentos entre 1 y 10 (recomendado). Cada intento significa que el HMI enviará una trama y esperará recibir otra en el Pórtico Serial. El sistema envía por default una sola vez, si este parámetro no es cambiado.

[15] Para modo ASCII se ha incluido para cada campo, en petición y respuesta, los caracteres ASCII correspondientes al número HEX.

2.3.2.3. GENERACIÓN DE CHEQUEO DE ERRORES

2.3.2.3.1. GENERACIÓN DEL CRC

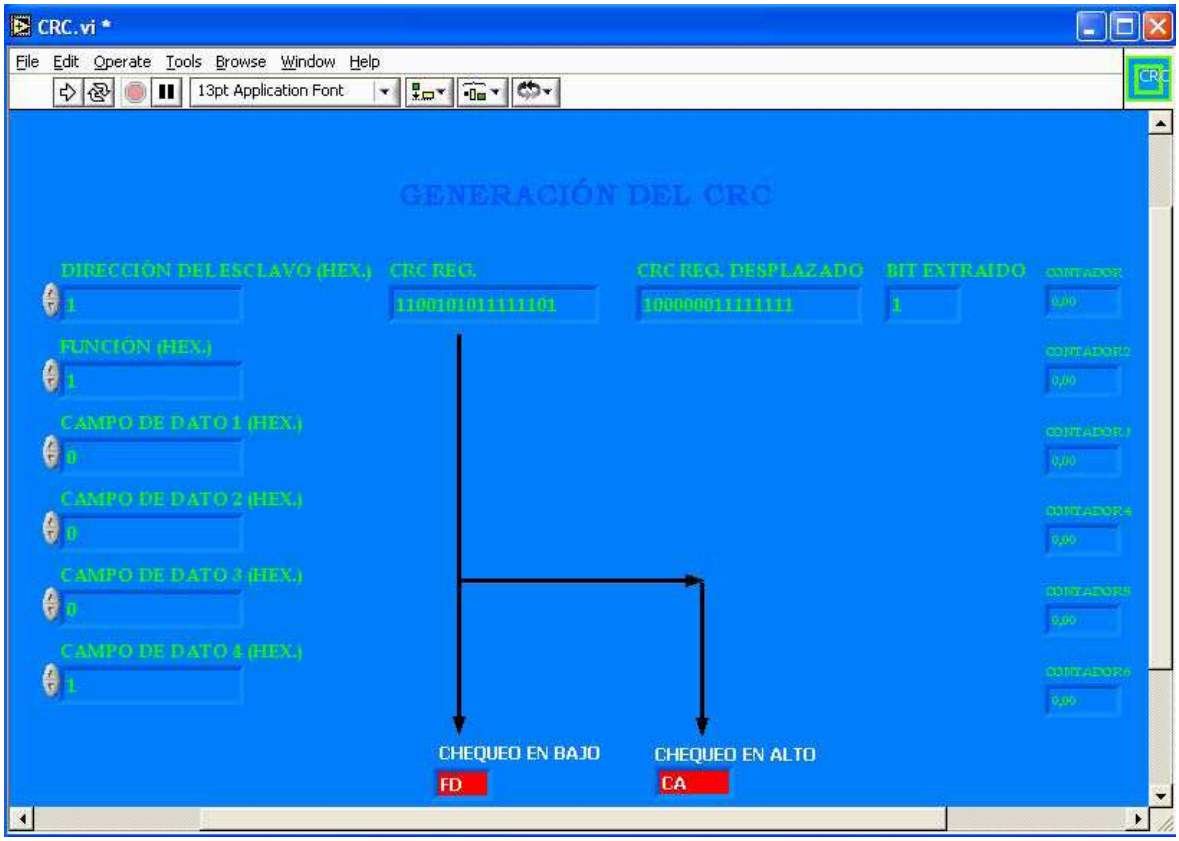

Figura 2.27 PANTALLA GENERACIÓN DEL CRC.

En esta pantalla de la Figura 2.27 se ingresan los datos de la trama MODBUS en modo RTU y automáticamente se generan los datos que deberán ir acompañados en los dos últimos campos, este VI[16] es utilizado para generar el CRC cuando se ingresan los datos en MODO DIDÁCTICO.

El campo CRC es de dos bytes. Para calcular el valor CRC se precarga un registro de 16 bits, CRC REG en la Figura 2.28, todos ellos a 1. Luego comienza un proceso que toma los sucesivos bytes del mensaje y los opera con el contenido del registro y actualiza éste con el resultado obtenido.

[16] Virtual Instrument, llamado así a todos los programas realizados en la plataforma de LABVIEW.

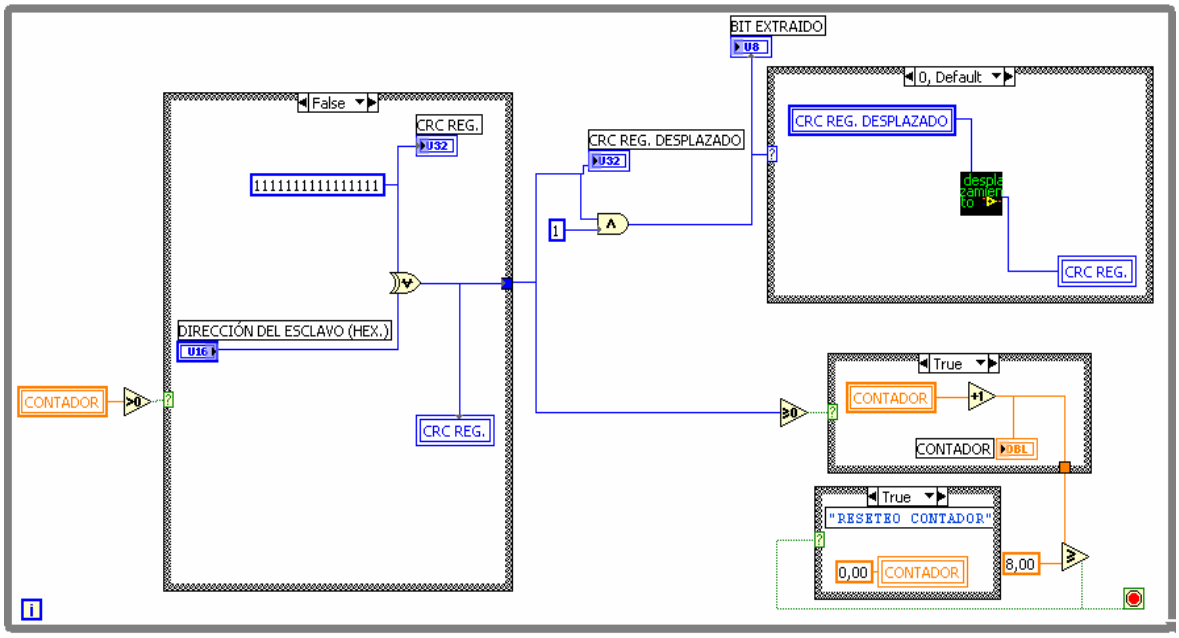

Figura 2.28 SECUENCIA 1 GENERACIÓN DEL CRC

En la Figura 2.28, durante la generación del CRC, se efectúa una operación booleana OR exclusivo (XOR) con el contenido del registro CRC REG. El LSB es extraído y examinado. Es importante señalar que solo en el primer desplazamiento se realiza el OR exclusivo con la trama, se puede ver en la misma Figura que el contador se va incrementando cada vez que el lazo se completa. Cuando el contador es mayor que cero el lazo CASE de la izquierda cambia a TRUE y solo se tienen el CRC REG sin ninguna operación.

Si el LSB extraído fuese un 1, se desplaza y posteriormente se realiza un XOR entre el registro y un valor fijo preestablecido[17], tal como muestra la Figura 2.29.

[17] El valor preestablecido es A001 hex., correspondiente al polinomio generador CRC16 'Inverso', que es el que se aplica al CRC MODBUS.

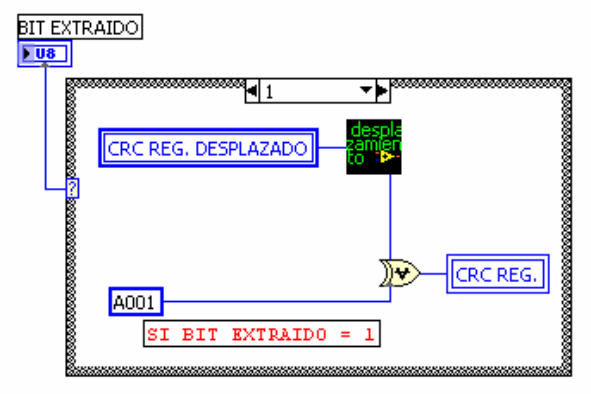

Figura 2.29 CASO BIT EXTRAIDO IGUAL A 1

Si el LSB fuese un 0, no se efectúa el XOR, tal como muestra la Figura 2.30

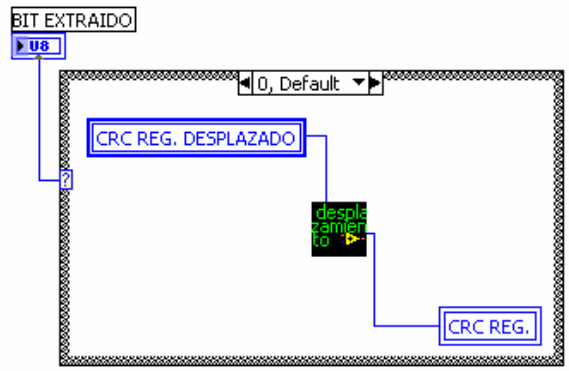

Figura 2.30 CASO BIT EXTRAIDO IGUAL A 1

Este proceso es repetido con el mismo registro CRC REG que se actualiza cada vez que se realiza una operación con el mismo hasta haber cumplido 8 desplazamientos.

Después del último desplazamiento (el octavo), el próximo byte o próximo campo de la trama es operado XOR con el valor actual del registro y el proceso se repite con ocho desplazamientos más. El contenido final del registro, después de que todos los bytes del mensaje han sido procesados, es el valor del CRC.

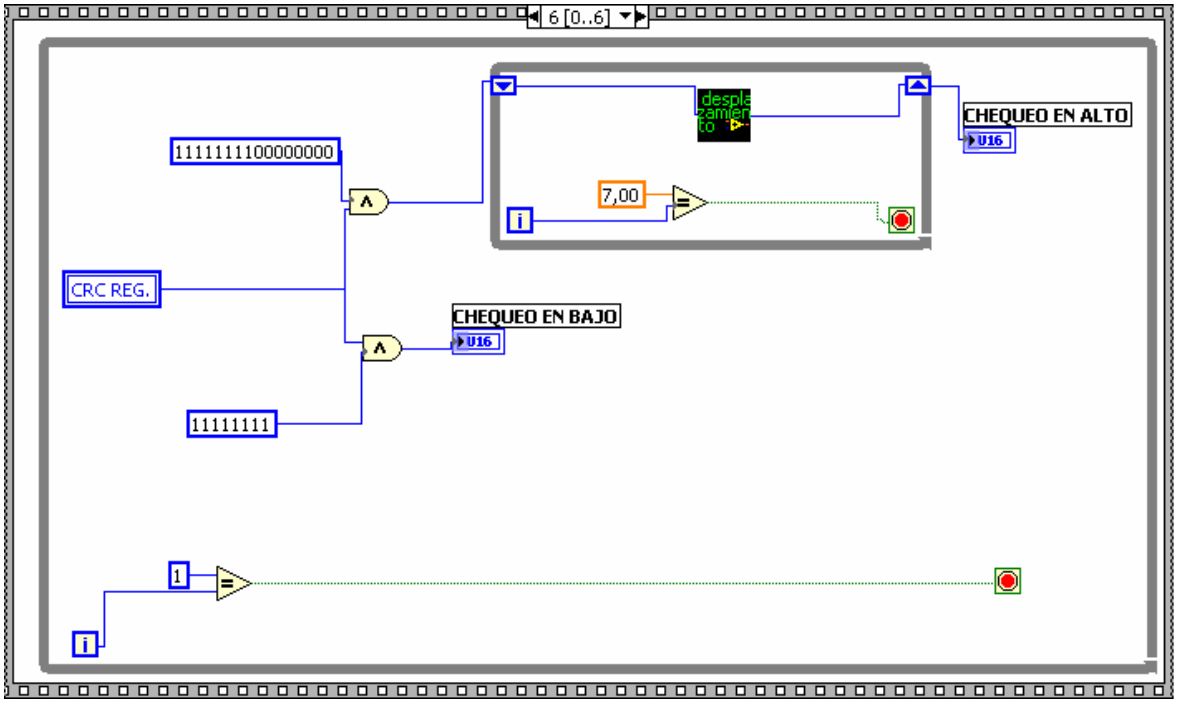

Figura 2.31 DIAGRAMA SECUENCIA FINAL CRC

El contenido del CRC REG es dividido en dos campos, un Byte en bajo y el otro en alto.

Para obtener el registro en bajo basta con hacer una operación AND con el número 11111111, es decir se elimina la parte alta.

El chequeo en alto se lo obtiene realizando una operación AND con el número 1111111100000000 y desplazando ocho veces hacia la derecha.

- Cuando el CRC es añadido al mensaje, primero se añade el byte de orden bajo seguido del byte de orden alto.

2.3.2.3.2. GENERACIÓN DEL LRC

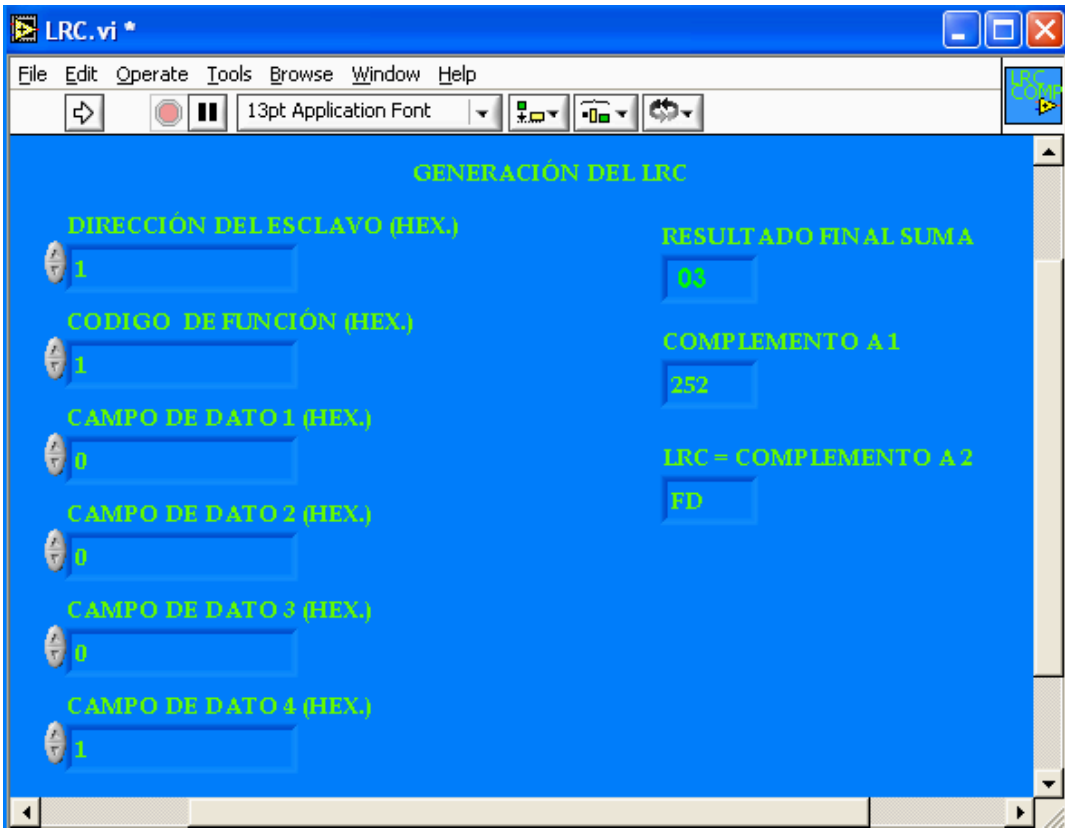

Figura 2.32 PANTALLA GENERACIÓN LRC.VI

En modo ASCII, los mensajes incluyen un campo de comprobación de error que está basado en un método de ***Comprobación Longitudinal Redundante*** (LRC). El campo LRC controla el contenido del mensaje, a excepción de los ':' ([3A]) del comienzo y el par CRLF ([0D] [0A]).

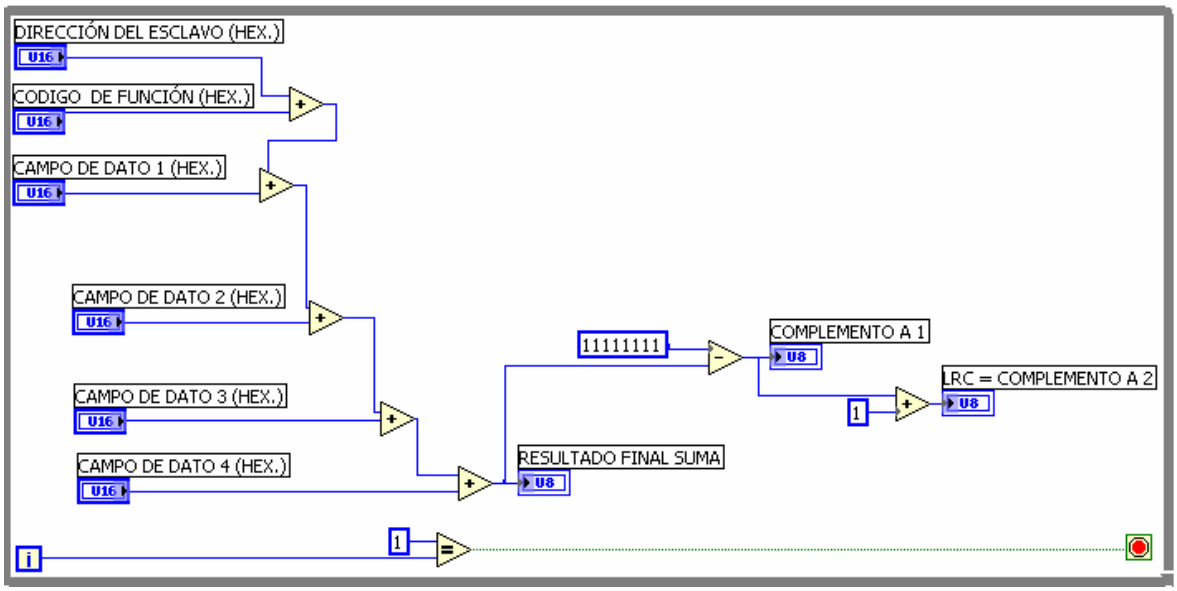

Figura 2.33 DIAGRAMA GENERACIÓN DEL LRC

El campo LRC es un byte, conteniendo un valor binario de ocho bits.

