IMPLEMENTACION DE UNA RED MODBUS CON LABVIEW

JULIAN J. HERNANDEZ ZAKZUK
ALFREDO R. GUTIERREZ RAPALINO

UNIVERSIDAD TECNOLÓGICA DE BOLIVAR
FACULTAD DE INGENIERIA MECÁNICA Y MECÁTRONICA
CARTAGENA
2006

MONOGRAFÍA
IMPLEMENTACION DE UNA RED MODBUS CON LABVIEW

JULIAN J. HERNANDEZ ZAKZUK
ALFREDO R. GUTIERREZ RAPALINO

PRESENTADO A:

GRUPO DE INVESTIGACIÓN EN AUTOMATIZACIÓN INDUSTRIAL Y CONTROL (GAICO).

DIRECTOR:
MSc. JORGE E. DUQUE

UNIVERSIDAD TECNOLÓGICA DE BOLIVAR
FACULTAD DE INGENIERIA MECÁNICA Y MECÁTRONICA
CARTAGENA
2006

Nota de aceptación:

_____
_____
_____

_____

Director

_____

Calificador

_____

Calificador

Cartagena, Junio de 2006

A nuestras familias, que nos apoyaron durante todo el tiempo que estudiamos.

Cartagena de Indias, 15 de Junio de 2006

Señores
Universidad Tecnológica de Bolívar
Comité de evaluación de Proyectos
Ciudad

Estimados señores:

Con la presente me dirijo a ustedes para poner a consideración el trabajo final titulado “Implementación de una Red Modbus con Labview”, el cual fue desarrollado por los estudiantes Julián Hernández Zakzuk y Alfredo Gutiérrez Rapalino, para optar por el titulo de Ing. Mecatronicos.

Agradeciendo su amable atención,
Cordialmente,

_____

MSc. Jorge E. Duque

AUTORIZACIÓN

Cartagena de Indias, 15 de Junio de 2006

Señores
Universidad Tecnológica de Bolívar
Ciudad

Nosotros, JULIAN HERNANDEZ ZAKZUK y ALFREDO GUTIERREZ RAPALINO, con la presente, autorizamos a la **Universidad Tecnológica de bolívar**, para hacer uso de nuestro trabajo de grado y publicarlo en el catálogo Online de la biblioteca.

_____ _____

Julián Hernández Z. Alfredo Gutiérrez R.

RESUMEN

TITULO: IMPLEMENTACION DE UNA RED MODBUS CON LABVIEW.
AUTORES: Hernández Zakzuk, Julián J, y, Gutiérrez Rapalino, Alfredo R.
PALABRAS CLAVES: Modbus, RTU, Labview, AP, RS 232, RS485.
DESCRIPCIÓN:
En la siguiente monografía se presenta una de las soluciones para la comunicación de un PC con tres APs (Autómatas Programables) o PC`s esclavos, para el monitoreo de esta red se utilizara el protocolo de comunicaciones Modbus.

En el primer capítulo se describen las características generales del protocolo Modbus, como, modos de transmisión (RTU, ASCII), funciones más comunes del protocolo que van a ser implementadas en el programa, etc.

En el capitulo dos se hace una descripción de la implementación del protocolo Modbus en Labview, se hace también un análisis de las tramas que se envían y se reciben.

En el capítulo tres se describe las redes seriales que se van a utilizar (RS 232 y RS 485), una comparación entre estas y también se describe un conversor de estándar RS232 a RS 485 necesario para implementar el bus. En el cuarto capítulo se describen las Tramas que envía el PC maestro como aplicación a los tres PCs esclavos o APs, este capítulo también habla acerca de la aplicación en Labview y los Drivers de comunicación, aquí se presenta un análisis de las librerías de Modbus.
Por ultimo en el capítulo quinto se dan análisis de resultados y conclusiones de acuerdo a los resultados obtenidos.

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

INTRODUCCIÓN

En los últimos años las comunicaciones industriales se han desarrollado muy rápido de manera que podemos encontrar aplicaciones en todos los campos de la vida cotidiana, en el ámbito de la industria por ejemplo, para el control de plantas y procesos, monitoreo y cambiar estrategias de control, etc.

Se eligió el protocolo Modbus porque debido a que la universidad posee un laboratorio de control en el cual se encuentran diferentes equipos de regulación, se hace necesario implementar una red Modbus la cual permita supervisar dichos procesos. Por otra parte es un protocolo que para su implementación es muy fácil conseguir hardware y software.

Para el desarrollo de esta monografía se contó con librerías de la National Instrument, las cuales contienen VIs (Instrumentos Virtuales) con las principales funciones del Protocolo Modbus implementadas. Con la ayuda de estas funciones se programo en Labview el Protocolo Modbus como Maestro y también se programaron tres esclavos, las interfaz del Maestro cuentan con un indicador de la trama Modbus que se envía para verificar la petición del maestro, también tiene un indicador de la trama que se recibe y los parámetros de configuración del puerto, como velocidad, tiempo de espera, paridad, etc. Esto se logro con VIs que se encuentran en las librerías Modbus.

En este trabajo se describen las características principales del protocolo Modbus, como trama enviada, modos de transmisión, arquitectura y da una breve explicación de las cuatro funciones principales de este protocolo; se realiza una breve explicación de las redes seriales más importantes RS 232 y RS 485, debido a que el primer estándar se utilizada en computadores y el segundo es

generalmente el que utiliza el protocolo Modbus en aplicaciones industriales, se debió implementar un conversor de RS 232 a RS 485.

En este documento se presentan los planos y la explicación del funcionamiento del conversor junto con un anexo de la programación que se hizo para el buen funcionamiento del conversor. También se anexa una explicación de cómo se debe conectar el conversor junto con las aplicaciones, todos estos recursos, los conversores, los cables y las aplicaciones son donados a la universidad para la enseñanza e investigación en el campo de las comunicaciones industriales.

Capítulo 1

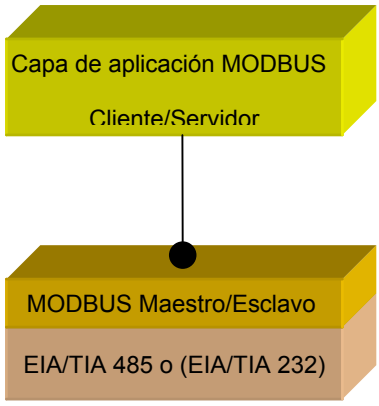
GENERALIDADES DEL PROTOCOLO MODBUS

1.1 Introducción

"Modbus nació como una marca registrada de Gould Inc. y posteriormente fue adquirida por el grupo Sncheneider quien liberó el protocolo en el año 2000" [1]. Como en tantos otros casos, la designación no corresponde propiamente al estándar de red, incluyendo todos los aspectos desde el nivel físico hasta el de aplicación, sino a un protocolo de enlace (nivel OSI 2). Puede, por tanto, implementarse con diversos tipos de conexión física y cada fabricante suele suministrar un software de aplicación propio, que permite parametrizar sus productos. En la figura 1, se muestra el posicionamiento de Modbus sobre el modelo OSI.

Figura 1. Posicionamiento de Modbus sobre el modelo OSI/ISO.

capa	Modelo OSI/ISO	
7	Aplicación	Protocolo de aplicación Modbus
6	Presentación	Vacía
5	Sesión	Vacía
4	Transporte	Vacía
3	Red	Vacía
2	Enlace	Protocolo de línea serial Modbus
1	Física	EIA/TIA-485 o (EIA/TIA-232)



Fuente: http://www.modbus.org/

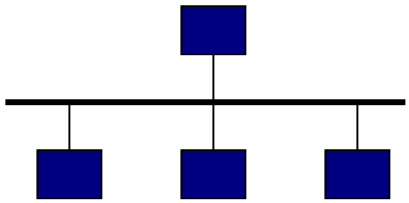
[1] Duque, Jorge. "PROCESADOR DE COMUNICACIÓN MODBUS. IMPLEMENTACIÓN CON MICROCONTROLADOR". Tesis de Maestría Universidad Industrial de Santander. 2005.

El estándar Modbus define un protocolo ubicado en la capa de aplicación, que provee mediante un esquema cliente / servidor, envío de mensajes entre dispositivos conectados en diferentes tipos de buses o redes.

Se suele hablar de MODBUS como un estándar de bus de campo, cuyas características esenciales son las que se detallan a continuación.

- Topología tipo Bus a la cual se conectan todos los dispositivos, en esta topología un nodo coordina y distribuye las tareas.

Figura 2. Topología Bus.



- El protocolo determina cómo cada controlador conocerá su dirección de dispositivo y reconocerá un mensaje direccionado a él, determinará el tipo de acción a tomar y extraerá cualquier dato u otra información contenida en el mensaje.
- Modo de emisión banda base. En banda base, el tren de bits que representan los datos es codificado por medio de un codificador de banda base (ejemplo: «0» corresponde al nivel 12 V, «1» corresponde al nivel -12 V)
- Velocidad de transmisión que varia de los 75 bps a 19200 bps.

Otras características de Modbus de mayor importancia se describen a continuación.

1.2 Medio físico

El la capa física de conexión Modbus emplea interfases físicas diferentes que pueden ser un bus semidúplex (half duplex) (RS-485 o fibra óptica) con par trenzado o dúplex (full duplex) (RS-422, BC 0-20mA o fibra óptica).
La interfaz física de Modbus esta hecha para trabajar en conjunto con el estándar RS 485, que es un esquema multipunto que generalmente tiene una configuración de 2 hilos, en algunos casos se maneja la configuración de 4 hilos .

Todos los dispositivos están conectados en paralelo mediante un cable constituido de tres conductores, dos de los cuales manejan la comunicación bidireccional.