El valor LRC se calcula sumando todos bytes de la trama (parte izquierda de la Figura 2.33), descartando cualquier acarreo y luego complementando a dos el valor resultante. Esto se realiza sobre el contenido del campo de mensaje ASCII excluyendo el carácter ':' de comienzo del mensaje y excluyendo el par CRLF de final de mensaje.

En el siguiente capitulo se encuentra el diseño del hardware del sistema.

CAPÍTULO 3

DISEÑO Y CONSTRUCCION DEL HARDWARE DEL SISTEMA

CAPÍTULO 3

DISEÑO Y CONSTRUCCION DEL HARDWARE DEL SISTEMA

3.1 INTRODUCCION

En este capítulo se describe los múltiples factores que se tomaron en consideración para el diseño del hardware construido. Se incluyen diagramas esquemáticos, fotografías y otros para un mejor entendimiento.

En general se trata de dos módulos, uno que comprende el hardware necesario para el Dispositivo Master y otro para los Dispositivos Esclavos. Se aclara en esta introducción que el hardware de cada Dispositivo Esclavo es exactamente igual en los tres dispositivos.

El diagrama de bloques del sistema es el que se indica en la Figura 3.1:

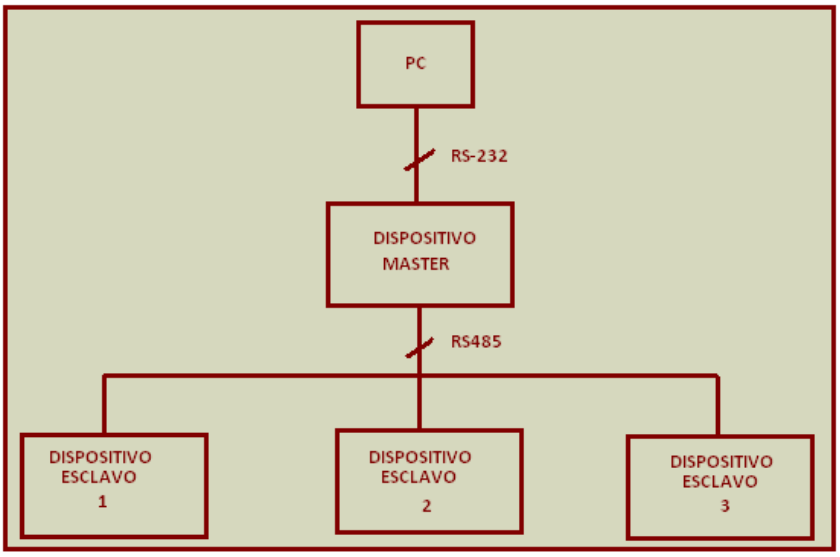

Figura 3.1 DIAGRAMA DE BLOQUES DEL SISTEMA

3.1.1. DISEÑO DEL HARDWARE REQUERIDO PARA EL DISPOSITIVO MASTER

Para el diseño de este dispositivo se partió de la idea que la selección del modo de comunicación (ASCII o RTU), se la realizaría físicamente y que la función del Dispositivo Master es de procesar las tramas recibidas del computador y enviar los datos mediante comunicación RS-485 a los Dispositivos Esclavos.

En la Figura 3.2 se muestra el diagrama de bloques de las partes funcionales del Dispositivo Master:

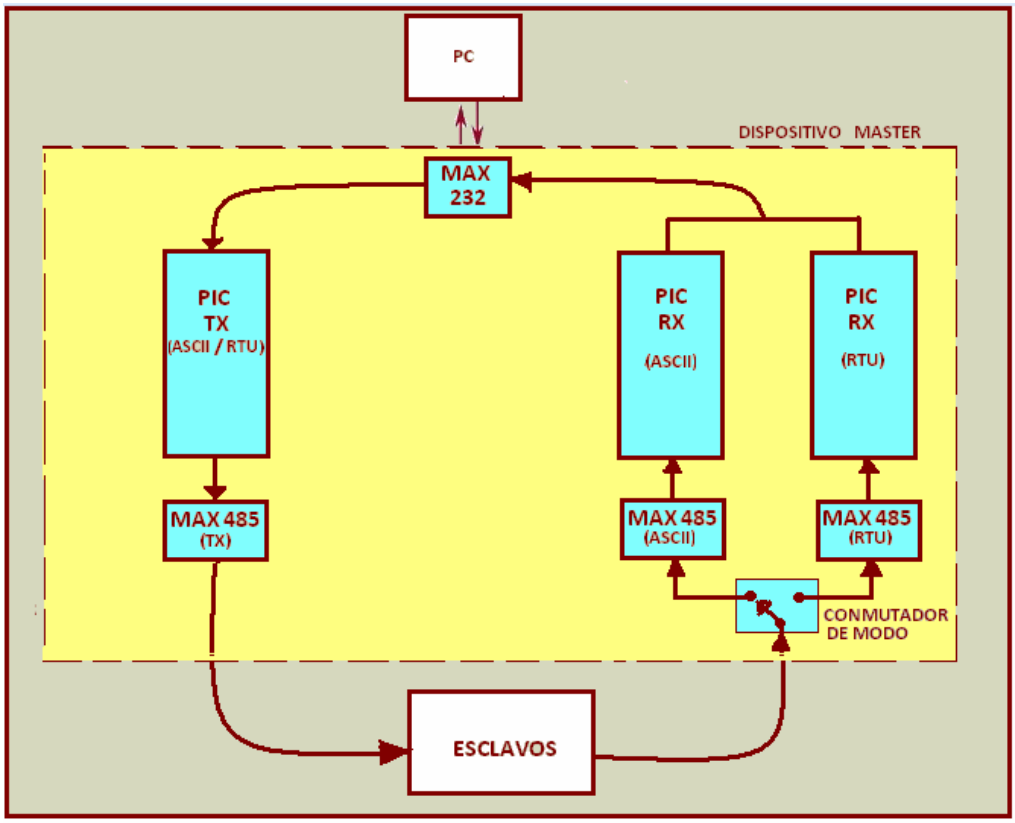

Figura 3.2 DIAGRAMA DE BLOQUES DEL DISPOSITIVO MASTER

Para todos los microcontroladores PIC se añadió un circuito para detectar fallas en los mismos (Detector de pulso faltante), el funcionamiento y el diseño del mismo se detallará más adelante.

Para la conexión entre la computadora y el Dispositivo Master se cuenta con un conector DB9. En la Figura 3.3 se muestra los pines utilizados por dicho conector

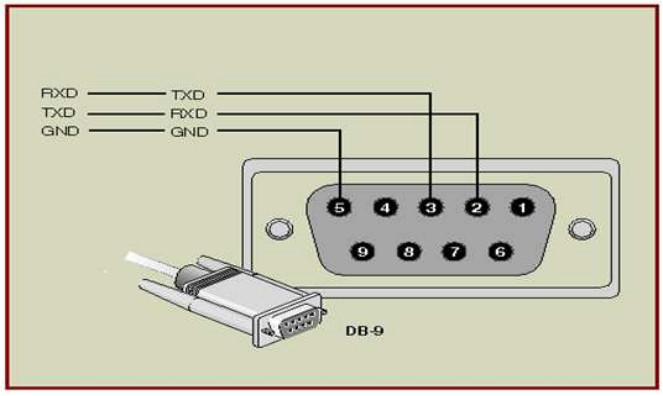

Figura 3.3. CONEXIÓN DEL DB9

En la Figura 3.4 se muestra la conexión entre el conector DB9 y el circuito integrado MAX232, y la conexión al microcontrolador PIC. Según las especificaciones que el fabricante sugiere, el circuito integrado MAX232 debe utilizar capacitores de 1µF.

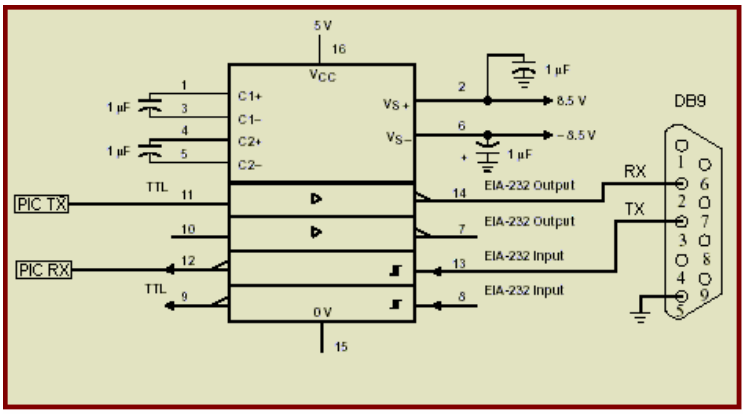

Figura 3.4 CONEXIÓN DEL MAX 232

Como muestra la Figura 3.2, el primer microcontrolador (*PIC TX*) se encarga de recibir los datos del MAX 232, mediante recepción serial, procesarlos y enviar a todos los Dispositivos Esclavos sobre RS 485.

En la tabla 3.1 se indica los estados que puede tener el circuito integrado MAX485 asociado al microcontrolador PIC de transmisión. En la Figura 3.5 se muestra la conexión del circuito mencionado con la señal de control y transmisión del microcontrolador.

SEÑAL DE CONTROL	ESTADO
0	HABILITA RECEPCION DEL MAX485
1	HABILITA TRANSMISION DEL MAX485

Tabla 3.1.- TABLA DE ESTADOS

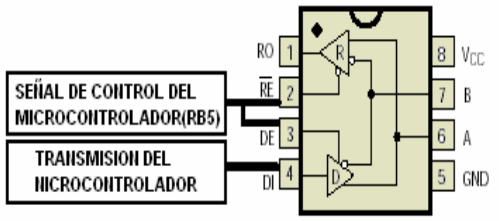

Figura 3.5 DIAGRAMA DE CONEXIÓN

Mientras se transmite la trama, desde el microcontrolador PIC TX *(ver Figura 3.2)* hacia los Dispositivos Esclavos, la señal de control colocará a los MAX485 asociados a la recepción en un estado de alta impedancia, en concordancia con el cuadro de la tabla 3.2.

La Figura 3.6 muestra la conexión de uno de los circuitos integrados MAX485 (ASCII o RTU) con los pines del microcontrolador PIC correspondientes a la recepción y señal de control.

SEÑAL DE CONTROL	ESTADO
0	HABILITA RECEPCION DEL MAX485
1	COLOCA EN ALTA IMPEDANCIA

Tabla 3.2 TABLA DE ESTADOS DE LOS MAX485

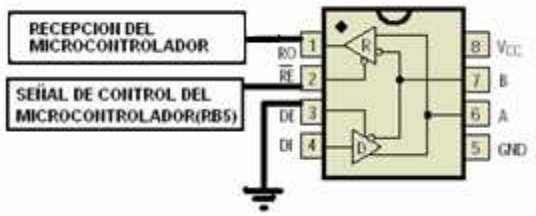

Figura 3.6 DIAGRAMA DE CONEXIÓN

Los dos microcontroladores PIC adicionales que conforman este dispositivo contienen el software MODBUS de recepción en sus dos modos (ASCII o RTU).

El usuario, mediante el “conmutador de modo”, puede escoger el modo de comunicación en el que se va a trabajar (*ver Figura 3.2*).

Además, el conmutador realiza la operación de unir el Pin de transmisión del microcontrolador PIC de recepción seleccionado con el Pin de recepción (TTL) del MAX232 para poder enviar la trama hacia el computador, tal como muestra la Figura 3.7.

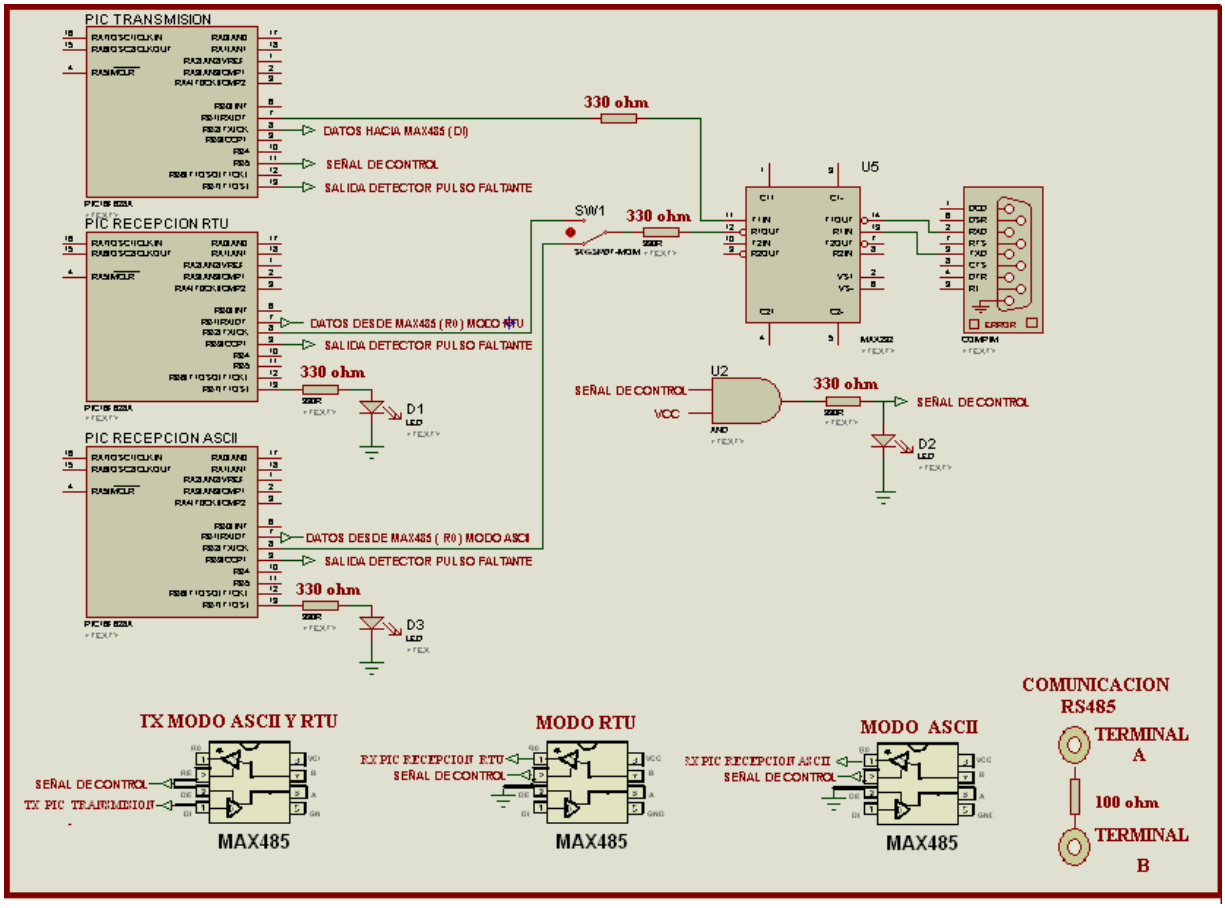

Figura 3.7 ESQUEMATICO GENERAL DE LA TARJETA MASTER

En la Figura 3.7 se muestra el esquemático de las conexiones de los elementos que se han utilizado en esta tarjeta.

Todas los circuitos integrados MAX485 tienen unidos sus pines A entre si, y sus pines B de igual manera. Adicionalmente, se ha colocado a los terminales de salida (A y B) una resistencia de 100 Ω, recomendada por el fabricante.

En la Figura 3.8 se muestra la tarjeta Master.

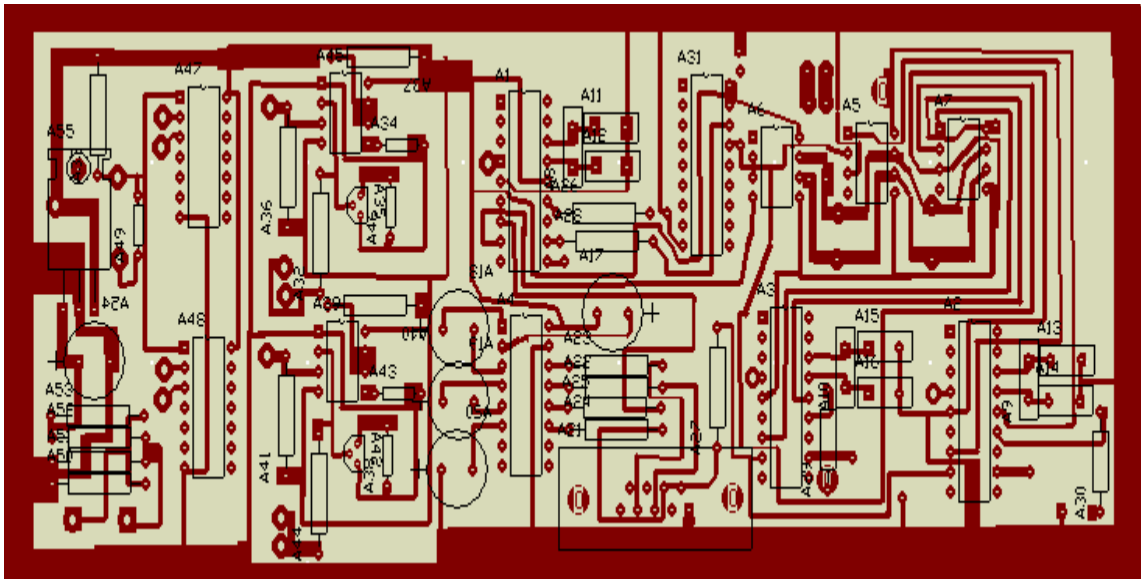

Figura 3.8 DIAGRAMA DE LA TARJETA MASTER

En la Figura 3.9 se muestra el Dispositivo Master construido.