La comunicación es asíncrona y las velocidades de transmisión previstas van desde los 75 baudios a 19.200 baudios, como se mencionó anteriormente. La máxima distancia entre estaciones depende del nivel físico, pudiendo alcanzar hasta 1200 m sin repetidores, no obstante, para alcanzar estas distancias se debe colocar terminadores de línea .

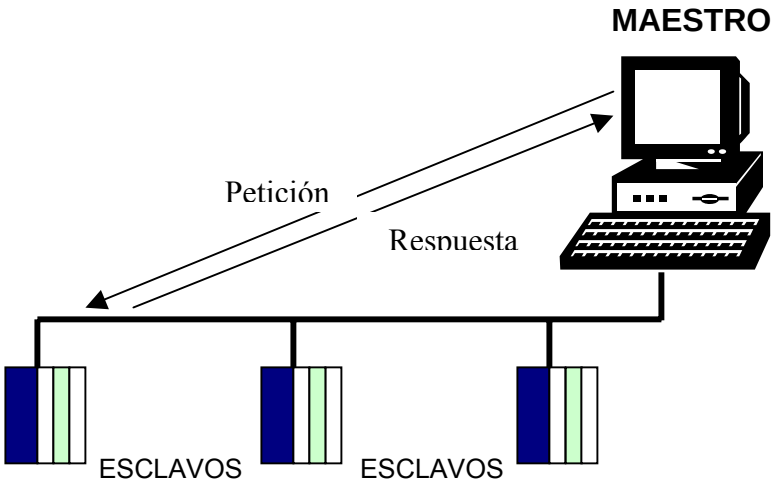
1.3 Acceso al medio

Este protocolo accede al medio utilizando la arquitectura maestro / esclavo, en donde solo un maestro es conectado al bus, y uno o varios esclavos son conectados al mismo bus; una comunicación Modbus siempre es inicializada por el nodo Maestro, ninguno de los nodos esclavos podrá transmitir sin una petición del nodo Maestro, además los nodos esclavos no pueden comunicarse entre si.

El nodo Maestro emite una petición Modbus al nodo esclavo en dos modos:

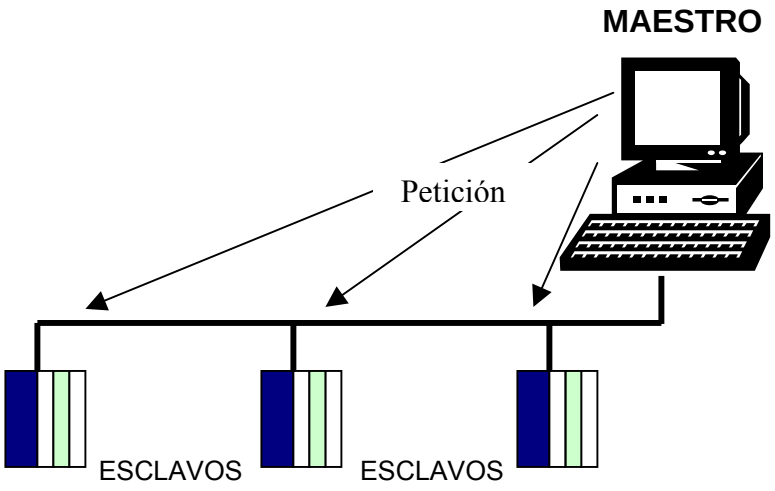
1.3.1 Modo Único: El maestro se dirige a un esclavo individual que después de recibir y procesar la petición, el esclavo devuelve un mensaje ó respuesta al maestro. Ver Figura 3.

Figura 3. Modo Único.



1.3.2 Modo Difusión: El maestro envía una petición a todos los esclavos y estos no responden. Estos mensajes son comandos de escritura y todos los esclavos deben obligatoriamente aceptar la difusión para la función de escritura. Ver figura 4.

Figura 4. Modo Difusión.



1.4 Modos de transmisión serie

En el protocolo Modbus se le debe permitir al usuario la configuración de los dos modos de transmisión serial ASCII y RTU, junto con los parámetros de comunicación del puerto serie. Tanto el modo, como los parámetros series deben ser los mismos en ambos dispositivos conectados a la red Modbus (maestro y esclavos).

Aunque el modo ASCII es requerido en algunas aplicaciones especificas, la interoperabilidad entre los dispositivos Modbus puede ser extendida solamente si cada dispositivo tiene el mismo modo de transmisión.

La elección del modo ASCII o RTU define los bits contenidos en los campos del mensajes transmitido y determina como debe ser empaquetada y decodificada la información en los campos del mensaje.

1.4.1 Modo RTU

Cuando los dispositivos de comunicación son configurados para comunicarse sobre una red Modbus usando el modo RTU (Unidad de Transmisión Remota), cada byte de 8 bits en una trama que se envía 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 de cada byte (11 bits) en modo RTU es:

- **Codificación del sistema:** 8 bits binario.
- **Bits por Byte:**
    - 1Bit de arranque.
    - 8 bits de datos (el menos significativo se envía primero).
    - 1 bit de paridad.
    - 1 bit de parada.

CRC o Palabra de control de errores (2 bytes): El CRC se calcula con una fórmula polinómica según el algoritmo (Chequeo de Redundancia Cíclica), que se muestra en el anexo 1.

El uso de no paridad requiere de 2 bits de parada. Ver figura 5.

Figura 5. Secuencia de transmisión con y sin paridad.

Con paridad										
Start	1 LSB	2	3	4	5	6	7	8 MSB	Par	Stop
Sin paridad										
Start	1 LSB	2	3	4	5	6	7	8 MSB	Stop	Stop

1.4.2 Modo ASCII

Cuando un dispositivo de la red se configura para comunicarse en una red Modbus en modo ASCII (Código Estándar Americano para Intercambio de Información) cada byte de 8 bits en una trama 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.

Este modo es menos eficiente que el modo RTU dado que cada byte necesita dos caracteres. Véase el siguiente ejemplo.

Se tiene el byte **10100001**(8 bits), la parte alta es **1010** y la parte baja **0001**, la parte alta y baja respectivamente en hexadecimal son (A_H y 1_H).
Cada parte es convertida en un código ASCII de acuerdo a la siguiente regla:
Se le suma 30_H a la parte alta o baja si esta entre 0_H y 9_H.
Se le suma 37_H a la parte alta o baja si esta entre A_H y F_H.
De acuerdo a lo anterior:

$A_H + 37_H = 41_H$ $1_H + 30_H = 31_H$

Luego entonces **10100001 = 41_H31_H** Código ASCII.

El formato de cada byte en ASCII es:

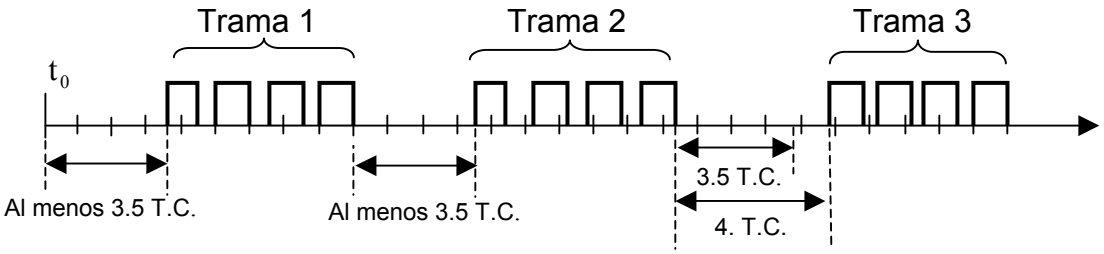
- **Codificación:** Hexadecimal, caracteres ASCII 0,.,9,A,.,F
- **Bits por byte:**
  - 1 bit de arranque.
  - 7 bits de datos (el menos significativo de envía primero).
  - 1 bit de paridad.
  - 1 bit de parada o 2 si no hay paridad.

1.5 Formatos de trama Modbus

1.5.1 Trama RTU

En la trama Modbus RTU, existe un tiempo de silencio de intervalo entre dos tramas de al menos 3.5 veces el tiempo de duración de un carácter. Ver figura 6.

Figura 6. Transmisión de tramas en modo RTU.



Mensaje Modbus

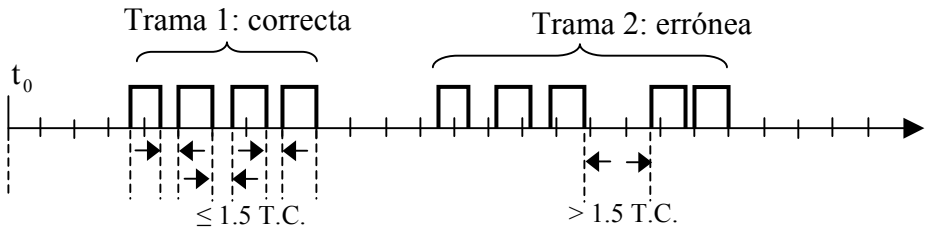
Arranque	Dirección	Función	Datos	CRC	Fin
≥ 3.5 caracteres	8 Bits	8 Bits	N *8 Bits	16 Bits	≥ 3.5 caracteres

Fuente: http://www.modbus.org/

Los caracteres permitidos para todos los campo son 0-9, A-F. Los dispositivos conectados vigilan el bus de red continuamente, incluso en los intervalos de silencio. Cuando se recibe el primer campo (el Campo de dirección), cada unidad lo decodifica para averiguar si es el dispositivo direccionado.

Existe un tiempo de silencio de intervalo entre 2 caracteres: Máximo 1.5 caracteres.

Figura 7. Error en la transmisión de la trama RTU.