Figura 3.9. FOTOGRAFÍA DEL DISPOSITIVO MASTER

3.1.2.- DISEÑO DEL HARDWARE REQUERIDO PARA LA TARJETA ESCLAVO

Al tratarse de un hardware con muchos componentes se construyó un dispositivo modular, tres tarjetas componen el Dispositivo Esclavo:

- Tarjeta Principal y de Comunicación
- Tarjeta de Visualización
- Tarjeta de Periféricos.

De igual manera, el usuario puede escoger el modo de comunicación (ASCII o RTU), mediante un conmutador de 2 posiciones (*conmutador de modo*), tal como se realiza en el Dispositivo Master. Los microcontroladores PIC son conectados a sus periféricos mediante dicho conmutador. Así, finalmente se tiene totalmente aislados los periféricos a un microcontrolador, mientras el otro se mantiene trabajando, tal como muestra la Figura 3.10.

Figura 3.10 CONMUTADORES DE MODO

Por otro lado se conmutan las líneas del módulo LCD para lograr el mismo efecto, mediante la habilitación de los integrados 74LS244 y 74LS245, cuyo funcionamiento se detallará más adelante.

3.1.2.1 DISEÑO DE LA TARJETA PRINCIPAL Y DE COMUNICACIÓN

La Tarjeta Principal es la base del Dispositivo Esclavo, en ella se encuentran los microcontroladores PIC que contienen el software desarrollado para el

Sistema MODBUS e incluye la parte correspondiente a la comunicación RS-485.

La Figura 3.11 muestra el esquemático general de las conexiones de un microcontrolador con sus periféricos, en un Dispositivo Esclavo. Cabe recordar que cada Dispositivo Esclavo se compone de dos microcontroladores PIC 16F877A, cada uno de ellos cargado con el software MODBUS en modo ASCII o RTU, respectivamente.

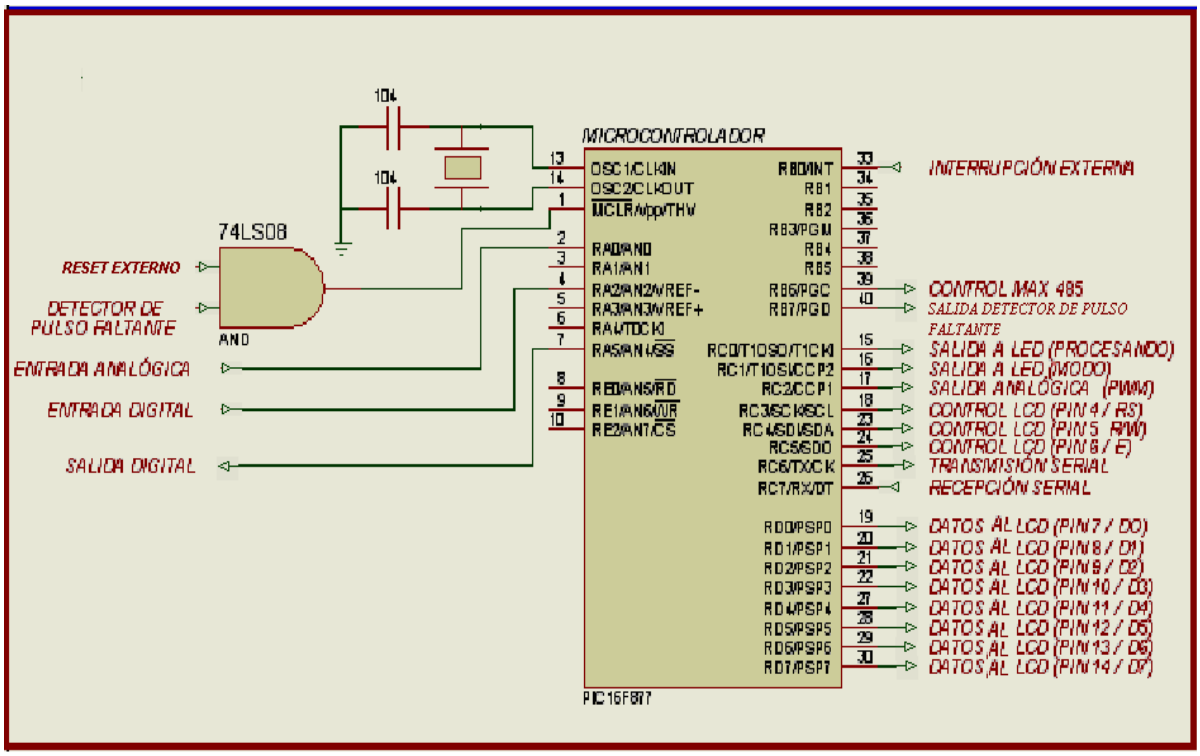

Figura 3.11 ESQUEMÁTICO GENERAL DE ENTRADAS Y SALIDAS DE UN MICROCONTROLADOR EN UN DISPOSITIVO ESCLAVO

En la Tarjeta Principal, se destaca:

- Circuitos de detector de pulso faltante (2)
- Microcontroladores PIC 16F877A (2)
- Interfaces MAX 485 (2)

Se ha escogido al microcontrolador PIC 16F877A como el dispositivo programable que realizará las diferentes actividades que se requieren (procesamiento de datos digitales, control y comunicación). Este microcontrolador es fabricado por MICROCHIP. El modelo 16F877A posee varias características que hacen a este microcontrolador un dispositivo muy versátil, eficiente y práctico para ser empleado en esta aplicación.
Algunas de las características importantes de este microcontrolador, para el presente proyecto, se muestran a continuación:

- Soporta modo de comunicación serial.
- Amplia memoria para datos y programa.
- Set de instrucciones reducido (tipo RISC), pero con las instrucciones necesarias para facilitar su manejo.
- Capacidad de Almacenamiento en su memoria EEPROM.

En la Tabla 3.3 se especifican las entradas y salidas del microcontrolador PIC, según la Figura 3.10:

NRO. DE PIN	***NOMBRE***	**FUNCIÓN**
1	*MCLR*	*RESET*
2	RA0	*ENTRADA ANÁLOGA*
3	RA1	
4	RA2	*ENTRADA DIGITAL*
5	RA3	
6	RA4	
7	RA5	*SALIDA DIGITAL*
8	REO	
9	RE1	
10	RE2	
11	Vdd	+5 VDC
12	Vss	GND
13	OSC1/CLK	
14	OSC1/CLK	

15	RCO	*SALIDA A LED (procesando)*
16	RC1	*SALIDA A LED (modo)*
17	RC2/CCP1	*SALIDA ANÁLOGA (PWM)*
18	RC3/SCK	*CONTROL LCD*
19	RD0/PSP0	*DATOS A LCD*
20	RD1/PSP1	*DATOS A LCD*
21	RD2/PSP2	*DATOS A LCD*
22	RD3/PSP3	*DATOS A LCD*
23	RC4/SD1	*CONTROL LCD*
24	RC5/SD0	*CONTROL LCD*
25	RC6/Tx	*TRANSMISION SERIAL*
26	RC7/RX	*RECEPCIÓN SERIAL*
27	RD4/PSP4	*DATOS A LCD*
28	RD5/PSP5	*DATOS A LCD*
29	RD6/PSP6	*DATOS A LCD*
30	RD7/PSP7	*DATOS A LCD*
31	Vss	*GND*
32	Vdd	*+5 VDC*
33	RBO/INT	*INTERRUPCIÓN EXTERNA (VISUALIZANDO)*
34	RB1	
35	RB2	
36	RB3	
37	RB4	
38	RB5	
39	RB6/PGC	*CONTROL MAX 485*
40	RB7/PGD	*DETECTOR DE PULSO FALTANTE*

Tabla 3.3 PINES DEL MICROCONTROLADOR PIC 16F877A

3.1.2.2 DISEÑO DEL CIRCUITO DE RESET

El circuito para reiniciar el sistema en condiciones iniciales consta de un reset externo asociado a un Circuito Detector de Pulso faltante mediante una compuerta lógica AND como muestra la Figura 3.12

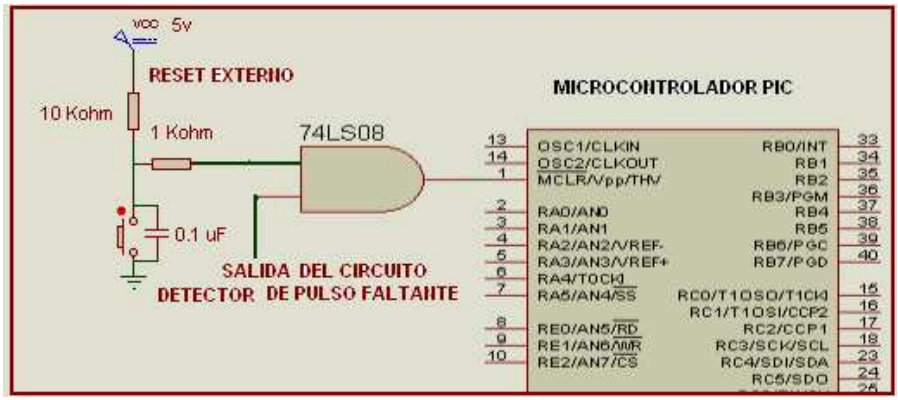

Figura 3.12 ESQUEMATICO CIRCUITO RESET

El funcionamiento del circuito de la Figura 3.12 se describe a continuación:

- Cuando el sistema funciona correctamente el Detector de Pulso Faltante envía siempre un 1 lógico.
- El reset externo está normalmente conectado a la alimentación VCC (+ 5vdc), es decir envía siempre un 1 lógico.

Cuando cualquiera de los dos (o los dos) envía un cero lógico a la salida de la compuerta AND se obtiene un cero lógico que hace al sistema reiniciarse. Esto corresponde a la tabla de verdad de la compuerta AND de 2 entradas, como indica la Tabla 3.4:

Pulsador	WDT	Salida
0	0	0
0	1	0
1	0	0
1	1	1

Tabla 3.4 TABLA DE VERDAD DE LA COMPUERTA AND

Se puede ver claramente que la salida solamente es "1" (1 lógico, nivel alto) cuando tanto la entrada "*Reset Externo*" como la entrada "*Detector de Pulso Faltante*" están en "1".

3.1.2.2.1 DISEÑO DEL RESET EXTERNO

El pulsador utilizado para reiniciar el sistema es conectado a una configuración típica de un filtro de rebote como la que se muestra en la Figura 3.13. Los filtros antirebote son utilizados generalmente para eliminar ruidos en las señales de entradas de los interruptores en circuitos electrónicos. Los valores fueron tomados de las sugerencias presentadas en el manual del fabricante.

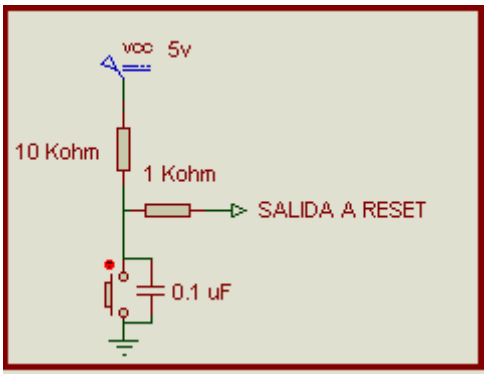

Figura 3.13 ESQUEMATICO CIRCUITO PULSADOR RESET

3.1.2.2.2 DISEÑO DEL DETECTOR DE PULSO FALTANTE

La principal función del Detector de Pulso Faltante es reiniciar al microcontrolador PIC después de que ocurra una falla o problema de software, en cuyo caso se reinicia el programa del microcontrolador en ejecución.

El circuito de la Figura 3.14 es disparado continuamente por pulsos entrantes desde la salida "*Detector de Pulso Faltante*" del microcontrolador. Cuando esto ocurre, el capacitor se mantendrá en un estado de carga y descarga, generando una salida equivalente a 1 lógico. En un tiempo mayor a t=Ra*C,

en el que no llegasen pulsos a la entrada, el capacitor se cargará completamente, haciendo que la salida cambie a 0 lógico, lo que producirá que el microcontrolador se reinicie.

T = Ra * C

T = 1 Mohm * 4.7 uF

T = 4.7 Segundos

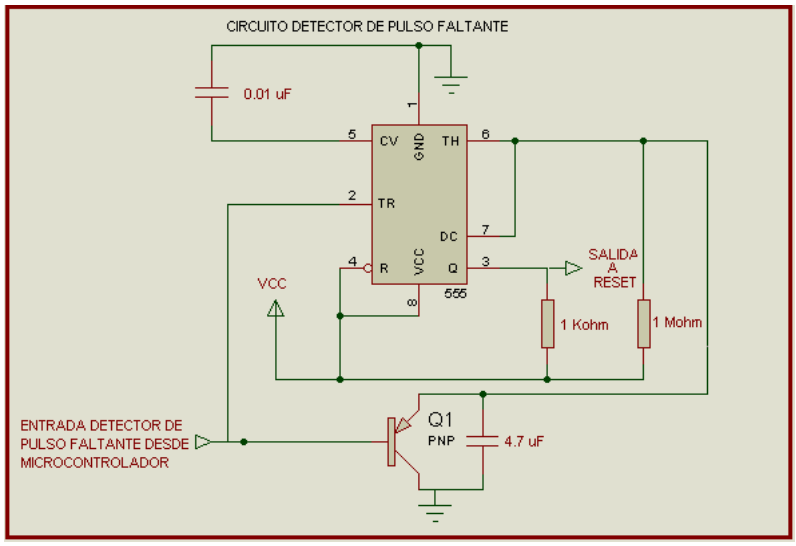

Figura 3.14 ESQUEMATICO CIRCUITO DETECTOR DE PULSO

3.1.2.3 SALIDAS A LED

El criterio para obtener una buena intensidad luminosa es escoger la corriente que atraviesa el LED; el voltaje de operación en el caso presente es de 5 voltios aproximadamente, y con una resistencia de 330 ohmios, se obtiene una corriente de 15,15 mA, así:

I = V / R

I = 5v / 330 ohm

I = 15,15 mA

La forma de conexión de los LEDS, en el proyecto se muestra en la Figura 3.15.

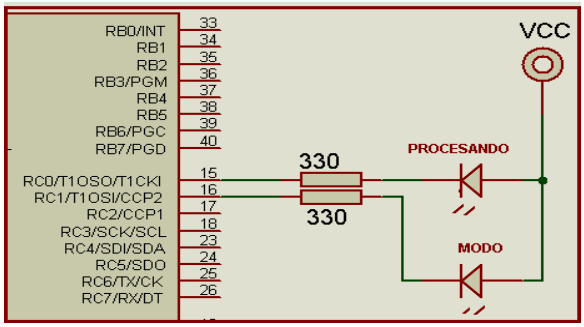

Figura 3.15 CONEXIÓN FÍSICA DE LOS LEDS

Tanto para RC1 como para RC0 (ver Figura 3.15), el microcontrolador envía normalmente un 0L (0 Vdc) con lo cual se tiene polarización positiva en el ánodo y negativa en el cátodo, entonces el Led permanece encendido.

3.1.2.3.1 SALIDA A LED (modo)

Este Led es utilizado para visualizar el modo en el que está trabajando el Dispositivo Esclavo (ASCII o RTU), como muestra la Figura 3.16.

Figura 3.16 LEDS DE MODO(modo)

3.1.2.3.2 SALIDA A LED (procesando)

Este Led indica que el Dispositivo Esclavo ha recibido una trama del Dispositivo Master. Comúnmente el LED está encendido. Mientras esté titilando indica que el Dispositivo Esclavo está procesando la trama recibida, como se muestra en la Figura 3.17.

Figura 3.17 LEDS DE MODO(procesando)

3.1.2.4 DISEÑO DE LA INTERRUPCIÓN EXTERNA

La conexión física de la interrupción externa se muestra en la Figura 3.18. La resistencia R1 (330 ohm) se escogió para que ingrese al microcontrolador una corriente de 15,15 mA, con un voltaje de 5 voltios. La resistencia R2 se escogió de un valor menor a 40K para limitar la corriente.

Figura 3.18 CONEXIÓN DE LA INTERRUPCIÓN INTERNA

Conforme se presione el pulsador “VISUALIZACIÓN”, cambiará la información en el LCD. Secuencialmente la información aparecerá de la siguiente manera:

- CONFIGURACIÓN (ASCII o RTU)
- ESTADO DEL PERIFÉRICO: SALIDA DIGITAL
- ESTADO DEL PERIFÉRICO: SALIDA ANÁLOGA
- ESTADO DEL PERIFÉRICO: ENTRADA DIGITAL
- ESTADO DEL PERIFÉRICO: ENTRADA ANÁLOGA

3.1.2.5 COMUNICACIÓN RS-485

A nivel de capa física la comunicación entre tarjetas utiliza la norma RS-485. Para convertir las señales de los niveles TTL a RS-485 y viceversa se utiliza un integrado MAX 485.

El microcontrolador PIC, mediante el Pin "*Control Max 485*", habilitará la recepción o transmisión de este integrado; tal como muestra la Figura 3.19. Se puede observar la utilización de una compuerta AND como buffer para mejorar la señal de control y activar al Led de transmisión.

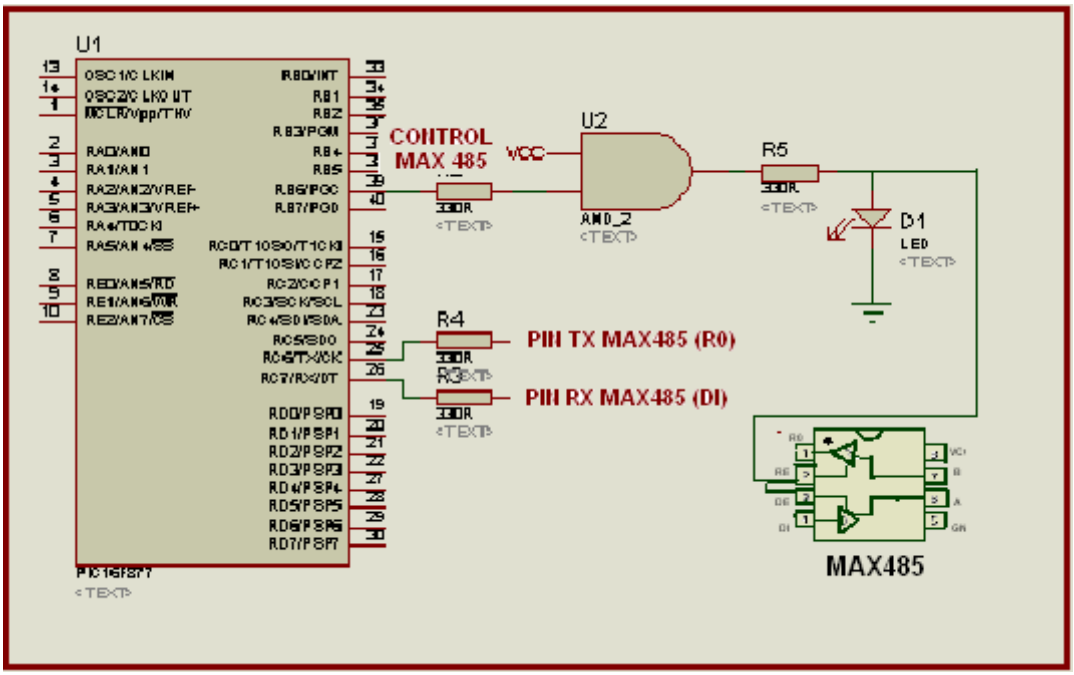

Figura 3.19 CIRCUITO PARA LA COMUNICACIÓN RS-485

En la Figura 3.20 se muestra el ruteado de la tarjeta principal y de comunicación.