Fuente: http://www.modbus.org/

1.5.2 Trama ASCII

La diferencia con la trama RTU es que en la trama ASCII se incluye un carácter de encabezamiento, los dos puntos («:»=3AH) y los caracteres CR y LF(Retorno de Carro y Salto de Línea) (ASCII $0D_H$ y $0A_H$) al final del mensaje como se muestra en la figura 8. Pueden existir también diferencias en la forma de calcular el CRC, ya que en ASCII se utiliza el algoritmo LRC (Chequeo de Redundancia Longitudinal).

Figura 8. Trama Modbus ASCII.

Arranque	Dirección	Función	Datos	LRC	Fin
1 carácter («:»=$3A_H$)	2 caracteres	2 caracteres	0 a 2*256 caracteres	2 caracteres	2 caracteres CR y LF

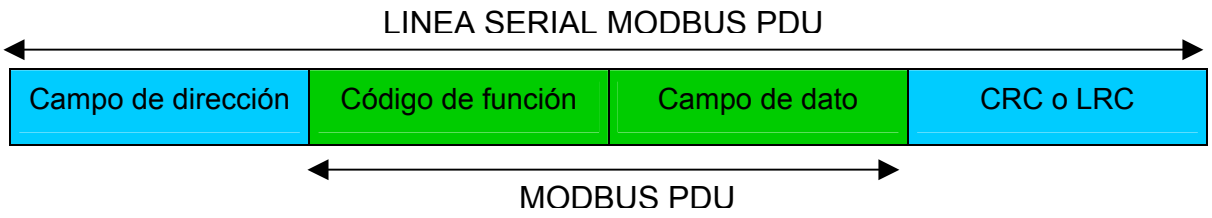
Fuente: http://www.modbus.org/

Los caracteres permitidos en la transmisión para todos los demás campos son 0-9, A-F. Los dispositivos conectados vigilan la red continuamente para detectar el carácter (:). Cuando se recibe, cada dispositivo decodifica el siguiente campo (Campo de dirección) para averiguar si es el dispositivo direccionado.

1.5.3 Trama Modbus

La trama Modbus sobre una línea serial se muestra en la figura 9. y se describe a continuación.

Figura 9. Trama Modbus sobre línea serial.



Fuente: http://www.modbus.org/

1.5.3.1 Campo de dirección

El maestro coloca la dirección del esclavo al que quiere consultar en el campo Dirección del mensaje, para iniciar la comunicación con este. Cuando el esclavo envía su respuesta, coloca su propia dirección en el campo Dirección de la respuesta para confirmar al maestro que es su respuesta.

- Modo ASCII: Contiene 2 caracteres en código ASCII.
- Modo RTU: Contiene 1 byte.
- Direcciones validas para los esclavos: 1-247 decimal.
- La dirección 0 se utiliza para modo difusión, esta es reconocida por todos los dispositivos esclavos.

1.5.3.2 Código de función

Cuando se envía un mensaje desde el maestro a un dispositivo esclavo, el campo de función contiene el Código de Operación de alguna función Modbus, el cual define la acción que debe realizar el esclavo.

- Modo ASCII: Contiene 2 caracteres en código ASCII.
- Modo RTU: Contiene 1 byte.
- Respuesta normal: El esclavo copia el mismo código de la operación original.
- Respuesta de excepción: El esclavo cambia el código de función para indicar que ha ocurrido una anomalía.

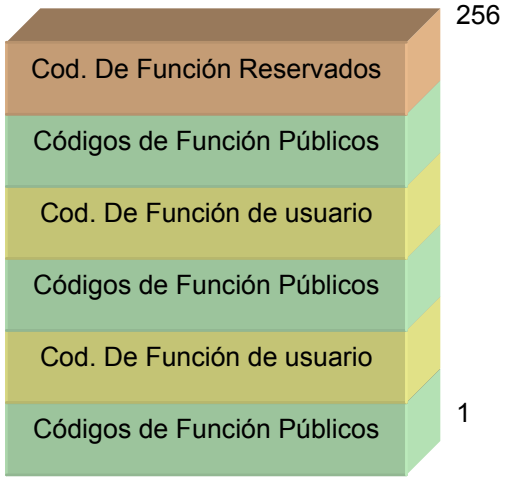
Hay tres categorías de código de operación Modbus, que se pueden apreciar en la figura 10. Códigos de función públicos, códigos de función definidos por el usuario y códigos de función reservados.

Códigos de Función Públicos: Son códigos de función bien definidos, validados y públicamente documentados por la comunidad modbus.org. Los códigos de función público tienen tres rangos estos son de 1 a 65, 72 a 100 y 110 a 127 (decimal).

Códigos de Función Definidos por el Usuario: Son códigos de función que el usuario define para su aplicación específica. Existen dos rangos para estos códigos: 65 a 72 y de 100 a 110 (decimal). El usuario puede implementar estos códigos sin necesidad de su aprobación por la comunidad Modbus.org.

Códigos de Función Reservados: Son códigos de función utilizados por algunas compañías para sus productos y que no son de uso público (rangos: 128-256).

Figura 10. Categorías de códigos de función Modbus.



Códigos Funciones de control y de datos: Aquí se presenta una lista de las principales funciones de control, ver tabla 1. El nombre de la función y una breve descripción de la tarea que realiza.

Las funciones sombreadas tienen permitido el modo difusión (dirección del esclavo = 00), esto significa que la función seleccionada forza la misma referencia en todos los esclavos conectados.

Tabla 1. Códigos de las Funciones de Control.

CÓDIGO		NOMBRE	TAREA
D	Hex		
01	01	**Leer estado de las bobinas.**	Lee el estado de las salidas discretas.
02	02	**Leer estado de las entradas.**	Lee el estado de las entradas discretas.
03	03	**Leer registros internos.**	Lee los contenidos binarios de los registros internos
04	04	**Leer registros de entrada.**	Lee los contenidos binarios de los registros de entrada.
05	05	**Forzar una "Bobina"**	Forzar una "Bobina"
06	06	**Fijar un valor en un registro.**	Fijar un valor en un registro.
07	07	**Leer estado de la excepción.**	Lee el contenido de ocho bits de condición de excepción. Los bits están definidos previamente en cada dispositivo.
08	08	**Diagnostico.**	Permite una serie de pruebas para comprobar el sistema de comunicación entre el maestro y esclavo.
11	0B	**Solicitud de contador de Eventos.**	Devuelve una palabra de estado y un contador de eventos de comunicaciones del esclavo.
12	0C	**Solicitud de diario de Eventos de comunicaciones.**	Devuelve una palabra de estado, un contador de eventos, un contador de mensajes, y un campo de bytes de evento del esclavo.
15	0F	**Forzar varias Bobinas.**	Forzar varias bobinas consecutivas.
16	10	**Fijar Varios registros.**	Fija valores en varios registros internos consecutivos.

1.5.3.3 Campo de datos

Éste campo se forma a través de conjuntos de dos dígitos hexadecimales, en el rango 00-FF_H (recuerde que en modo ASCII cada byte del campo de datos es convertido a código ASCII, ocupando el doble de longitud que en modo RTU).
En las consultas del maestro a los esclavos, el campo de datos contiene información adicional para que el esclavo pueda llevar a cabo la petición del maestro.
El modelo de direccionamiento Modbus esta integrado por 4 bloques de datos estos son:

- Bobinas (Salidas Discretas).
- Entradas Discretas.
- Registros internos.
- Registros de entrada.

1.5.3.4 Campo de control de errores (CRC O LRC)

En redes estándar Modbus se usan dos tipos de comprobación de error. Que dependen del modo de transmisión Modbus de la red:

- Modo ASCII: El campo de comprobación de error contiene dos caracteres ASCII. El valor es el resultado de un Chequeo de Redundancia Longitudinal (LRC) basado en el contenido del mensaje, excluyendo el símbolo ASCII de inicio (:) y los caracteres de control ASCII finales CR-LF (retorno de carro-salto de línea). Los caracteres de LRC se añaden al mensaje como último campo seguidos de los caracteres CR-LF.

- Modo RTU: El campo de comprobación de error contiene dos bytes. El valor es el resultado de un cálculo de Chequeo de Redundancia Cíclica basado en el contenido del mensaje. Los caracteres de CRC se añaden en el último campo del mensaje.

1.6 Principales funciones del protocolo Modbus

Las cuatro principales funciones del protocolo Modbus implementadas se describen en la tabla 2.

Tabla 2. Listado de funciones implementadas en la aplicación.

FUNCIÓN	CÓDIGO	TAREA
1	01_H	LEER ESTADO DE LAS BOBINAS
4	04_H	LEER REGISTROS DE ENTRADA
5	05_H	FORZAR UNA BOBINA
6	06_H	FIJAR UN VALOR EN UN REGISTRO

1.6.1 FUNCION 01: LEER ESTADO DE LAS BOBINAS

Lectura de bits. La forma de direccionamiento de los bits es a base de dar la dirección de la palabra que los contiene y luego la posición del bit. Esta función corresponde básicamente a la lectura del estado ON/OFF de las bobinas discretas de un esclavo.

Ejemplo de Petición – Respuesta:

La trama de petición especifica la bobina inicial de petición y la cantidad de bobinas a leer, las bobinas son direccionadas comenzando en cero: bobinas de la 1 a la 16 son direccionadas de la 0 a la 15.

En la figura 11, se muestra la petición de los estados de las bobinas 20 a 56 del esclavo 17.

Figura 11. Petición de la función 01.

PETICION		
Nombre del Campo	**(Hex)**	
Dirección del esclavo	11	Esclavo # 17
Función	01	
Dirección Inicial Hi	00	
Dirección Inicial Lo	13	Bobina de inicio (20)
Numero de salidas Hi	00	
Numero de salidas Lo	25	Cantidad de bobinas (37)
CRC o LRC	-----	

El estado de las bobinas en la trama de respuesta es codificado como una bobina por bit del campo de datos.

Este estado es representado como 1 = ON y 0 = OFF.

Si la cantidad de bobinas retornadas no es múltiplo de 8, como en este caso, los bits restantes al final del byte serán rellenados con ceros (esto es hacia la parte alta del byte).

Veamos el ejemplo respuesta a la petición anterior.

Figura 12. Respuesta a la función 01.

RESPUESTA		
Nombre del Campo	**(Hex)**	**Binario**
Dirección del esclavo	11	
Función	01	
Conteo de byte	03	
Estados Bobinas(27-20)	CD	11001101
Estados Bobinas(35-28)	6B	01101011
Estados Bobinas(43-36)	B2	10110010
Estados Bobinas(51-44)	0E	00001110
Estados Bobinas(56-52)	1B	**000**11011
CRC o LRC	-----	

El estado de la bobinas de la **27** a la **20** es mostrado con el valor hexadecimal **CD** que en binario corresponde al valor **1**1001101, en este valor binario encontramos que el estado de la bobina **27** corresponde al bit más significativo de la parte alta del byte y el estado de la bobina **20** corresponde al bit menos significativo de la parte baja del byte.

Entonces los estados de las bobinas de la **27** a la **20** de izquierda a derecha son: ON-ON-OFF-OFF-ON-ON-OFF-ON. Y así para los demás estados.

1.6.2 FUNCIÓN 04: LEER REGISTROS

Lectura de palabras. Esta función lee los registros de entrada en el esclavo. La trama de petición especifica el registro de comienzo y la cantidad de registros a ser leídos.

En la figura 13 se muestra un ejemplo de una petición a leer los estados del registro 9 del esclavo 17.

Figura 13. Petición y respuesta con la función 04.

PETICIÓN		RESPUESTA	
Nombre del Campo	**(Hex)**	**Nombre del Campo**	**(Hex)**
Dirección del esclavo	11	Dirección del esclavo	11
Función	04	Función	04
Dirección inicial Hi	00	Conteo de byte	02
Dirección inicial Lo	08	Datos Hi (registro 9)	00
Numero de salidas Hi	00	Datos Lo (registro 9)	0A
Numero de salidas Lo	01		
CRC o LRC	----	CRC o LRC	----

El dato de los registros en la trama de respuesta, es codificado como dos bytes por registro, el byte de la derecha es el menos significativo (contiene los bits menos significativos) y el byte de la izquierda es el más significativo (contiene los bits más significativos). En la respuesta del esclavo el contenido del registro se expresa con dos bytes, 00 $0A_H$.

1.6.3 FUNCIÓN 05: FORZAR UNA BOBINA

Escritura de un bit. El direccionamiento del bit se efectúa como se mostró en la función 01. básicamente forza el estado de una bobina ON/OFF del esclavo, el estado de la bobina permanecerá forzado siempre y cuando no este en la lógica programada del dispositivo, de lo contrario cambiara su valor.

El valor FF 00 hex solicita que la bobina pase a ON. El valor 00 00 hex solicita que pase a OFF. Todo lo demás valores son ilegales y no afectarán al estado de la bobina.

La figura 14 muestra un ejemplo de petición y respuesta de forzar la bobina 173 a ON en el esclavo 17.

Figura 14. Petición y respuesta con la función 05.

PETICIÓN		RESPUESTA	
Nombre del Campo	**(Hex)**	**Nombre del Campo**	**(Hex)**
Dirección del esclavo	11	Dirección del esclavo	11
Función	05	Función	05
Dirección Bobina Hi	00	Dirección Bobina Hi	00
Dirección Bobina Lo	AC	Dirección Bobina Lo	AC
Forza Dato Hi	FF	Forza Dato Hi	FF
Forza Dato Lo	00	Forza Dato Lo	00
CRC o LRC	----	CRC o LRC	----

La respuesta normal como se observa el la figura anterior es un eco de la consulta, devuelve el estado de la bobina después del forzado.

1.6.4 FUNCIÓN 06: FIJAR UN VALOR EN UN REGISTRO

Escritura de una palabra. Esta función fija el valor de un registro interno, este valor permanecerá forzado siempre y cuando no este en la lógica del programa, de lo contrario cambiara su valor.

La trama de consulta especifica la referencia del registro a modificar, Los registros se direccionan empezando en cero: El registro 1 se directora como 0.

La figura 15 muestra un ejemplo de petición y respuesta, en donde al registro 0002 se le pone el valor 0003.

Figura 15. Petición y respuesta con la función 06.

PETICIÓN		RESPUESTA	
Nombre del Campo	**(Hex)**	**Nombre del Campo**	**(Hex)**
Dirección del esclavo	11	Dirección del esclavo	11
Función	06	Función	06
Dirección Registro Hi	00	Dirección Registro Hi	00
Dirección Registro Lo	01	Dirección Registro Lo	01
Dato a escribir Hi	00	Dato a escribir Hi	00
Dato a escribir Lo	03	Dato a escribir Lo	03
CRC o LRC	----	CRC o LRC	----

La respuesta normal es un eco de la consulta, se devuelve después que el registro ha sido modificado.

Capítulo 2

IMPLEMENTACIÓN DEL PROTOCOLO MODBUS

Para la implementación del protocolo Modbus en Labview, se utilizaron los VI's de la librería *NImodbus*[2] de *National Instruments*, la cual permite un desarrollo rápido y relativamente sencillo de la aplicación.

Estos VI's tienen una serie de entradas y salidas que enlazándolas unas con otras y agregando una programación previa, se pueden obtener dos aplicaciones totalmente funcionales y que se comuniquen por medio del protocolo Modbus.

Se desarrollo una aplicación específica donde se pueden modificar datos como, estados lógicos, valores de registros y se pueden visualizar las tramas enviadas y recibidas en un maestro Modbus (ver figura 16).

2.1 Configuración del puerto

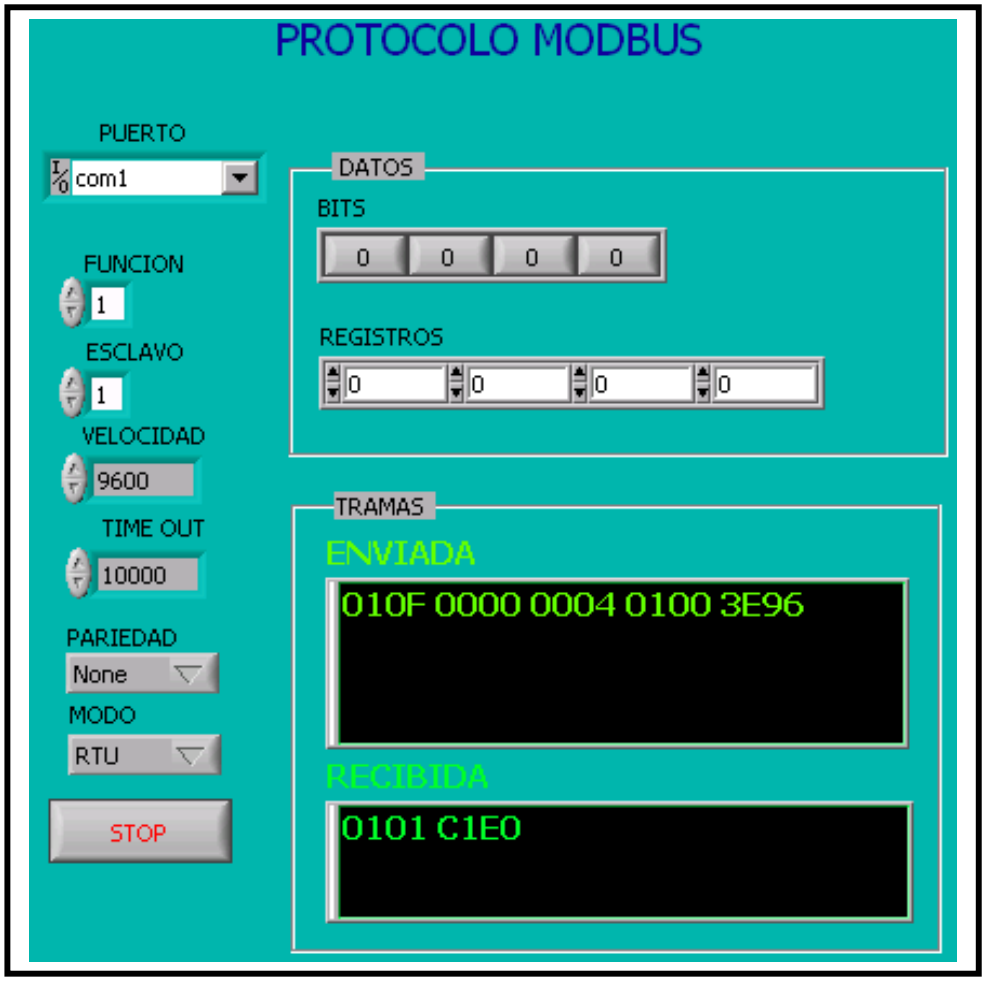
El protocolo Modbus establece que los siguientes parámetros del puerto de comunicaciones deben ser configurados por el usuario, los cuales se pueden apreciar en la figura 16.

- Nombre del puerto (COM1, COM2, COM3, etc).
- Función del protocolo Modbus que se va utilizar.
- Dirección del esclavo al cual se le va a enviar una petición.

[2] National Instruments.
http://sine.ni.com/apps/we/niepd_web_display.display_epd4?p_guid=F1582737BACF5CA8E0340003BA7CCD71&p_node=DZ52363&p_source=External

- La velocidad de transmisión del puerto de comunicaciones.
- Modo de transmisión.
- Paridad.
- Tiempo de espera.