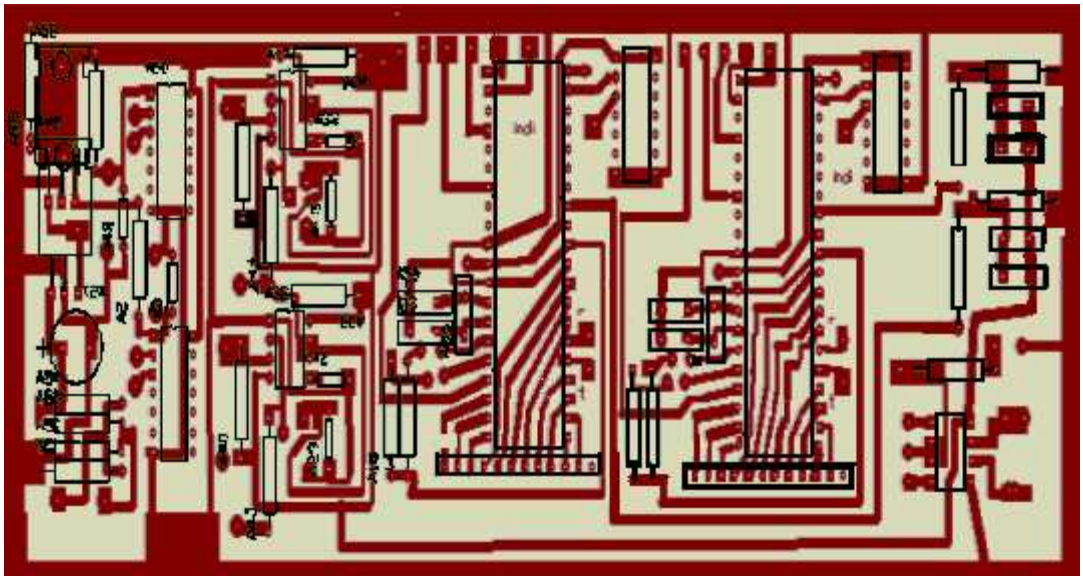

Figura 3.20 DIAGRAMA DE LA TARJETA PRINCIPAL Y DE COMUNICACIÓN

3.1.3 DISEÑO DE LA TARJETA DE VISUALIZACIÓN (LCD)

La conexión entre un módulo LCD y un microcontrolador PIC16F877A debe realizarse como se ilustra en la Figura 3.21.

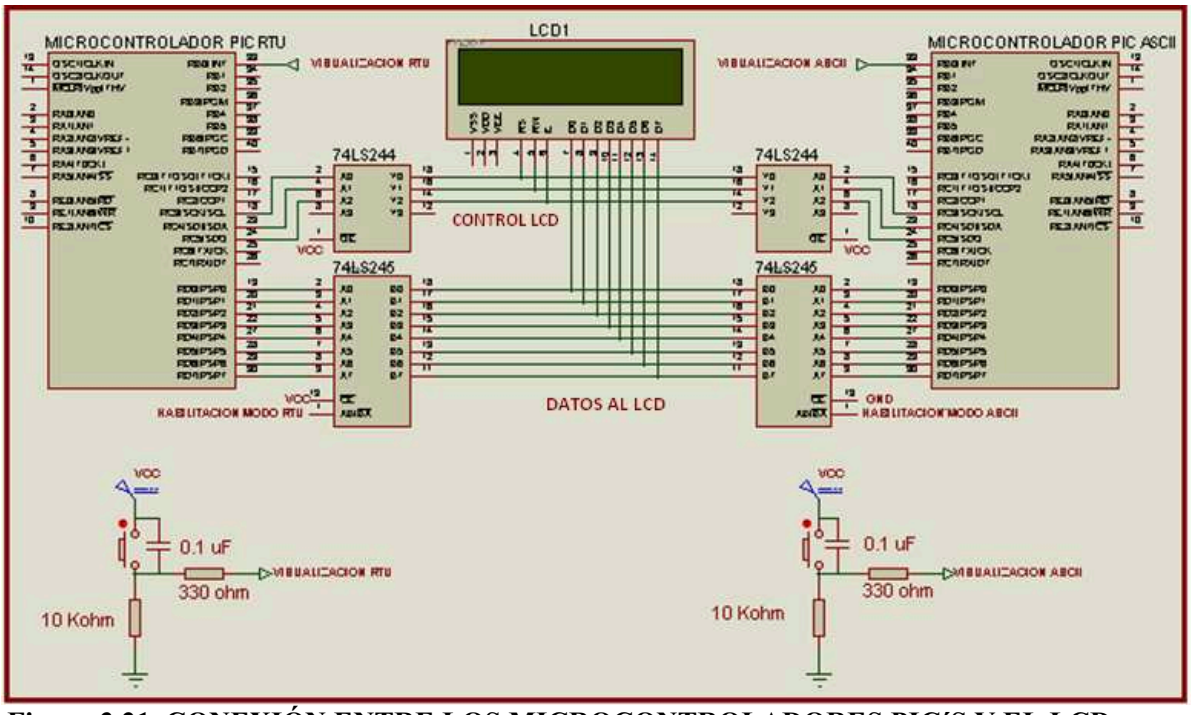

Figura 3.21 CONEXIÓN ENTRE LOS MICROCONTROLADORES PIC´S Y EL LCD

Como se muestra en la Figura 3.21, las líneas "*Datos al LCD*" (de cada microcontrolador PIC) se conecta a los pines de entrada del buffer de 3 estados ***74LS245***; mientras que las líneas ***"Control LCD"*** (de cada microcontrolador PIC) se conectan al integrado ***74LS244***, igualmente buffer de 3 estados.

La habilitación de los integrados *74LS244 y 74LS245,* asociados a cada microcontrolador PIC, determina cual de ellos enviará los datos al LCD; mientras el otro se mantiene totalmente separado de este bus.

La distribución de pines y la tabla de habilitación de estos integrados se puede ver en los Anexos de este trabajo.

3.1.4 DISEÑO DE LA TARJETA DE PERIFÉRICOS (SENSORES)

La Tarjeta de Periféricos diseñada incluye:

***Periféricos de Entrada**:* Son los que introducen datos externos al Dispositivo Esclavo para su posterior tratamiento por parte del microcontrolador PIC. Los periféricos de entrada implementados son:

- 1 Entrada Digital *(interna o externa)*
- 1 Entrada Análoga *(interna o externa)*

***Periféricos de salida**:* Son los que reciben información que es procesada por el microcontrolador PIC y la reproducen para que sea perceptible para el usuario. Los periféricos de salida implementados son:

- 1 Salida Digital
- 1 Salida Análoga

3.1.4.1 DISEÑO DE LA ENTRADA ANÁLOGA

La entrada análoga se ha diseñado de tal manera que el usuario pueda escoger, mediante un selector, entre una entrada interna *(sensor de temperatura internamente acondicionado)* y una entrada externa, tal como muestra la Figura 3.22.

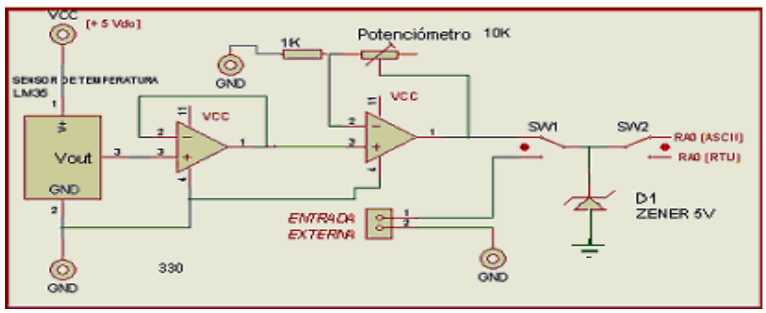

Figura 3.22 SELECTOR DE ENTRADA ANÁLOGA

En la Figura 3.22 se muestra la manera como se conecta el periférico "*entrada análoga*". El circuito consta de un sensor de temperatura LM35, acondicionado, y una bornera para conectar una entrada externa. Mediante el conmutador SW1 se puede seleccionar entre estas opciones para su posterior conexión hacia el microcontrolador PIC.

Como se detalló en la TABLA 3.3, el pin RA0 del microcontrolador PIC servirá para la entrada análoga.

Para la entrada análoga interna, se emplea un sensor se temperatura LM35, internamente acondicionado, mediante un amplificador operacional LM324.
En el diseño se ha implementado un Amplificador en dos etapas.
La *primera etapa* consiste en un *seguidor de voltaje*, como se muestra en la Figura 3.23, para tener a la salida la misma tensión de entrada (Vin=Vout).

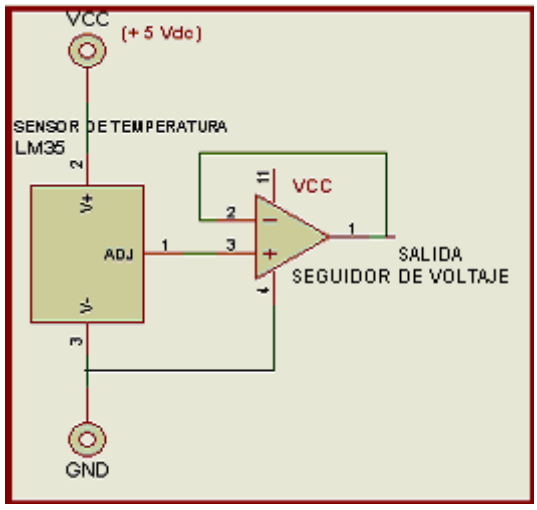

Figura 3.23 CIRCUITO SEGUIDOR DE VOLTAJE

La *segunda etapa* consiste en un Amplificador No Inversor. La ganancia de este circuito viene dada por Av = 1 + R1 / R2.tal como muestra la Figura 3.24.

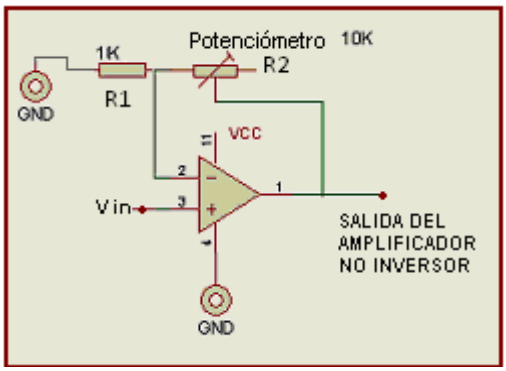

Figura 3.24 CIRCUITO AMPLIFICADOR NO INVERSOR

El valor de R2 se ajusta mediante el potenciómetro de precisión, hasta conseguir el valor deseado para la calibración del sensor de temperatura.

Se ha incluido un zener a la salida del operacional para proteger al microcontrolador de señales mayores a 5.1V, como se puede ver en la Figura 3.22.

3.1.4.2 DISEÑO DE LA ENTRADA DIGITAL

Al igual que en la entrada análoga, en este caso se ha implementado también un selector, el cual permite escoger entre una entrada interna y otra externa.

La entrada digital se ha diseñado de tal manera que el usuario pueda escoger, mediante un selector, entre una entrada interna *(interruptor)* y una entrada externa, tal como muestra la Figura 3.25.

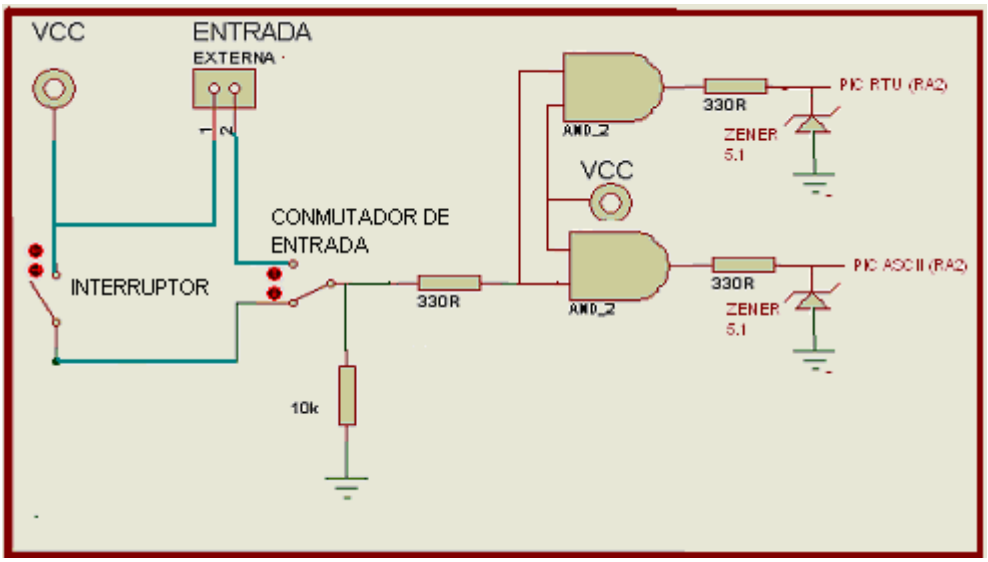

Figura 3.25 CIRCUITO DE CONMUTACIÓN DE ENTRADAS

Se ha incluido un zener a la salida de la compuerta AND para proteger al microcontrolador de señales mayores a 5.1V.

Para este caso la señal de entrada será enviada a los dos microcontroladores PIC, sin importar el modo en que el Dispositivo Esclavo se encuentre funcionando.

Para el diseño se tomó una corriente de 15,15 mA, por recomendaciones del fabricante, a las entradas de los microcontroladores PIC, por lo que se utilizaron resistencias de 330 ohm.

V= R*I

R= V / I

R = 5 / 15,15 mA

R= 330 ohm

3.1.4.3 DISEÑO DE LA SALIDA ANÁLOGA (PWM)

La señal PWM del microcontrolador PIC comandará el circuito de potencia de la salida análoga.

El circuito de potencia consiste en un convertidor DC-DC, un circuito que transforma corriente continua de una tensión a otra. Este es un circuito eficiente para controlar la velocidad en motores DC. El circuito descrito permite controlar motores que manejen un amperaje bajo u otros dispositivos con características similares.

El circuito de la Figura 3.26 se basa en un transistor TIP 110, el cual recibe la señal generada por el microcontrolador, un tren de pulsos, y a su vez este acciona por pulsos el motor de continua. El diodo en paralelo con el motor impide que, cuando se quita la corriente, el transistor se queme. El voltaje máximo que puede entregar este es el de la fuente conectada; es decir 9 VDC.

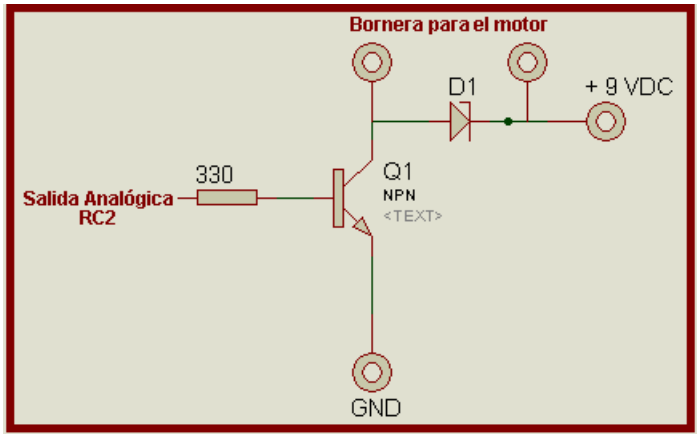

Figura 3.26 CIRCUITO SALIDA ANÁLOGA

Para el diseño se tomó una corriente de 15,15 mA de salida del microcontrolador PIC, por lo que se utilizo una resistencia de 330 ohm.

V= R*I

R= V / I

R = 5 / 15,15 mA

R= 330 ohm

3.1.4.4 SALIDA DIGITAL

La salida digital se utiliza para controlar otros dispositivos electrónicos a través de un relé. Por ejemplo para activar o desactivar un dispositivo o actuador digital de salida, como puede ser un foco.

En la Figura 3.27 se muestra el circuito diseñado para la salida digital. Una señal ON/OFF que entrega el Pin "*Salida Digital*" del microcontrolador activa o desactiva, a través de la base el transistor, que a su vez conecta o desconecta la alimentación a la bobina del relé.

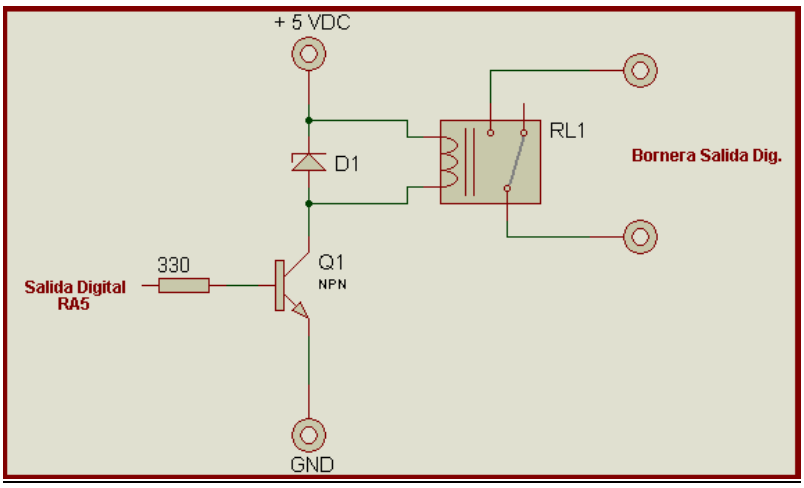

Figura 3.27 CIRCUITO SALIDA DIGITAL

Para el diseño se tomó una corriente de 15,15 mA de salida del microcontrolador PIC, por lo que se utilizo una resistencia de 330 ohm.

V= R*I

R= V / I

R = 5 / 15,15 mA

R= 330 ohm

La gran ventaja de los relés es la completa separación eléctrica entre la corriente de accionamiento (la que circula por la bobina del electroimán) y los circuitos controlados por los contactos, lo que hace que se puedan manejar altos voltajes o elevadas potencias con pequeñas tensiones de control. Además la posibilidad de control de un dispositivo a distancia mediante el uso de pequeñas señales de control.

En la Figura 3.28 se muestra el ruteado de la tarjeta principal y de comunicación, mientras que en la Figura 3.29 se muestra el Dispositivo construido.

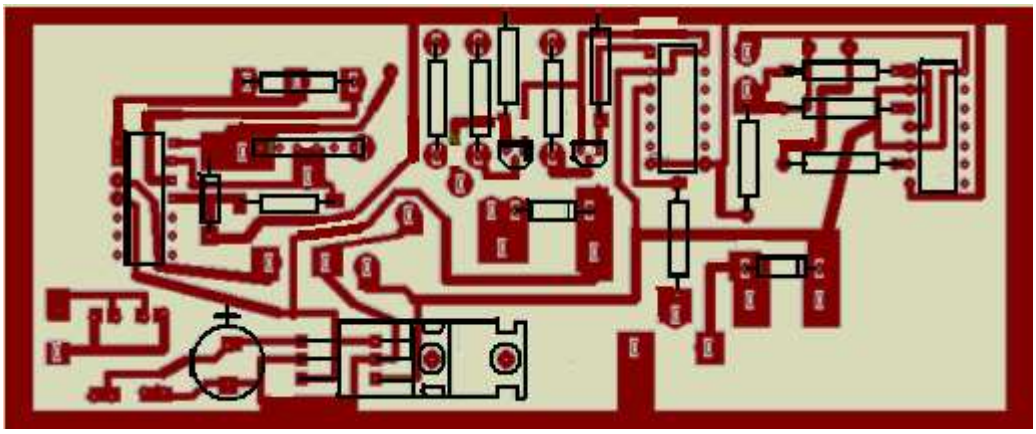

Figura 3.28 DIAGRAMA DE LA TARJETA DE PERIFÉRICOS

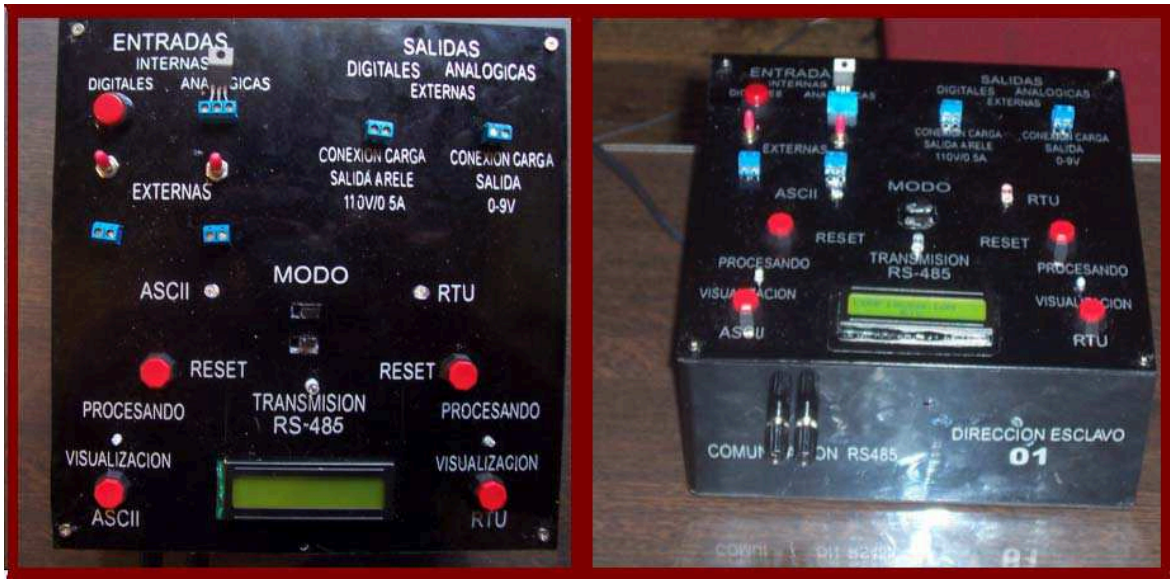

Figura 3.29 FOTOGRAFÍA DISPOSITIVO ESCLAVO

En el capítulo siguiente se realizan las Pruebas y Resultados del sistema implementado.

CAPÍTULO 4

PRUEBAS Y RESULTADOS

CAPÍTULO 4

PRUEBAS Y RESULTADOS

En este capítulo se describen las pruebas realizadas para la comprobación del correcto funcionamiento del sistema, una vez concluida su construcción. De igual manera, se analizarán los resultados obtenidos en dichas pruebas.

4.1 SOFTWARE UTILIZADO PARA PRUEBAS DE COMUNICACIÓN

Para las pruebas se utilizó el software ***MODSCAN32*** elaborado por Win-Tech, cuyo propósito fundamental es el de simular un Dispositivo Master.

Este software envía Tramas MODBUS (ASCII o RTU) y valida las tramas de respuesta si estas cumplen con el protocolo MODBUS. Además permite visualizar dichas tramas.

Las tramas MODBUS de respuesta pueden ser visualizadas en formatos tales como: decimal, hexadecimal, binario y notación de punto flotante, tanto en modo RTU como en modo ASCII.

Para las pruebas, en primer lugar se configuraron los parámetros relacionados a la comunicación serial, como el puerto, la velocidad de transmisión, la paridad y los bits de parada. En la Figura 4.1 se muestra la ventana de “detalles de conexión” para la configuración de los parámetros mencionados.

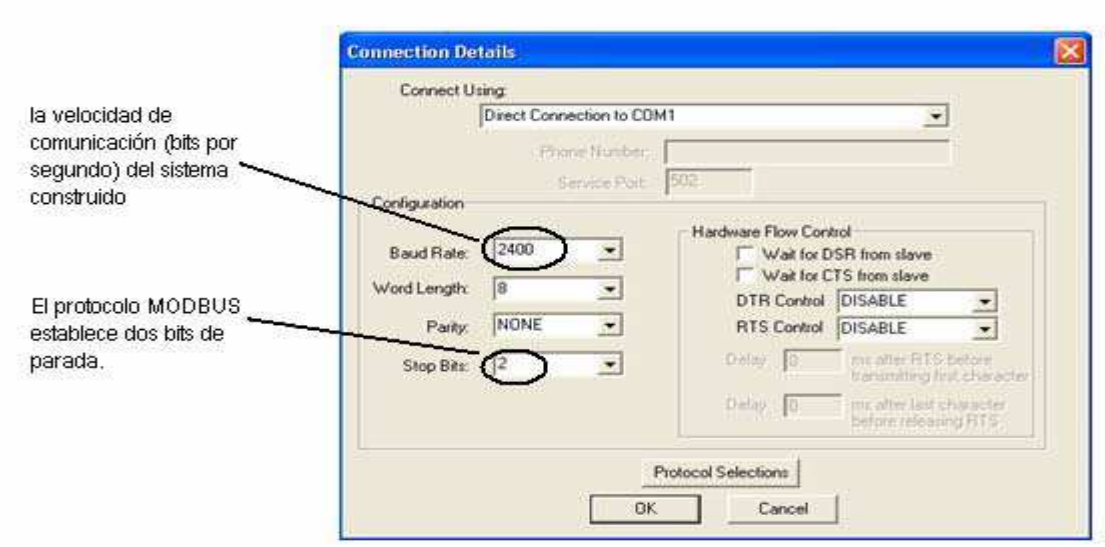

Figura 4.1 VENTANA PARA CONFIGURAR LOS PARÁMETROS DE LA COMUNICACIÓN SERIAL

Se debe especificar el modo de Transmisión (ASCII o RTU). También debe configurarse el tiempo entre trama y trama (*delay between polls*) para comunicación continua, y el tiempo de respuesta para validar la trama (*Slave response timeout*).

En la Figura 4.2 se muestra la ventana selección del protocolo MODBUS para la configuración de los parámetros mencionados.

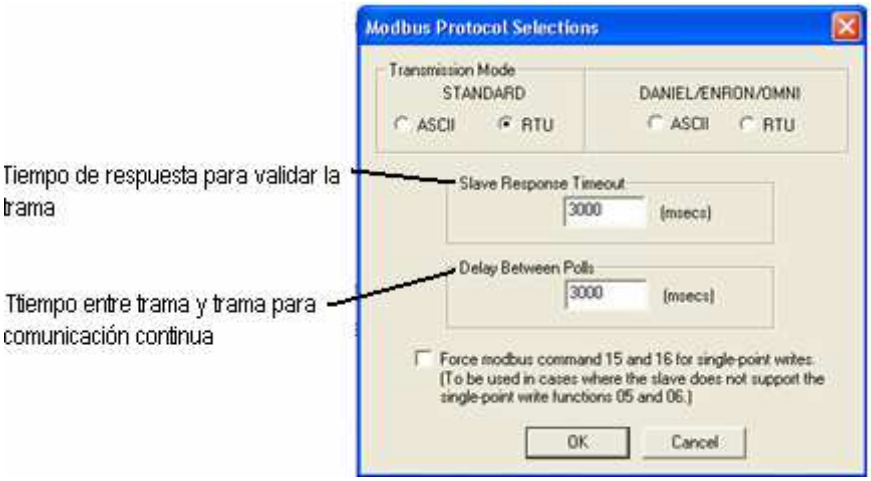

Figura 4.2 VENTANA PARA CONFIGURAR LOS MODOS DE TRANSMISIÓN Y LOS TIEMPOS RELACIONADOS

En la ventana principal se puede construir la trama MODBUS de petición que será enviada hacia el Dispositivo Esclavo. En esta ventana se puede configurar

la dirección del Dispositivo Esclavo, función, dirección de los registros y el número de registros. En esta ventana también se puede visualizar el número de tramas enviadas y el número de respuestas validas, tal como muestra la Figura 4.3.

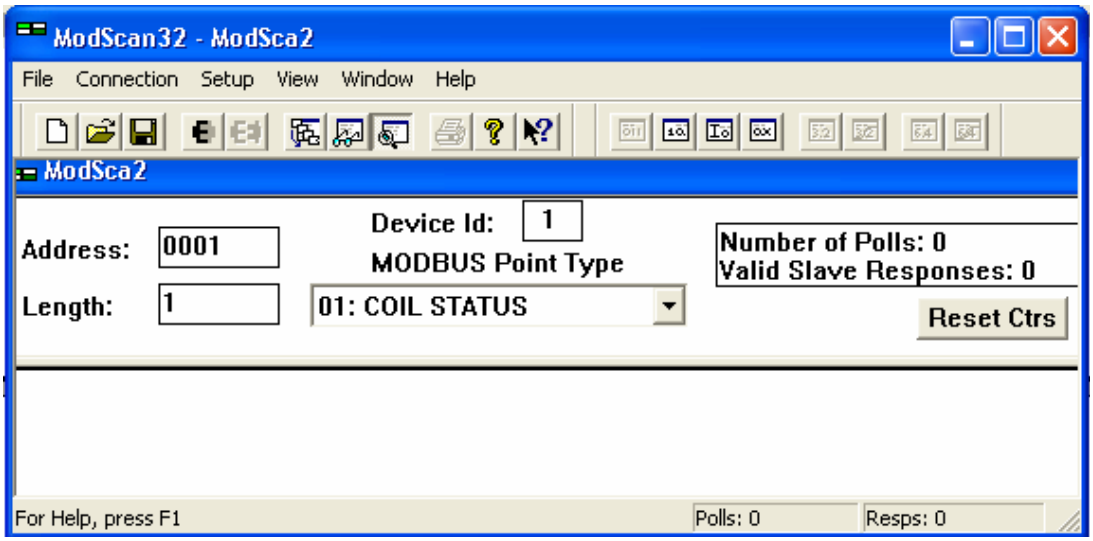

Figura 4.3 VENTANA PRINCIPAL

El software ***MODSCAN32*** permite forzar salidas digitales (*Función 15*), programar registros de memoria (*Función 16*), así como también forzar una salida digital (*Función 05*) y programar un registro de memoria (*Función 06*).

Una de las aplicaciones que se utilizó en el presente proyecto es la de forzar una salida digital (*Función 05*). A manera de ejemplo, el procedimiento que se sigue es el siguiente:

En la ventana de visualización de datos, se selecciona la Función 01 (*leer estado de salidas digitales*) y se ingresa la salida digital que se desea forzar. En el menú que aparece a continuación se procede a configurar la dirección del Dispositivo Esclavo, la dirección de la salida que se quiere forzar y el valor de la salida digital. En la Figura 4.4 se muestra la ventana para la configuración de la Función 05.

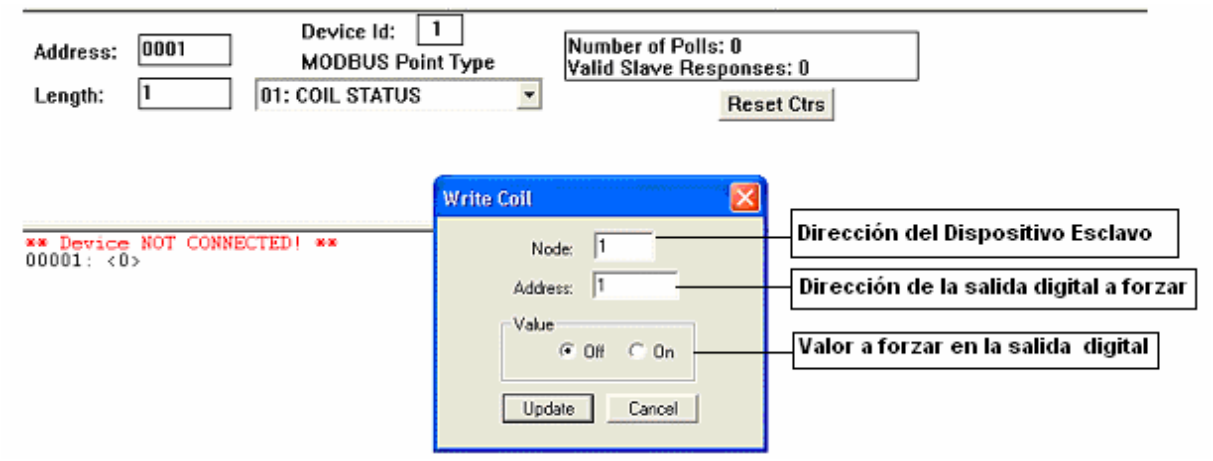

Figura 4.4 VENTANA PARA CONFIGURAR LA FUNCIÓN 05 (FORZAR SALIDA DIGITAL)

Otra Función que se utilizó en el presente proyecto es la de programar un registro de memoria (*Función 06*), la cual se realizó de la siguiente manera:

En la ventana de visualización de datos, nos ubicamos en la Función 03 (*leer registros de memoria*) e ingresamos el registro a programar. En el menú que aparece a continuación, se procede a configurar la dirección del Dispositivo Esclavo, la dirección del registro de memoria a programar y el valor que se desee programar en el registro de memoria. En la figura 4.5, se muestra la ventana para la configuración de la Función 06.

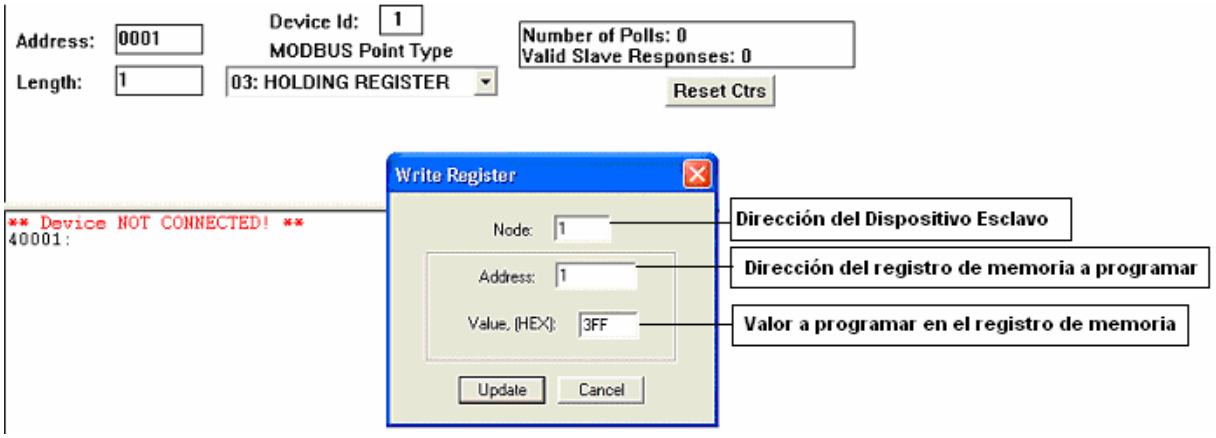

Figura 4.5 VENTANA PARA CONFIGURAR LA FUNCIÓN 06 (PROGRAMAR UN REGISTRO DE MEMORIA)

4.2 PRUEBAS DE COMUNICACIÓN ENTRE EL DISPOSITIVO MASTER Y LOS DISPOSITIVOS ESCLAVOS.

El propósito fundamental de este proyecto es la correcta elaboración y transmisión de las tramas MODBUS de petición en los dos modos (ASCII o RTU), desde la HMI del computador hasta los Dispositivos Esclavos, pasando por el Dispositivo Master. Una vez procesada la información por los Dispositivos Esclavos, estos elaboran la trama MODBUS de respuesta para enviarla hacia el HMI del computador, pasando por el Dispositivo Master.