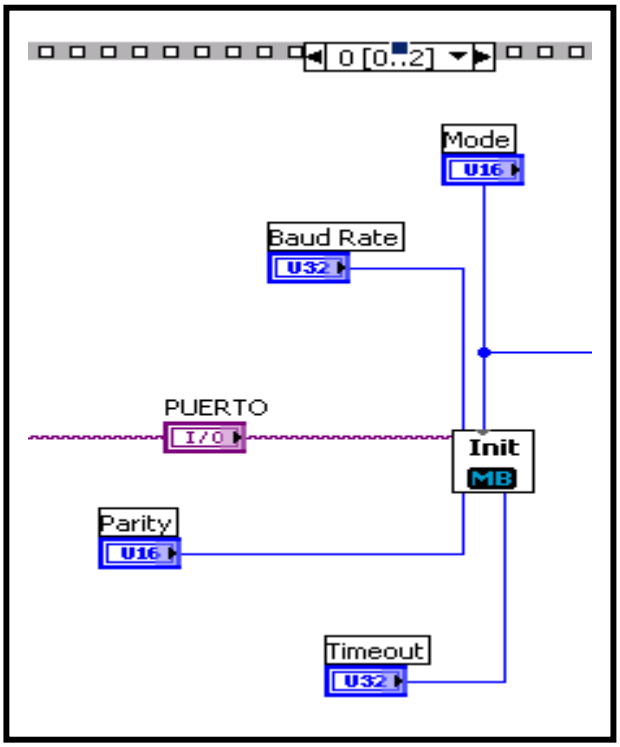
Figura 16. Panel Frontal.



Lo primero que se hace en el programa, es darle las condiciones iniciales al puerto serial para establecer comunicación, estas condiciones se le dan al puerto

utilizando el **VI Init**, al cual se le agregan una serie de controles para poder modificar cada una de las entradas de este bloque (Ver figura 17).

Figura 17. Configuración del puerto serial con el VI Init.

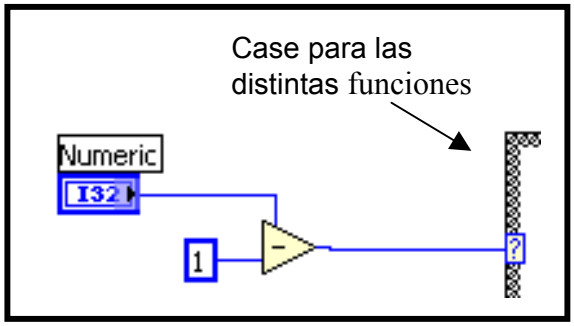


Los controles que se agregaron son: Modo de transmisión (RTU/ASCII), Velocidad (bps), selección del puerto (COM1, COM2, ...), paridad y tiempo de espera (ms).
El dato de modo de transmisión, se envía a través de un túnel hacia la secuencia siguiente, con el objetivo de formar un vector con la información de los parámetros del puerto serial (modo y dirección del esclavo), necesarios para el **VI WR Modbus**, que se explicara mas adelante.

Luego de configurar los parámetros del puerto se agrega una secuencia en donde se selecciona la función que se desea utilizar agregando un control en el panel frontal.

Como el valor de la función, lo establecimos entre 1 y 4, entonces le debemos restar 1 para poder ingresar a la condicional *case*, el cual tiene como primer caso el numero cero '0' (ver figura 18).

Figura 18. Elección de la función.

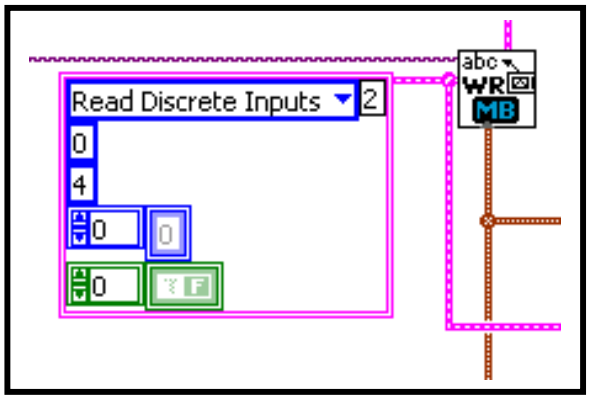


2.2 Función 01: Leer bobinas

Para la implementación de esta función y las demás se utilizo el **VI WR Modbus**, que se encuentra el la librería Modbus el cual por medio de una serie de parámetros nos permite implementar cada una de las funciones. En la figura siguiente se muestra como se utilizo este VI.

A través del parámetro **Modbus Command**, que es una de las entrada de este VI se agrega una constante y se crea el Cluster en donde se selecciona la función. Para este caso (Read discrete inputs).

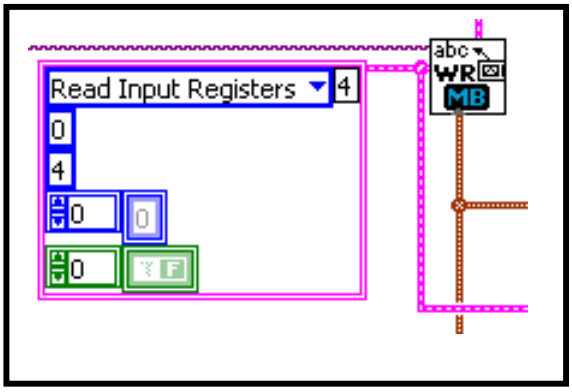
Figura 19. Implementación de la función 01.



2.3 Función 04: Leer registros

Con el mismo VI anterior y a través del mismo parámetro se implementa la función 04 (ver figura 20).

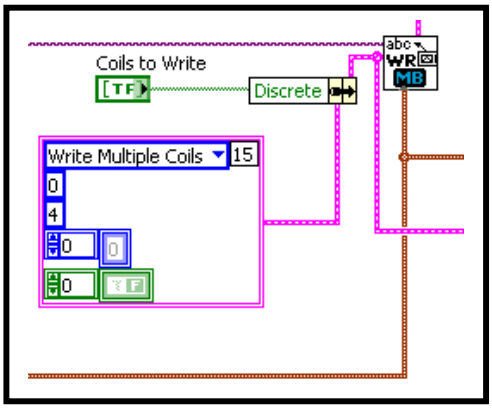
Figura 20. Implementación de la función 04.



2.4 Función 05: Forzar una bobina

Con el mismo VI, pero aparte de seleccionar la función con el parámetro **Modbus Command**, se le agrega el dato que se va a enviar en dicha función, en este caso un vector con la información de 4 bobinas (ver figura 21).

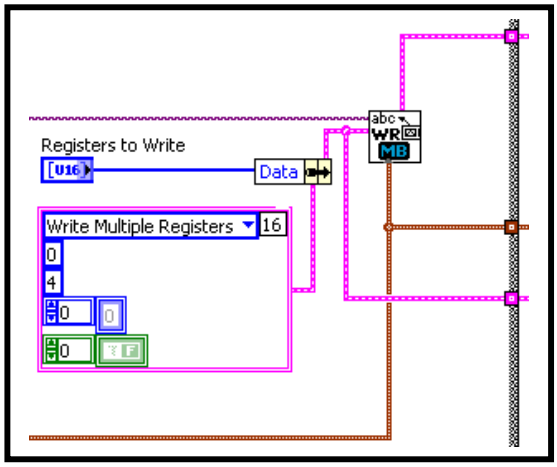
Figura 21. Implementación de la función 05.



2.5 Función 06: Fijar un valor en un registro

Para implementar esta función el procedimiento es el mismo que el anterior solo que el tipo de dato que se va a escribir es diferente, y en el Cluster hay que seleccionar la función adecuada. (ver figura 22).

Figura 22. Implementación de la función 06.



Hasta ahora solo se ha tenido en cuenta un solo parámetro del VI que es el **Modbus Command**, pero como se observa en las figuras anteriores este tiene otra serie de parámetros que son los siguientes.

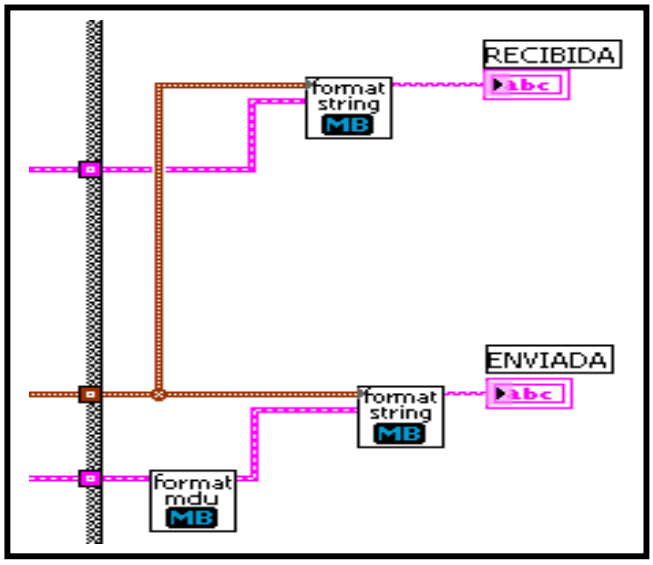
El **VISA resourse name** que es la referencia para el nombre del puerto, **serial parameters** con este parámetro se selecciona el número del esclavo al cual se desea acceder.

2.6 Implementación de la trama

Además de la implementación de cada función, en la parte externa del *case* , y a través de unos túneles se envía la información del **Comand Modbus** (utilizando el VI anterior), que equivale a la trama enviada y también el **Modbus Data Unit Out**, el cual me puede dar la información de la trama recibida. Cabe anotar que esta información la tenemos en formato unidad de datos Modbus, y para visualizarla en pantalla en formato Hexadecimal, debemos convertirla a formato String.

Para visualizar las tramas que se envían y reciben se utilizaron los VI's Format mdu y Format String, y por ultimo las tramas se muestran a través de un visualizador String (ver figura 23), los parámetros de estos VI's se explican en el capitulo 4.

Figura 23. Trama enviada y recibida.