Generando la trama MODBUS con el software ***MODSCAN32,*** el Dispositivo Master envía esta trama hacia los Dispositivos Esclavos. Obviamente, todos los Dispositivos Esclavos deben estar conectados y seleccionados en un mismo modo de comunicación. Para comprobar que no existan fallas de comunicación, los Esclavos deben enviar la trama MODBUS de respuesta hacia el Dispositivo Master y este enviarla nuevamente hacia el computador para ser validada.

En el Dispositivo Master para dar a conocer que la comunicación se está realizando se tiene un diodo led, el cual indica cuando una trama se está dirigiendo hacia los Dispositivos Esclavos. De la misma manera, existe un diodo led en los Dispositivos Esclavos que indicará cuando la trama MODBUS de petición ha sido validada por el Dispositivo Esclavo y está siendo enviada hacia el Dispositivo Master.

4.2.1. RESULTADOS OBTENIDOS DE LA PRUEBA DE COMUNICACIÓN EN MODO ASCII

En esta sección se indicarán las tramas de petición y de respuesta de todas las funciones con las que se trabajó en modo ASCII.

4.2.1.1 RESULTADO DE LA PRUEBA DE COMUNICACION UTILIZANDO LA FUNCIÓN 01 EN MODO ASCII

Se realizó la prueba de comunicación para la Función 01 en modo ASCII, enviando una trama MODBUS de petición con la dirección y el número de la salida digital a leer, para que el Dispositivo Esclavo, al cual esté dirigida la trama, construya una trama MODBUS de respuesta con el estado actual de la salida digital y envié la trama MODBUS hacia el Dispositivo Master. Esta prueba se realizó con los tres Dispositivos Esclavos.

En la Figura 4.6 se muestran los resultados de la prueba mencionada: la trama MODBUS de petición en color blanco y la trama MODBUS de respuesta en color negro.

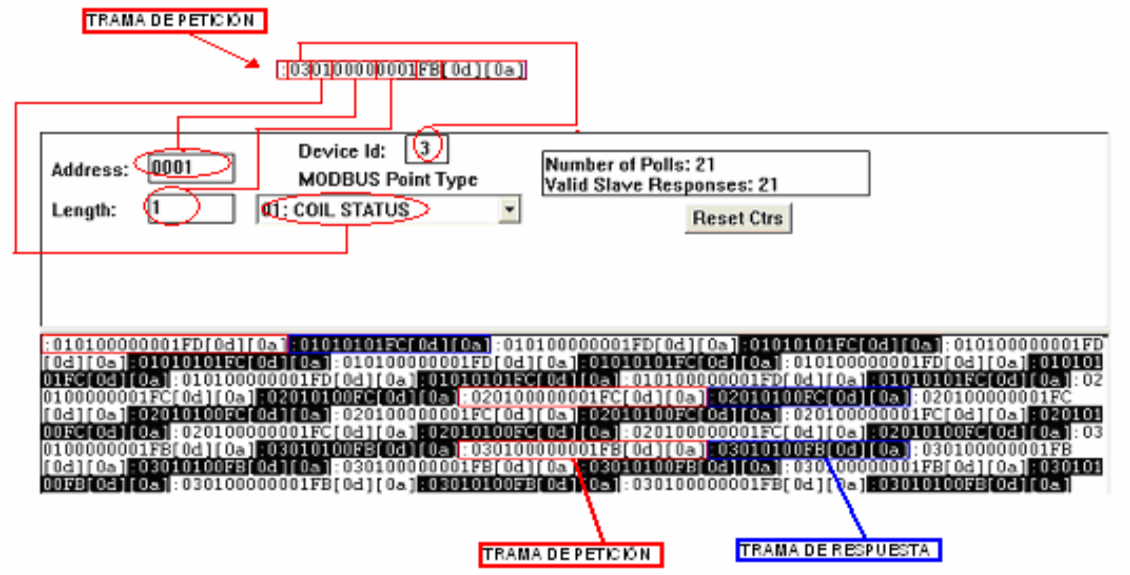

Figura 4.6 VISUALIZACIÓN DE TRÁFICO DE DATOS VÁLIDOS DE LA FUNCIÓN 01

A continuación se explica el significado de los datos contenidos en dichas tramas para su mejor comprensión:

- ✓ ***Trama de Petición (enviada por el Dispositivo Máster):***

:	Carácter de Inicio
03	Dirección del Esclavo (*varía según el esclavo*)
01	Función 01 de Lectura de Salidas Digitales

00	Dirección Alta de las Salidas Digitales a leer
00	Dirección Baja de las Salidas Digitales a leer
00	Registro Alto del número de Salidas Digitales a leer
01	Registro Bajo del número de Salidas Digitales a leer
FB	LRC de la trama de petición

- ✓ ***Trama de Respuesta (enviada por el Dispositivo Esclavo):***

:	Carácter de Inicio
03	Dirección del esclavo (*varía según el esclavo*)
01	Función 01 de Lectura de Salidas Digitales
01	Número de bytes de estado de las Salidas Digitales
00	Estado de la Salida Digital
FB	LRC de la trama de respuesta

De igual manera, se efectuó la prueba enviando una trama MODBUS de petición en modo ASCII con una dirección de la salida digital a leer no válida; con lo cual el Dispositivo Esclavo construirá un mensaje de excepción con un código de dirección ilegal y la enviará hacia el Dispositivo Master. Esta prueba se la realizó en los 3 Dispositivos Esclavos.

En la Figura 4.7a se muestran los resultados de la prueba realizada en la ventana de visualización de tráfico y en la Figura 4.7b se muestran los resultados de la misma en la ventana de visualización de datos.

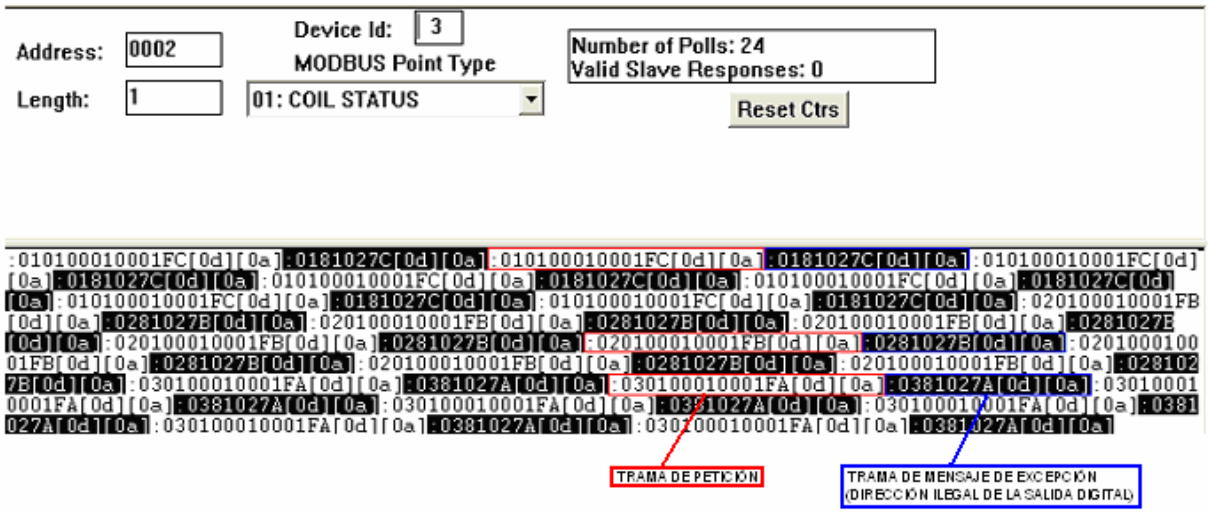

Figura 4.7 a) VISUALIZACIÓN DE TRÁFICO DE RESPUESTAS DE EXCEPCIÓN (ERROR EN DIRECCIÓN)

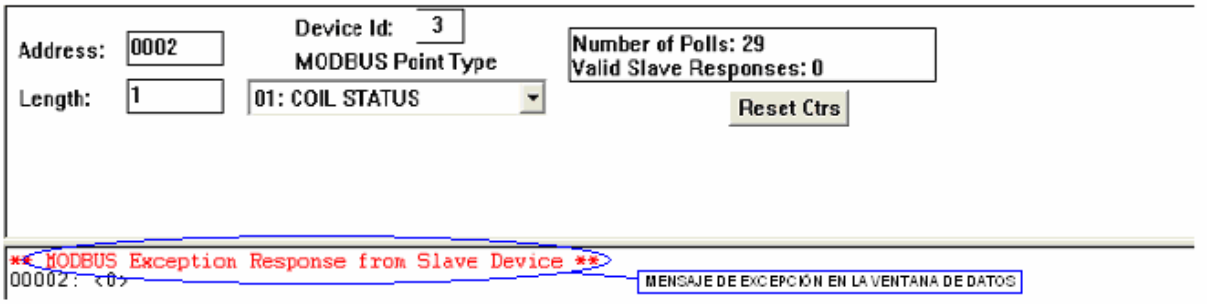

Figura 4.7 b) VISUALIZACIÓN DE DATOS DE RESPUESTAS DE EXCEPCIÓN (ERROR EN DIRECCIÓN)

A continuación se explica el significado de los datos contenidos en dichas tramas para su mejor comprensión:

- ✓ ***Trama de Petición (enviada por el Dispositivo Máster):***

:	Carácter de Inicio
03	Dirección del Esclavo
01	Función 01 de Lectura de Salidas Digitales
00	Dirección Alta de las Salidas Digitales a leer
01	Dirección Baja de las Salidas Digitales a leer (***dirección incorrecta***)
00	Registro Alto del número de Salidas Digitales a leer
01	Registro Bajo del número de Salidas Digitales a leer

FA	LRC de la trama de petición

✓ ***Mensaje de excepción (enviada por el Dispositivo Esclavo):***

:	Carácter de Inicio
03	Dirección del esclavo
81	Función *(función de la trama de petición + 80Hexadecimal)*
02	Código de excepción
7A	LRC del mensaje de excepción

Se realizó también la prueba enviando una trama MODBUS de petición en modo ASCII con el número de la salida digital a leer equivocada, con lo cual el Dispositivo Esclavo construirá un mensaje de excepción con un código de valor ilegal y la enviará hacia el Dispositivo Master. Esta prueba se la realizó con los 3 Dispositivos Esclavos.

En la Figura 4.8 se muestran los resultados.

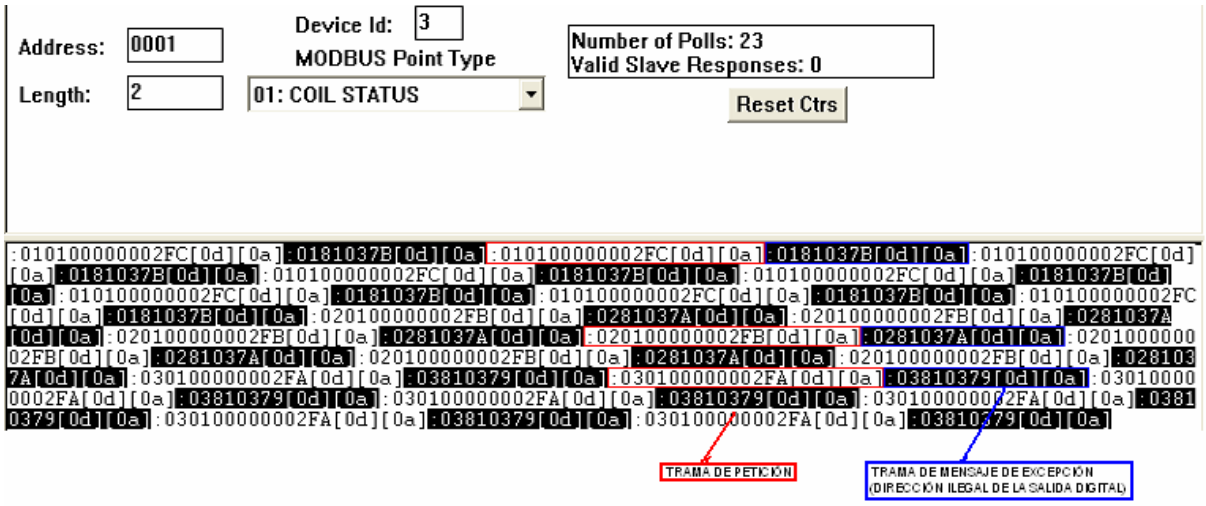

Figura 4.8 a) VISUALIZACIÓN DE TRÁFICO DE RESPUESTAS DE EXCEPCIÓN (ERROR EN DATOS)

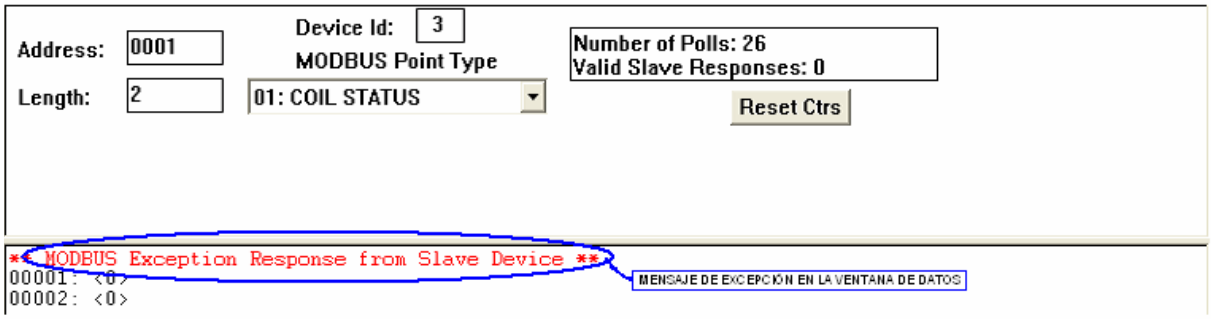

Figura 4.8 b) VISUALIZACIÓN DE DATOS DE RESPUESTAS DE EXCEPCIÓN (ERROR EN DATOS)

A continuación se explica el significado de los datos contenidos en dichas tramas para su mejor comprensión:

- ✓ ***Trama de Petición (enviada por el Dispositivo Máster):***

:	Carácter de Inicio
03	Dirección del Esclavo (*varía según el esclavo*)
01	Función 01 de Lectura de Salidas Digitales
00	Dirección Alta de las Salidas Digitales a leer
00	Dirección Baja de las Salidas Digitales a leer
00	Registro Alto del número de Salidas Digitales a leer
02	Registro Bajo del número de Salidas Digitales a leer ***(número de registro incorrecto)***
FA	LRC de la trama de petición

- ✓ ***Mensaje de excepción (enviada por el Dispositivo Esclavo):***

:	Carácter de Inicio
03	Dirección del esclavo (*varía según el esclavo*)
81	Función *(función de la trama de petición + 80Hexadecimal)*
03	Código de excepción
79	LRC del mensaje de excepción

De igual manera se realizaron las pruebas para las funciones 02, 03 y 04 en modo ASCII, enviando una trama MODBUS de petición para que el Dispositivo esclavo construya una trama MODBUS de respuesta con el estado actual del periférico de entrada o salida. Los resultados de estas pruebas se pueden ver en los Anexos de este trabajo.

Estas pruebas se realizaron con los tres Dispositivos Esclavos.

4.2.1.2.- RESULTADO DE LA PRUEBA DE COMUNICACION UTILIZANDO LA FUNCIÓN 05 EN MODO ASCII

A diferencia de las funciones de la 01 a la 04, las funciones 05 y 06 soportan modo BROADCAST.

Para la prueba de la Función 05, el software MODSCAN32 utiliza la ventana de datos en la Función 01 *(ver Figura 4.4).* Cabe recalcar que para forzar una salida digital este software primero mostrará las tramas de petición y respuesta de la Función 01 y luego las tramas de petición y respuesta de la Función 05 tal como muestra la Figura 4.9.

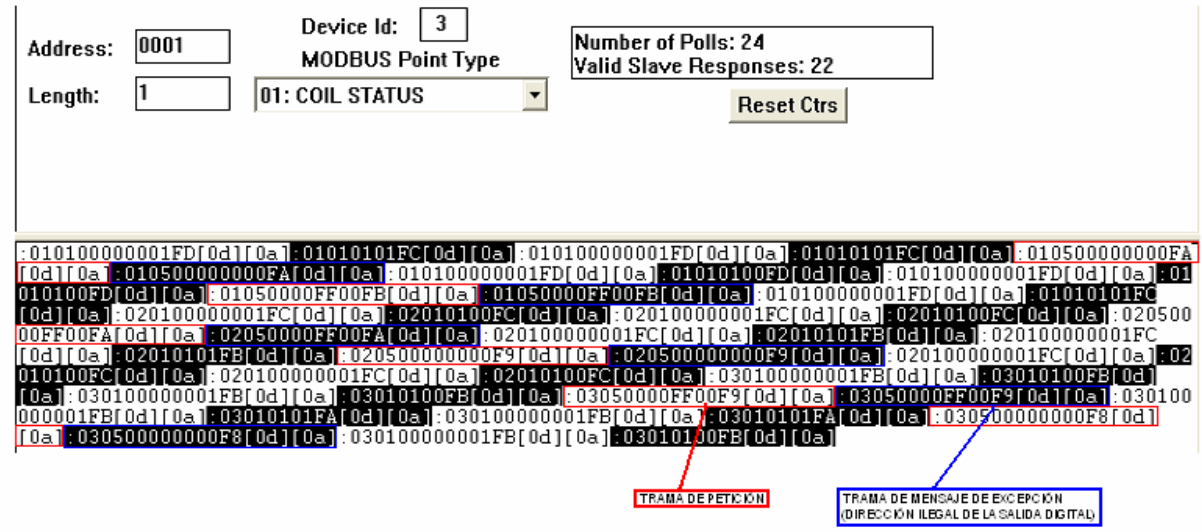

Figura 4.9 VISUALIZACIÓN DE TRÁFICO DE DATOS VÁLIDOS DE LA FUNCIÓN 05

A continuación se explica el significado de los datos contenidos en dichas tramas para su mejor comprensión:

- ✓ ***Trama de Petición (enviada por el Dispositivo Máster):***

:	Carácter de Inicio
03	Dirección del esclavo *(varía según el esclavo)*
05	Función 05 forzar una salida digital
00	Dirección Alta de la salida digital a forzar
00	Dirección Baja de la salida digital a forzar
FF	Valor Alto concerniente a la acción a realizar la salida digital
00	Valor Bajo concerniente a la acción a realizar la salida digital
F9	LRC de la trama

La Trama de Respuesta (enviada por el Dispositivo Esclavo) será la misma trama que la trama de petición.