Capítulo 3

REDES SERIALES

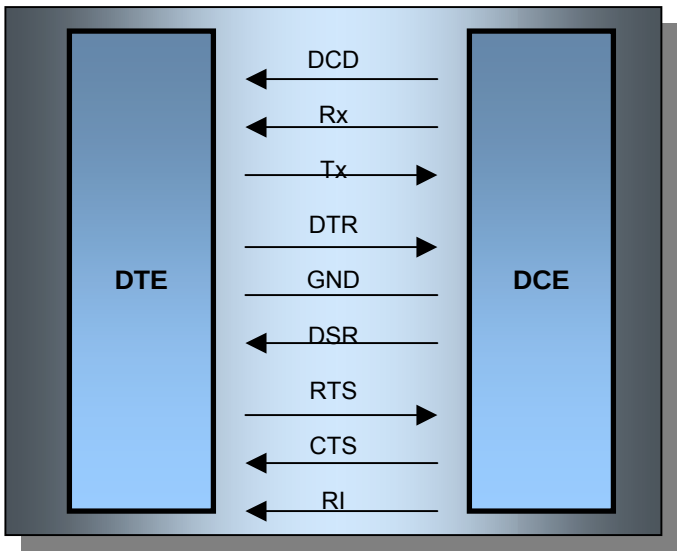
3.1 ESTANDARES RECOMENDADOS

3.1.1 RS 232

Inicialmente se creo el estándar recomendado (RS), en su revisión A, posteriormente se realizarón revisiones: RS-232B, C, D y E. La norma también se encuentra recogida por la asociación de la industria electrónica, EIA, en la EIA 232D, así como por la ITU-T(Unión Internacional de telecomunicaciones, Sector Telecomunicaciones), en normas V24, V28, V10.

La norma RS 232 fue definida inicialmente como una interfaz estándar para conectar un equipo Terminal de datos (DTE, Equipo Terminal de Datos), como un ordenador personal, a un equipo de comunicación de datos (DCE, Equipo de Comunicación de Datos) típicamente un módem, en la figura 24, se muestra el diagrama de dirección de flujo. A pesar de ello, es una norma de conexión de interfaz ampliamente utilizada para conectar una gran variedad de equipos, como autómatas programables, equipos de instrumentación y todo tipo de periféricos. Veamos ahora algunas de sus características.

Figura 24. Dirección de flujo de información.



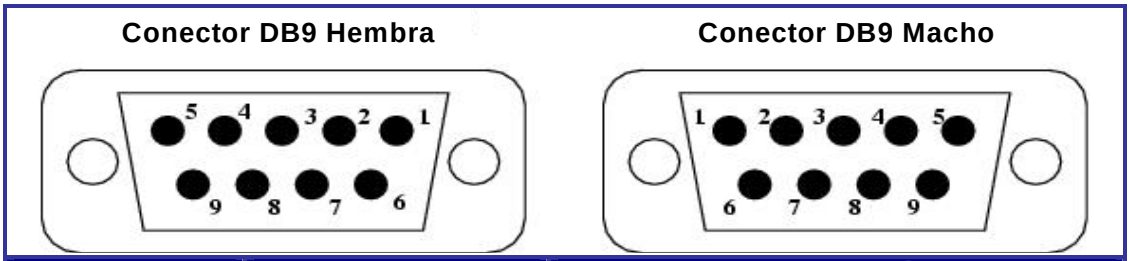
3.1.1.1 Especificaciones Mecánicas

Los conectores utilizados en esta norma suelen ser de dos tipos, DB25 y DB9, macho en el DTE y hembra en el DCE; pero también se encuentran los conectores DIN-8 y RJ-45.

- DB25: Dispone de 25 pines y da acceso a todas las señales recogidas por la norma (ISO 2110).
- DB9: Dispone de 9 pines con las señales más importantes.
- DIN-8: Es pequeño y casi circular. Es usado en Macs, y por ser compacto es a menudo utilizado en laptops. Él provee siete de las señales más comunes de una comunicación serial.
- RJ-45: Tienen ocho alambres. Debido a que son pequeños, ellos son a menudo usados en dispositivos que tienen muchos puertos juntos. Los terminales de servidores son un buen ejemplo de dispositivos que usan conectores RJ-45.

La longitud máxima del cable especificada en la norma es de 15 metros y una velocidad de transmisión máxima de 19200 baudios, auque es posible utilizar longitudes mayores utilizando cables apantallados de baja capacidad, dependiendo además el limite en la longitud de la velocidad de transmisión.

Figura 25. Conectores DB9 Hembra y Macho con designación de pines.



PIN	NOMBRE	DESCRIPCIÓN
1	DCD	Detección de portadora de datos
2	Rx	Recepción de datos
3	Tx	Transmisión de datos
4	DTR	Terminal de datos preparado
5	GND	Tierra
6	DSR	Terminal de comunicación preparado
7	RTS	Petición de envió
8	CTS	Listo para enviar
9	RI	Indicador de llamada

3.1.1.2 Especificaciones Eléctricas

Los niveles de tensión correspondientes a los niveles lógicos empleados en la transmisión se muestran en la tabla 3. Estos valores están referenciados a masa por lo tanto las líneas son desbalanceadas. Las conexiones son punto a punto y no se permite conexiones multipunto.

Tabla 3. Niveles en RS232.

	EMISOR		RECEPTOR	
15v	0 lógico (espacio) activada		0 lógico (espacio) activada	15v
5v	Región de transición		Región de transición	5v
-5v				-5v
-15v	1 lógico (marca) desactivada		1 lógico (marca) desactivada	-15v

Como se puede ver se utilizan niveles de tensión seguros, con corriente en torno a los 3 mA. El rango de voltaje entre los –5v y 5v es considerado como una región de transición conocida como “zona muerta” para el cual el estado de la señal no es asignado, en esta zona se absorbe el ruido de la línea.

3.1.1.3 Especificaciones Funcionales

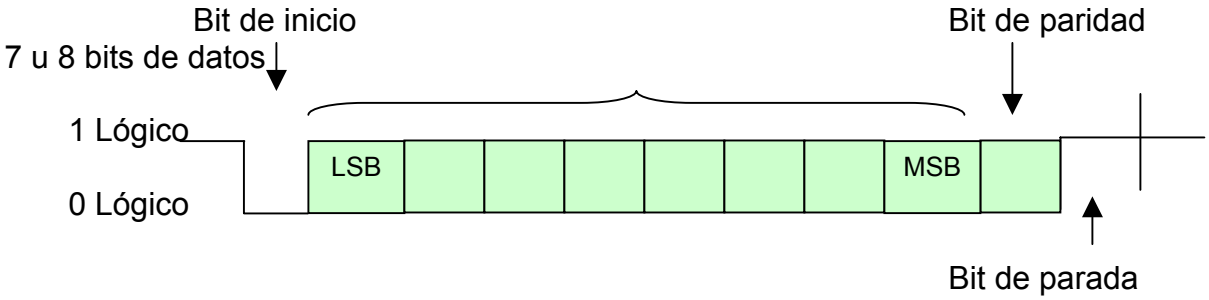
La norma define hasta 25 señales que permiten la transmisión en modo síncrono o asíncrono (con DB 9 solo es posible transmisión asíncrona).
En la transmisión asíncrona que es la que utiliza con mas frecuencia, ver figura 26, para cada carácter se envía.

- Un bit de inicio o arranque. Representado por un 0 lógico.
- 7 u 8 bits de datos, comenzando por el bit de menor peso hasta el de mayor peso.
- 0 o 1 bit de paridad para control de errores. La paridad puede ser:
    - Marca: El bit siempre esta a 1.
    - Espacio: El bit siempre esta a 0
    - Par: El numero total de unos (1) es par, incluyendo el de paridad.

- Impar: El numero total de unos (1) es impar, Incluyendo el de paridad.

- 1, 1.5 o 2 bits de parada, representados por 1 lógico.

Figura 26. Transmisión asíncrona.



Cuando no se esta realizando transmisión la línea permanece en nivel alto.
Las velocidades de transmisión establecidas en la norma son : 110, 300, 600, 1200, 2400, 4800, 9600 y 19200 baudios.

3.1.1.4 Señales más utilizadas

El estándar RS-232 dispone de 25 líneas para la transmisión, pero muchas de estas líneas pertenecen a conexiones donde el dispositivo DCE es un módem; para cualquier dispositivo DCE que no sea un módem, o cuando dos dispositivos DTE son conectados directamente, se requieren de pocas líneas. En la Tabla 4, se hace una descripción de las líneas de las señales más utilizadas.

Tabla 4. Señales más utilizadas.

SEÑAL	FUNCIÓN
GND	Esta línea proporciona la referencia de tierra para todas las demás señales.
Tx	Por esta línea se transmiten los datos. Cuando no hay transmisión de datos, la señal se mantiene en la condición de marca (1 lógico, voltaje negativo).
Rx	Por esta línea se reciben los datos. Cuando no se están recibiendo datos, la señal se mantiene en la condición de marca (1 lógico, voltaje negativo).
DSR	Esta señal es sostenida en 0 lógico (voltaje positivo) por el DCE para indicar al DTE que está conectado a la línea
CTS	Esta señal es sostenida en 0 lógico (voltaje positivo) por el DCE para informar al dispositivo DTE que la transmisión puede comenzar. RTS y CTS son usados comúnmente como señales para controlar el flujo de datos dentro del dispositivo DCE.
RTS	Esta señal es sostenida en 0 lógico (voltaje positivo) por el DTE para que el DCE se prepare a recibir el dato enviado. Tal preparación podría incluir la habilitación de los circuitos de recepción, o la colocación de la dirección del canal en aplicaciones half-duplex. Cuando el DCE está listo, éste acepta por medio de la línea CTS.
DTR	Esta señal es sostenida en 0 lógico (voltaje positivo) por el dispositivo DTE cuando éste quiere comenzar la comunicación.
RI	Esta señal es relevante cuando el dispositivo DCE es un módem, y es mantenido en 0 lógico (voltaje positivo) cuando una llamada está siendo recibida de la línea telefónica
DCD	Esta señal es sostenida en 0 lógico (voltaje positivo) por el DCE para avisar al DTE que está recibiendo una señal portadora con información.

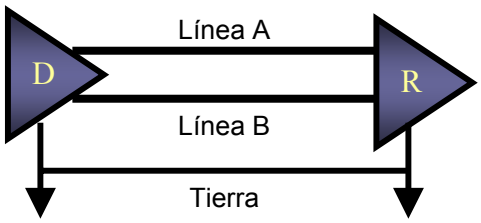
3.1.2 RS 485

Con respecto a la norma RS 232, se puede observar que tiene bastantes limitaciones; las conexiones pueden ser únicamente punto a punto, y además debido a que las señales son referenciadas a masa, no se pueden lograr grandes velocidades y distancias de transmisión.

Por otro lado el estándar RS 485 (existen normas anteriores como la RS422 y RS423, que no se trataran en este trabajo), permite la transmisión diferencial balanceada en redes multipunto, y su uso esta ampliamente extendido en ámbitos industriales.

Recordemos que, la transmisión diferencial balanceada de datos se usa para la transmisión confiable de información a altas velocidades y sobre grandes distancias y a través de ambientes ruidosos. Se usa un par de alambres para llevar cada señal, la información se envía y recibe en cada par, como un voltaje diferencial entre éstos, comúnmente llamados A (voltaje negativo) y B (voltaje positivo). Veamos una representación de esto en la figura 27.

Figura 27. Esquema de línea balanceada.



En el estándar RS 485, el dispositivo que envía la información por medio de un voltaje diferencial por las líneas A y B se llama Driver (D), y el dispositivo que recibe el voltaje diferencial se llama receiver (R) como se observa el la figura anterior. Veamos ahora algunas de las características del estándar en cuestión.

- Permite hasta 32 emisores receptores en la misma línea, (dependiendo ampliamente del tipo de *drivers* empleados).
- Transmisión diferencial con niveles de salida de $\pm$ 1.5v a $\pm$ 5v.
- Los niveles lógicos se representan por:
    - A<B → 1 Lógico (marca o señal de control desactivada).
    - B<A → 0 Lógico (espacio o señal de control activada).
- El voltaje en modo común puede variar en el rango +12v a –7v.
- Longitud del cable de hasta 1200m.
- Velocidad de transmisión de hasta 10Mbps, dependiendo de la longitud del cable, para velocidades muy grandes se requiere el empleo de terminadores en la línea.

3.1.2.1 Conexión de RS-485

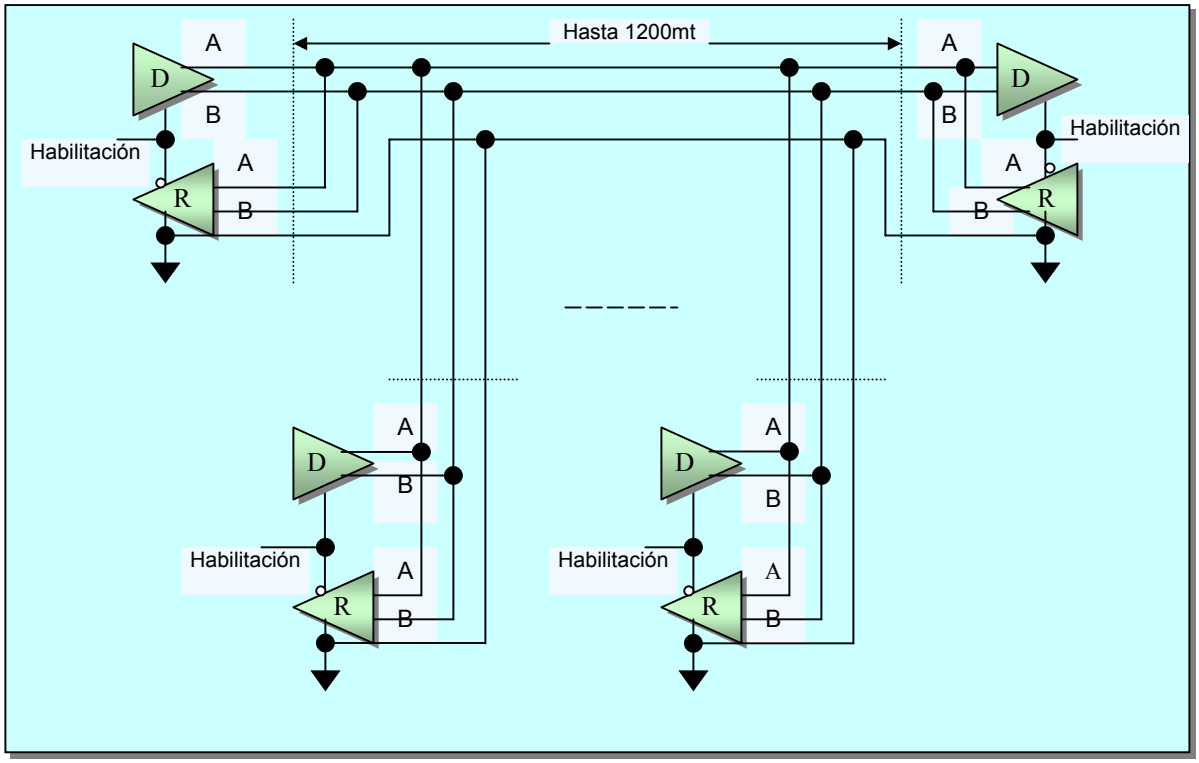
El estándar permite la conexión con topología de bus, conectándose así a varios dispositivos (comunicación multipunto); usando el mismo par de hilos para la transmisión y la recepción (comunicación half-duplex). Esto requiere un control en la conmutación de la línea, esta conmutación se hace por medio de la habilitación de los *drivers* y los *receivers*. El número máximo de dispositivos que pueden ser conectados al bus es de 32.

3.1.2.2 Conexión multipunto

Por medio de esta conexión cada dispositivo puede enviar y recibir información hacia y desde todos y cada uno de los demás dispositivos conectado al bus. La conexión multipunto utilizando el estándar RS-485 se aprecia en la Figura 28.
Cuando un dispositivo desea enviar información su driver debe estar habilitado y los drivers de los demás dispositivos no, además el receiver del dispositivo que envía información se deshabilita y los demás receivers en el bus tienen que

habilitarse. Esta configuración permite que cualquier dispositivo en un momento dado sea un maestro y los demás esclavos.

Figura 28. Conexión multipunto del estándar RS 485, se omite en el dibujo las resistencias de terminación de línea.



3.1.3 Comparación de los estándares

Como resumen de las normas que se mencionaron anteriormente (RS 232 y RS 485), podemos decir que son las más empleadas en el ámbito industrial, la primera de ellas, RS 232, es ampliamente utilizada para la conexión de PLCs a terminales de programación y pantallas de visualización de datos. Por otra parte RS 485 es una de las más empleadas en el nivel físico para la implementación de buses de campo, en muchos casos de tipo maestro / esclavo, como es el caso de

Modbus. En la tabla 5 se resumen las principales características que presenta cada una de estas normas, en la que se pueden apreciar las diferencias, especialmente en prestaciones.

Tabla 5. Comparación de los estándares RS 232 y RS 485.

	RS 232	RS485
Cableado	Punto a punto.	Multipunto.
Nº de dispositivos	1 emisor 1 receptor.	32 emisores 32 receptores.
Modo de comunicación	Dúplex completo.	Semidúplex.
Longitud máxima	15m a 19.2 Kbps.	1200m a 100 Kbps.
Velocidad máxima	19.2 Kbps para 15m.	10Mbps para 15m.
Tipo de señal	Referenciada a masa.	Diferencial balanceada.
Nivel lógico 1 (marca)	-5V máx. -15V mín.	1.5V mín. (B>A) 5V máx. (B>A)
Nivel lógico 0 (espacio)	5V mín. 15V máx.	1.5V mín. (A>B) 5V máx. (A>B)
Nivel entrada mínimo	±3V	200mV diferencial
Corriente de salida	500mA. (En PCs los driver están limitados a 10mA)	250mA.
Impedancia de salida del generador	3KΩ a 7KΩ	54Ω
Impedancia de entrada del receptor	3KΩ a 7KΩ	12KΩ

3.2 Conversor RS 232 A RS 485

La implementación de este conversor se hace necesario para la conexión de un PC maestro, el cual tiene implementado el protocolo Modbus en Labview, a tres PCs esclavos, como sabemos el estándar RS 232 solo soporta conexión punto a punto, por lo tanto es necesario para implementar la topología bus que se convierta de este estándar (RS232) a RS 485.

Básicamente el conversor convierte los niveles de tensión del estándar RS 232 a niveles de tensión TTL/CMOS por medio del circuito integrado MAX 232 y luego los niveles de tensión de TTL/CMOS son convertidos a niveles de tensión del estándar RS 485 por medio del circuito integrado ADM 485.

A continuación se presenta el diagrama lógico del circuito integrado MAX232 y del ADM485.

Figura 29. Diagrama lógico del MAX 232.