Se efectuó otra prueba enviando una trama MODBUS de petición con la dirección de salida digital a forzar equivocada, con lo cual el Dispositivo Esclavo construirá un mensaje de excepción con un código de dirección ilegal y la enviará hacia el Dispositivo Master. En la Figura 4.10 se muestran los resultados de la prueba mencionada.

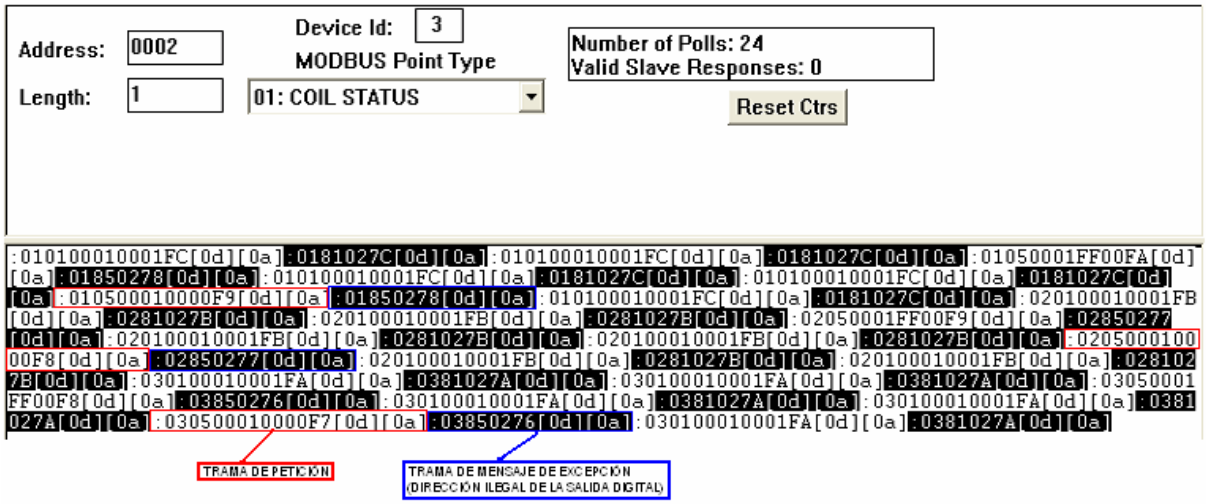

Figura 4.10 a) VISUALIZACIÓN DE TRÁFICO DE RESPUESTAS DE EXCEPCIÓN (ERROR EN DIRECCIÓN)

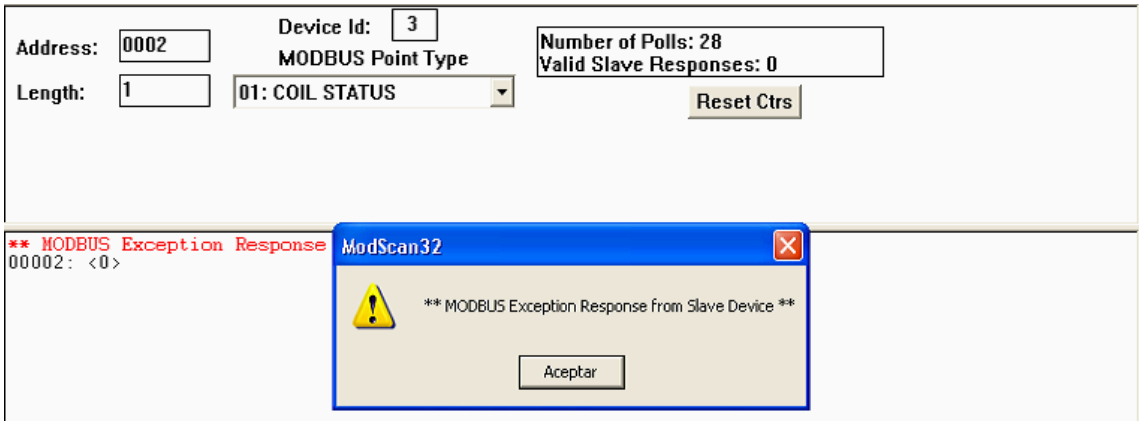

Figura 4.10 b) VISUALIZACIÓN DE DATOS DE RESPUESTAS DE EXCEPCIÓN (ERROR EN DIRECCIÓN)

A continuación se explica el significado de los datos contenidos en dichas tramas para su mejor comprensión:

- ✓ ***Trama de Petición (enviada por el Dispositivo Máster):***

:	Carácter de Inicio
03	Dirección del Esclavo (*varía según el esclavo*)
05	Función 01 de Lectura de Salidas Digitales
00	Dirección Alta de las Salidas Digitales a leer
01	Dirección Baja de las Salidas Digitales a leer
00	Registro Alto del número de Salidas Digitales a leer
00	Registro Bajo del número de Salidas Digitales a leer
F7	LRC de la trama de petición

- ✓ ***Mensaje de excepción (enviada por el Dispositivo Esclavo):***

:	Carácter de Inicio
03	Dirección del esclavo
85	Función *(función de la trama de petición + 80 Hexadecimal)*
02	Código de excepción
76	LRC del mensaje de excepción

De igual manera se realizó la prueba con la función 6, cuyos resultados se pueden ver en los Anexos de este trabajo.

4.2.2 RESULTADOS OBTENIDOS DE LA PRUEBA DE COMUNICACIÓN EN PROTOCOLO MODBUS RTU

En esta sección se indicarán las tramas de envío y recepción de todas las Funciones con las que se trabajo en todos los Dispositivos Esclavos en modo RTU.

4.2.2.1 RESULTADO DE LA PRUEBA DE COMUNICACION UTILIZANDO LA FUNCIÓN 01 EN MODO RTU

En la Figura 4.11 se muestran los resultados de esta prueba. La trama MODBUS de petición aparece en color blanco y la trama MODBUS de respuesta en color negro.

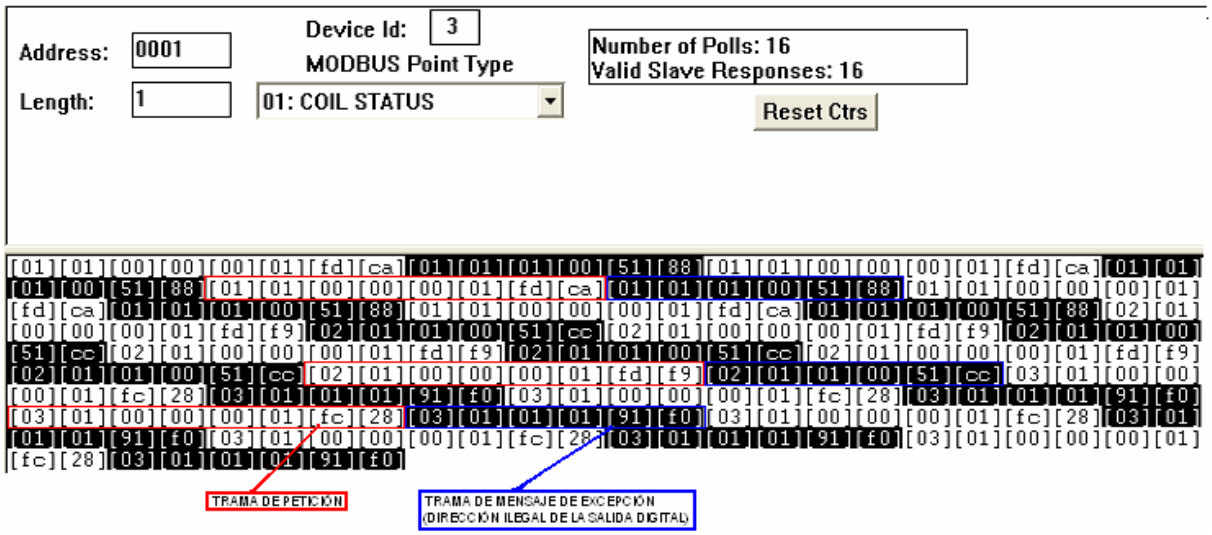

Figura 4.11 VISUALIZACIÓN DE TRÁFICO DE DATOS VÁLIDOS DE LA FUNCIÓN 01

A continuación se realizará el análisis de los datos contenidos en dichas tramas para su mejor comprensión:

- ✓ ***Trama de Pregunta (enviada por el Dispositivo Máster):***

03	Dirección del esclavo
01	Función 01 de Lectura de Salidas Digitales
00	Dirección Alta de las Salidas Digitales a Leer
00	Dirección Baja de las Salidas Digitales a Leer
00	Registro Alto del Número de Salidas Digitales a Leer
01	Registro Bajo del Número de Salidas Digitales a Leer
FC	Registro Alto del CRC de la trama
28	Registro Bajo del CRC de la trama

- ✓ ***Trama de Respuesta (enviada por el Dispositivo Esclavo):***

03	Dirección del esclavo, varia según el esclavo
01	Función 01 de Lectura de Salidas Digitales
01	Número de bytes de estado de las Salidas Digitales
01	Estado de la Salida Digital
91	Registro Alto del CRC de la trama
F0	Registro Bajo del CRC de la trama

Se efectuó la prueba enviando una trama MODBUS de petición en modo RTU con la dirección de la salida digital a leer equivocada, con lo cual el Dispositivo Esclavo construirá un mensaje de excepción con el código dato de dirección ilegal y la enviará hacia el Dispositivo Master. En la Figura 4.12 se muestran los resultados de la prueba mencionada.

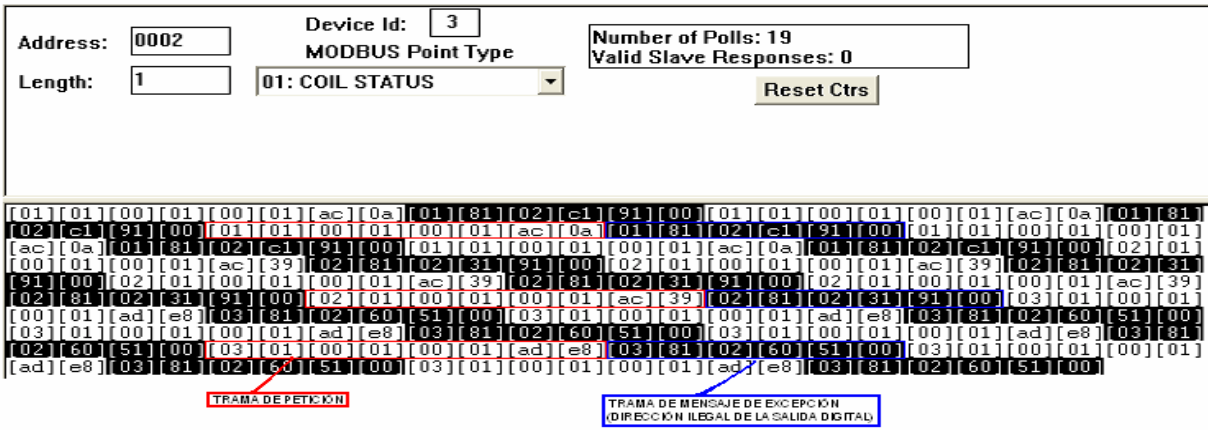

Figura 4.12 a) VISUALIZACIÓN DE TRÁFICO DE RESPUESTAS DE EXCEPCIÓN (ERROR EN DIRECCIÓN)

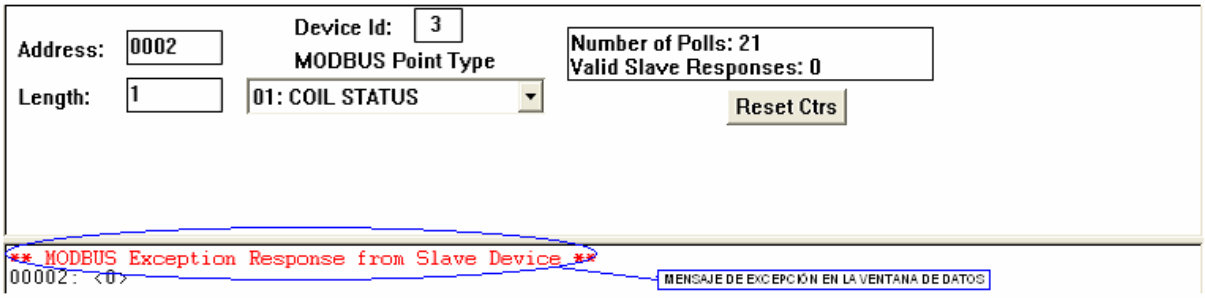

Figura 4.12 b) VISUALIZACIÓN DE DATOS DE RESPUESTAS DE EXCEPCIÓN (ERROR EN DIRECCIÓN)

Se realizó también la prueba enviando una trama MODBUS de petición en modo RTU con el número de la salida digital a leer equivocada, con lo cual el Dispositivo Esclavo construirá un mensaje de excepción con el código dato de valor ilegal y la enviará hacia el Dispositivo Master. En la Figura 4.13 se muestran los resultados de la prueba mencionada.

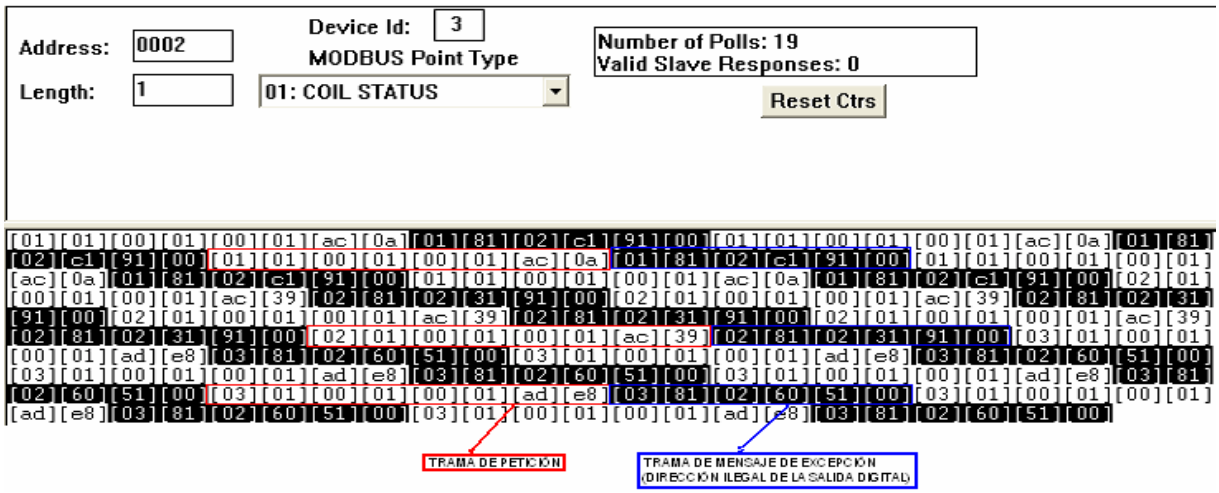

Figura 4.13 a) VISUALIZACIÓN DE TRAFICO DE RESPUESTAS DE EXCEPCIÓN (ERROR EN DATOS)

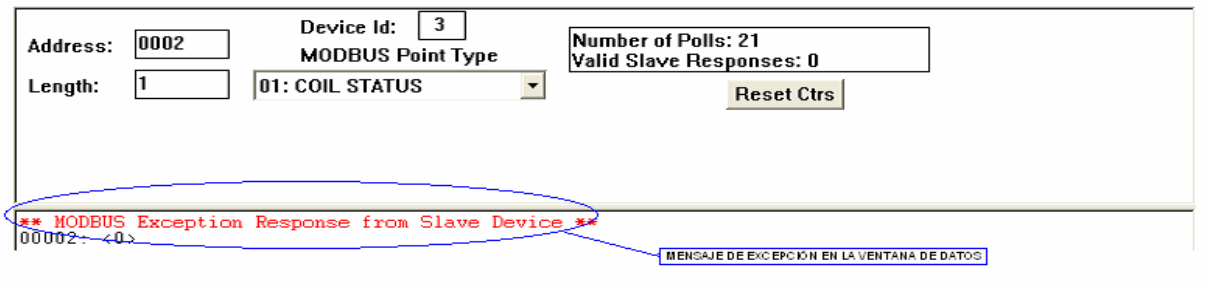

Figura 4.13 b) VISUALIZACIÓN DE DATOS DE RESPUESTAS DE EXCEPCIÓN (ERROR EN DATOS)

De igual manera se realizaron las pruebas para las funciones 02, 03 y 04 en modo RTU, enviando una trama MODBUS de petición para que el Dispositivo esclavo construya una trama MODBUS de respuesta con el estado actual del periférico de entrada o salida. Los resultados de estas pruebas se pueden ver en los Anexos de este trabajo.

Estas pruebas se realizaron en los tres Dispositivos Esclavos.

4.2.2.2 RESULTADO DE LA PRUEBA DE COMUNICACION UTILIZANDO LA FUNCIÓN 05 EN MODO RTU

En la Figura 4.14 se muestran los resultados de la prueba mencionada. La trama MODBUS de petición aparece en color blanco y la trama MODBUS de respuesta en color negro.

A diferencia de las funciones de la 01 a la 04, las funciones 05 y 06 soportan modo BROADCAST.

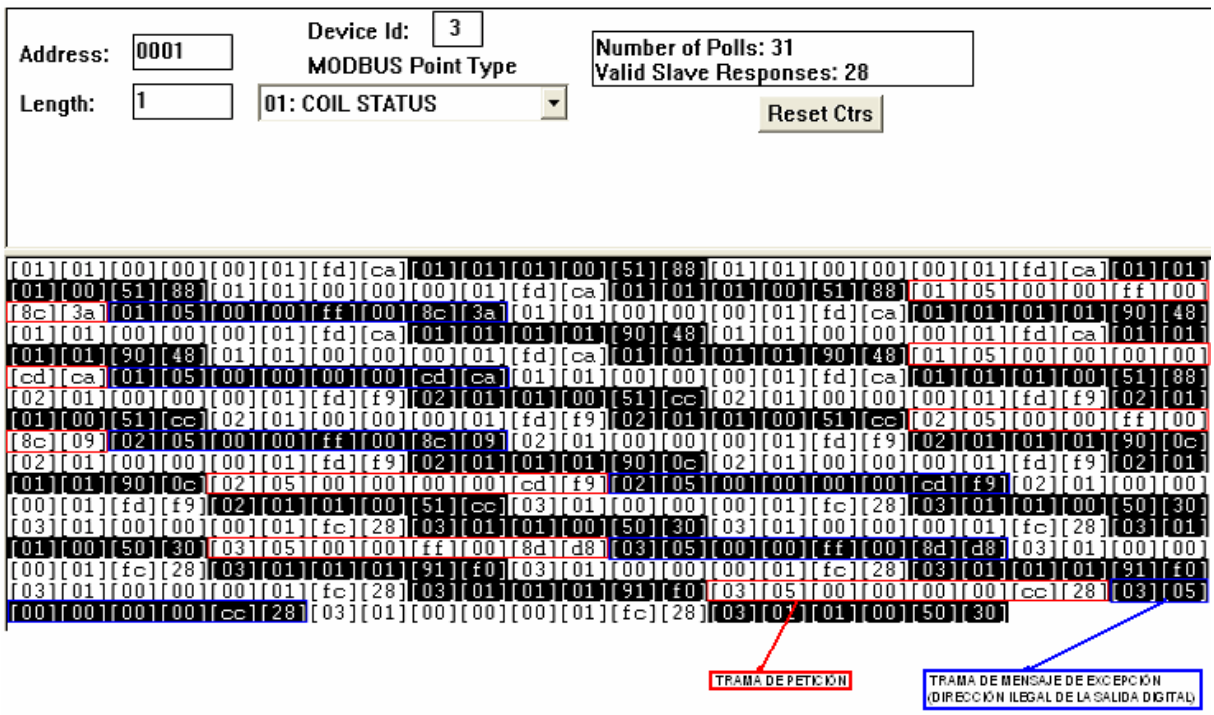

Figura 4.14 VISUALIZACIÓN DE TRAFICO DE DATOS VÁLIDOS DE LA FUNCIÓN 05

La Trama de Respuesta (enviada por el Dispositivo Esclavo) será la misma trama que de petición.

A continuación se realizará el análisis de los datos contenidos en dichas tramas para su mejor comprensión.

- ✓ ***Trama de Pregunta (enviada por el Dispositivo Máster):***

03	Dirección del esclavo *(varia según el esclavo)*
05	Función 05 forzar una salida digital
00	Dirección Alta de la salida digital a forzar
00	Dirección Baja de la salida digital a forzar
00	Valor Alto concerniente a la acción a realizar la salida digital
00	Valor Bajo concerniente a la acción a realizar la salida digital
CC	Registro Alto del CRC de la trama
28	Registro Bajo del CRC de la trama

Se efectuó la prueba enviando una trama MODBUS de petición en modo RTU con la dirección de salida digital a forzar equivocada, con lo cual el Dispositivo Esclavo construirá un mensaje de excepción con el código dato de dirección ilegal y la enviará hacia el Dispositivo Master. En la Figura 4.15 se muestran los resultados de la prueba mencionada.

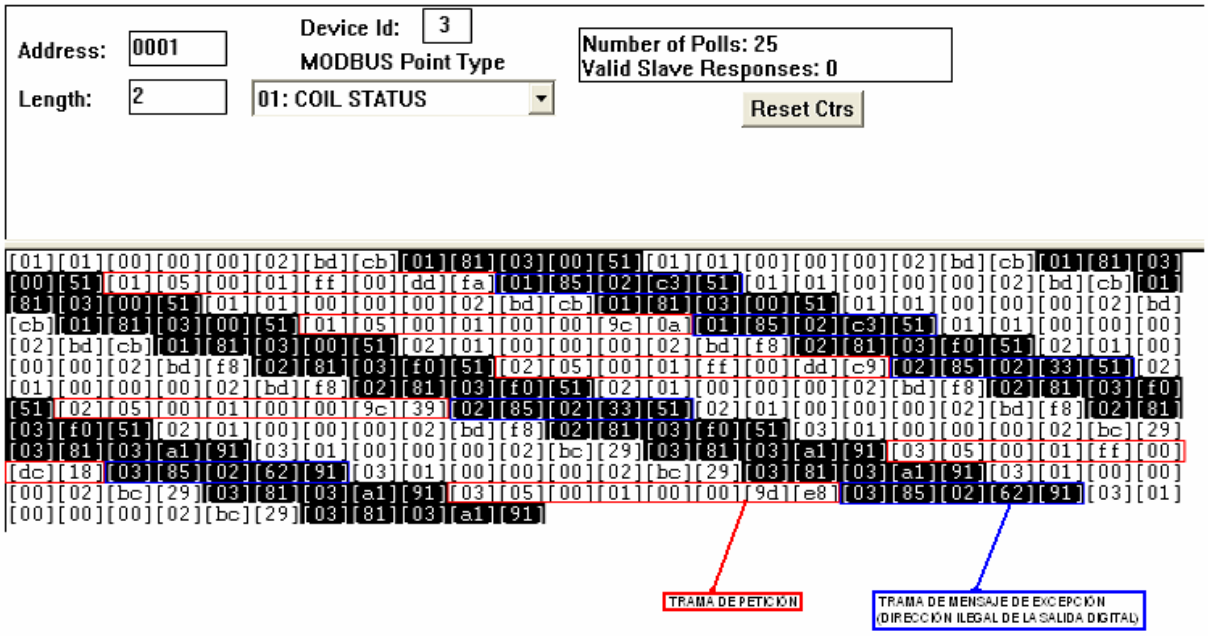

Figura 4.15 a) VISUALIZACIÓN DE TRÁFICO DE RESPUESTAS DE EXCEPCIÓN (ERROR EN DIRECCIÓN)

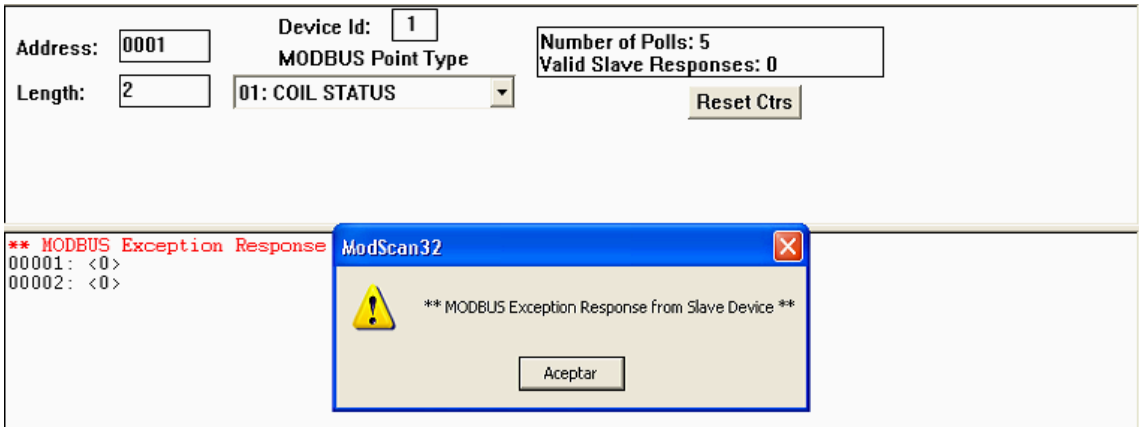

Figura 4.15b) VISUALIZACIÓN DE DATOS DE RESPUESTAS DE EXCEPCIÓN (ERROR EN DIRECCIÓN)

De igual manera se realizó la prueba con la función 6, cuyos resultados se pueden ver en los Anexos de este trabajo.

4.3 PRUEBAS DE FUNCIONAMIENTO DE PERIFÉRICOS EN LOS DISPOSITIVOS ESCLAVOS.

Se realizaron las pruebas a todos los periféricos de entrada y salida en los tres dispositivos esclavos, comprobando el correcto funcionamiento de cada uno de ellos.

4.3.1 PRUEBAS EN LOS PERIFÉRICOS DE ENTRADA

La primera prueba realizada con el Dispositivo Esclavo fue el funcionamiento de la entrada digital.

Para la prueba se cambió el estado de la entrada digital, comprobando el mismo en la pantalla LCD del Dispositivo Esclavo, tal como muestra la Figura 4.16

Figura 4..16 a) ENTRADA DIGITAL EN ESTADO ON b) ENTRADA DIGITAL EN ESTADO OFF

La prueba para la entrada análoga se la realizó cambiando el estado de la entrada análoga, subiendo la temperatura. El cambió se comprobó en la pantalla LCD del Dispositivo Esclavo, tal como muestra la Figura 4.17

Figura 4..17 a) PRIMERA LECTURA ENTRADA ANÁLOGA b) SEGUNDA LECTURA ENTRADA ANÁLOGA

4.3.2 PRUEBAS EN LOS PERIFÉRICOS DE SALIDA

La prueba de los periféricos de salida fue realizada desde el HMI. Una trama hacia el Dispositivo Esclavo fue enviada para cambiar el estado de cada uno de estos periféricos.

En la Figura 4.18a) se muestran los periféricos de salida apagados. En la Figura 4.18b) se muestran los periféricos de salida encendidos.

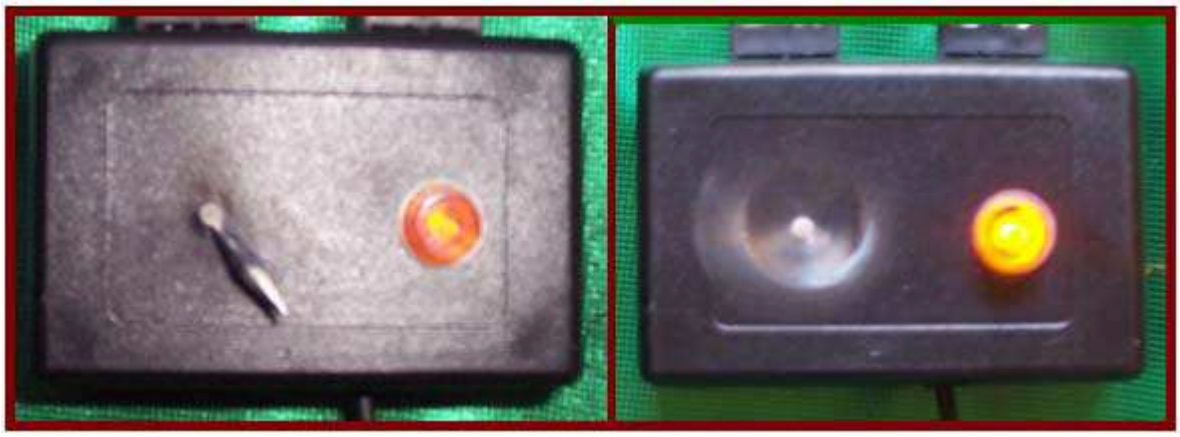

Figura 4.18 a) PERIFÉRICOS DE SALIDA ESTADO OFF **b) PERIFÉRICOS DE SALIDA ESTADO ON**

En el capítulo siguiente se extraen las Conclusiones y Recomendaciones que se pudieron obtener luego de haber realizado las pruebas mencionadas, tomando como base los resultados que dichas pruebas produjeron.

CAPÍTULO 5

CONCLUSIONES
Y
RECOMENDACIONES

CAPÍTULO 5
CONCLUSIONES Y RECOMENDACIONES

Una vez concluidas las pruebas del sistema, en este capítulo se redactarán las conclusiones respecto a este Proyecto de Titulación. De igual manera se escribirán todas las recomendaciones que se han podido extraer luego de la experiencia adquirida durante diseño y construcción de este proyecto.

5.1 CONCLUSIONES

- El objetivo principal del presente proyecto fue contar con una herramienta en el Laboratorio de Interfaces de Comunicación Industrial, para el estudio del Protocolo MODBUS. Para esto se ha implementado un modo de visualización normal donde los estudiantes podrán comprobar el correcto funcionamiento del sistema. Por otro lado, se ha implementado un modo de visualización didáctico en el cual se podrá estudiar las tramas MODBUS para los modos y funciones implementados, con lo cual se concluye que el objetivo planteado fue cumplido.

- El sistema se puede definir como un lazo que se inicia con el envió de la trama desde el computador, pasando por un conjunto de microcontroladores e interfaces (MAX 232 y MAX 485) que se llaman de ***transmisión*** en el Dispositivo Master, para ser posteriormente enviada a los Dispositivos Esclavos. Una vez procesada la trama por los Dispositivos esclavos, se devuelve al Dispositivo Master, a un conjunto de microcontroladores e interfaces (MAX 232 y Max 485) que se llaman de ***recepción***. Entonces el Dispositivo Master enviará esta trama hacia el computador para que el HMI lo procese; completando así el lazo. Los resultados de las pruebas realizadas permiten concluir que el sistema funciona correctamente de la manera descrita.

- De los modos de comunicación se puede decir, como resultado de los estudios realizados, que cada uno de ellos determina cómo debe ser empaquetada la información en los campos del mensaje y los tiempos que este toma. Se concluye que el modo RTU permite un mejor rendimiento que el modo ASCII para la misma velocidad, ya que como principal ventaja tiene su mayor densidad de carácter. En RTU cada mensaje es transmitido en un flujo continuo, mientras que en modo ASCII se tiene intervalos de tiempo de hasta un segundo entre caracteres.

- En lo concerniente a chequeo de errores, el LRC que se añade a la trama ASCII es un algoritmo mucho más sencillo que el chequeo de errores utilizado en el modo RTU. Se comprobó que varias tramas pueden tener el mismo campo LRC debido a que este se lo calcula simplemente sumando todos los campos de la trama (excepto el carácter ":" del inicio y los caracteres "0D" y "0A" del final) y sacando el complemento a 2, sin diferenciar cada campo y sabiendo además que en una suma el orden de los factores no alteran el resultado.

- Para el desarrollo de la interfaz HMI, se utilizó como plataforma el software LABVIEW 6.1 (National Instrument), debido a las facilidades de programación y eficiencia que este ofrece. Se puede concluir que LabView, por la ventaja de que está basado en un lenguaje de programación de tipo gráfico (lenguaje G) simplifica y reduce el tiempo de desarrollo de una interfaz. Consecuente se puede concluir que su selección fue acertada.

- Se optó por implementar en cada Dispositivo Esclavo dos microcontroladores, cada uno de ellos programado con un modo de comunicación (ASCII o RTU). De las opciones analizadas se escogió esta alternativa de separar los dos modos de comunicación, para conseguir que estén totalmente aislados los periféricos a un microcontrolador PIC mientras el otro se mantiene trabajando, lo cual redundó en beneficio del proyecto.

- Mediante el software especializado empleado para las pruebas del protocolo MODBUS, se comprobó que las tramas generadas por los Dispositivos como respuestas a las peticiones simuladas por el ModScan32 cumplen con los requerimientos de dicho protocolo. Por lo tanto se concluye que la parte correspondiente a la generación de tramas MODBUS es correcta.

- Después de haber realizado las pruebas del sistema se puede decir que el sistema operó de manera eficiente y tanto las entradas como salidas (digitales y analógicas) se encuentran en correcto funcionamiento. De igual manera, se comprobó los datos de las tramas recibidas y enviadas; así como los códigos de excepción que el Protocolo exige, tiempos y encapsulados. Todos los elementos que componen este proyecto fueron revisados y finalmente verificados en la puesta en marcha del sistema.

5.2 RECOMENDACIONES

- Se recomienda revisar las conexiones entre la PC y el Dispositivo Master y entre este último con los Dispositivos Esclavos antes de comenzar a usar el Sistema. Se debe recordar que para las conexiones “RS 485” los terminales A se deben conectar entre si, tanto en los terminales esclavos como en el terminal master, de igual manera entre los terminales B.

- Para la comprobación de las Tramas MODBUS se recomienda el software ***Modscan32*** por la simplicidad y prestaciones que este ofrece. De igual manera se recomienda emplear un tiempo (*Slave Response Timeout & Delay Between Polls*) igual o mayor a 3000ms para RTU y un tiempo (*Slave Response Timeout & Delay Between Polls*) igual o mayor a 7000ms para ASCII.

- El sistema, a través de su HMI, cuenta con un control para seleccionar el puerto serial a utilizar. Para saber que puertos seriales tiene la computadora (Windows 2000 y XP), se recomienda seguir los siguientes pasos:

- ✓ Abrir Panel de Control y seleccionar la opción ***Sistema***.
- ✓ Abrir ***Administrador de dispositivos*** en la pestaña Hardware.
- ✓ En la lista se encontrarán los puertos seriales que tenga instalados en su computadora; entonces se debe buscar en el elemento "***Puertos (COM Y LPT)***". Los puertos seriales pueden ser del COM1 al COM5.

- De la información recopilada se recomienda el uso de LABVIEW como plataforma para futuros proyectos. Sus principales ventajas son:

  - ✓ Simplicidad en su manejo, debido a que está basado en un nuevo sistema de programación gráfica, llamada lenguaje G.
  - ✓ Herramientas de presentación, gráficas, botones, indicadores y controles; los cuales serían complicados de realizar en bases como c++ donde el tiempo para lograr el mismo efecto sería muchas veces mayor.
  - ✓ Es un programa de mucho poder donde se cuentan entre sus librerías especializadas como la *librería de comunicaciones*.
  - ✓ Horas de desarrollo de una aplicación por ingeniero se reducen a un nivel mínimo.
  - ✓ La programación de subrutinas en módulos de bloques hace posible el uso de estos en otras aplicaciones.
  - ✓ Es un programa que permite pasar las aplicaciones entre diferentes plataformas como Macintosh y seguir funcionando.

- Finalmente, se recomienda leer el Manual de Usuario de este equipo para su óptima utilización y correcto funcionamiento, además de revisar la teoría acerca del Protocolo MODBUS, poniendo énfasis en las Funciones y Tramas.

REFERENCIAS BIBLIOGRAFICAS

[1] CORRALES, Luis, "*Interfases de Comunicación Industrial*", 2004

[2] COUGHLIN, Robert, "Amplificadores Operacionales", 1998

[3] RASHID, Muhamad, "Electrónica de Potencia", 1970

[4] SEIPPEL, Robert, "Transducers, Sensors & Detectors", 1983

[5] REYES, Carlos, "Aprenda rápidamente a programar microcontroladores PIC", 2004

LINKS

[1] http://www.modicon.com/openmbus

[2] http://es.wikipedia.org/wiki/Modbus

[3] http://www.automatas.org/modbus/dcon7.html

[4] http://www.modbus.org

ANEXOS