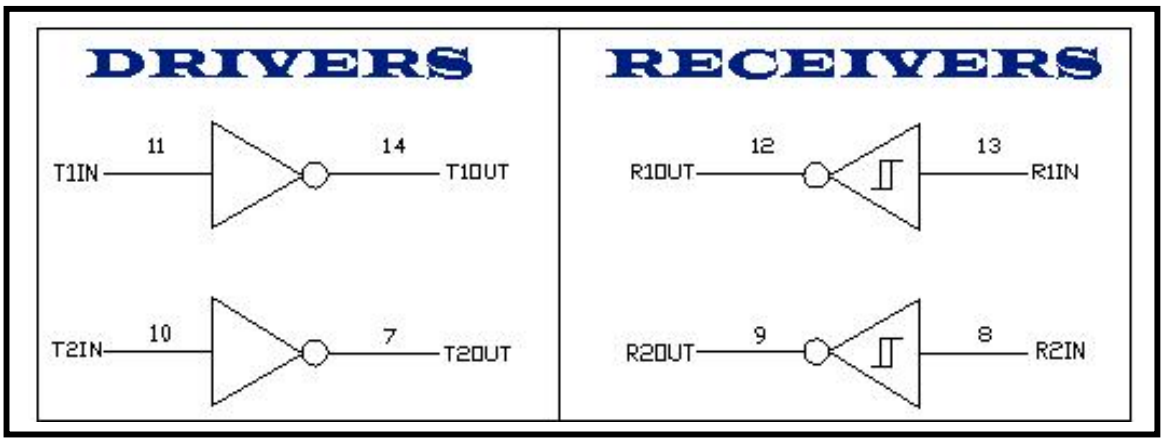
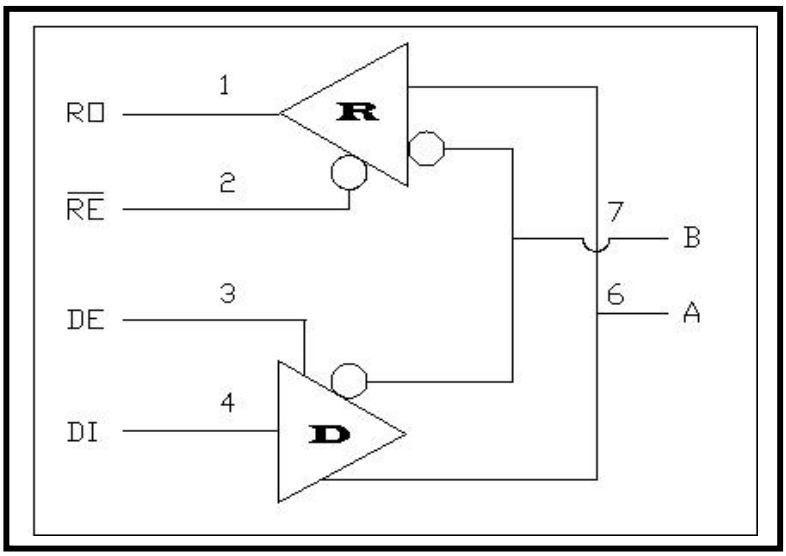


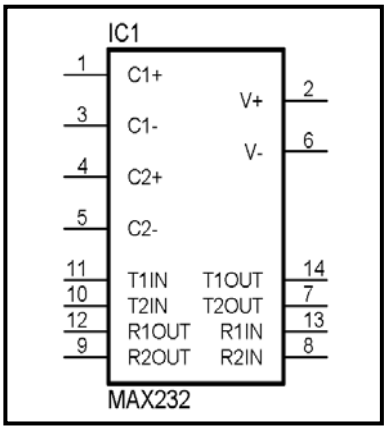
Figura 30. Diagrama lógico del ADM 485.



3.2.1 MAX232

Este circuito integrado nos permite convertir los niveles de tensión de la norma RS232 a niveles de tensión TTL/CMOS de 5V por medio de sus *receivers* y por medio de sus *drivers*, convierte entradas TTL/CMOS a niveles de tensión RS232, la Figura 31, muestra el diagrama de pines del MAX232, este circuito integrado se alimenta por el pin 16 con 5V y la tierra va puesta al pin 15, entre los pines 1 y 3, 4 y 5 se conectan capacitores de 22μF con la polaridad indicada en los subíndices, entre el pin 2 y tierra se conecta un capacitor de 22μF y entre el pin 6 y tierra también se conecta un capacitor de 22μF.

Figura 31. Circuito integrado MAX232.

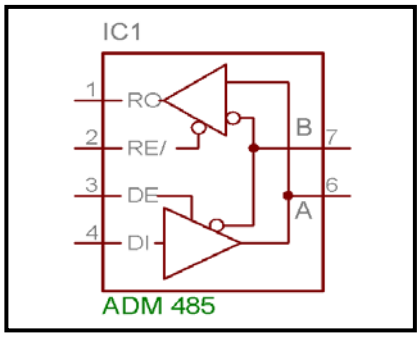


3.2.2 ADM485

Este circuito integrado nos permite cambiar los niveles de tensión TTL/CMOS referenciado a tierra a un voltaje diferencial RS 485 por medio de su *driver* y viceversa (de RS 485 a TTL/CMOS), por medio de su *receiver*. Este circuito integrado es Semidúplex, por lo tanto se le debe aplicar un señal de habilitación a los pines DE y $\overline{RE}$ para controlar el flujo o dirección de datos en el bus. La señal DE (*Driver enable*) habilita el driver para enviar información al bus y la señal $\overline{RE}$ (*Receiver enable*) habilita el *receiver* para recibir información del bus.

La Figura 32, muestra la distribución de pines del ADM485. El circuito integrado se alimenta por el pin 8 con una tensión de 5V y la tierra del circuito debe ser conectada al pin 5. La señal de habilitación del *driver* se conecta al pin DE (3) y la del *receiver* al pin (2). La conexión al bus se hace por medio de los pines A (6) y B (7), la salida del *receiver* es RO (1) y la entrada del *driver* es DI (4).

Figura 32. Circuito integrado ADM 485.



El *driver* necesita tener la habilitación DE en alto (1) para obtener una salida en A y B; de lo contrario las salidas A y B quedan en alta impedancia.

El *receiver* solo obtiene una salida en RO si se tiene la habilitación $\overline{RE}$ en bajo (0), si está en alto (1) la salida RO queda en alta impedancia. Si por algún caso las salidas quedan abiertas, es decir, sin conectar; y la habilitación $\overline{RE}$ se encuentra en bajo (0); en la salida RO se obtiene un alto.

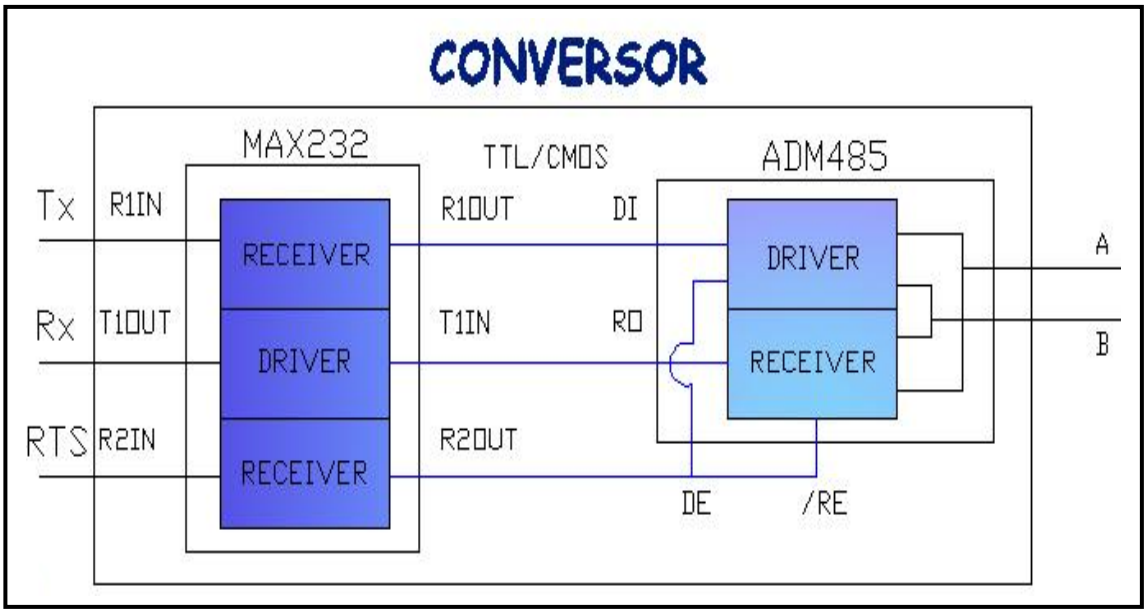
3.2.3 Control de flujo de información

El control de flujo de información se hace por el pin triestado que posee el *driver* y *receiver* del ADM485, que es fundamental para controlar el tiempo que dichos dispositivos se encuentran transmitiendo, como la conexión que utilizamos es la de dos hilos, se debe aplicar la misma señal de habilitación tanto al *driver* como al *receiver*, esto es a los pines DE y $\overline{RE}$ del ADM485; de modo que dicho nodo no reciba los datos cuando los está transmitiendo, obteniéndose así una sola dirección en la comunicación.

Para obtener la dirección adecuada en la comunicación se empleo el pin RTS del conector DB9 del nodo Maestro, como ésta señal está en niveles de tensión de RS 232 se pasa a través de un *receiver* del MAX232 para que se pueda aplicar a las habilitaciones DE y $\overline{RE}$ del ADM485.

A diferencia de las señales que envía el computador con el estándar RS 232 por el puerto DB9, cuando se usa la línea RTS de forma independiente se tiene que un 0 es un voltaje negativo y un 1 es un voltaje positivo. Cuando se envía información desde el nodo Maestro, un instante antes de comenzar la transmisión se coloca la línea RTS en 0 y cuando se hayan enviado los datos se pasa a 1, este cambio de estado de la línea RTS se puede ver en el anexo 2. Esto produce un pulso negativo que al pasar por un *receiver* del MAX232 se convierte en un pulso positivo, habilitando el *driver* (y deshabilitando el *receiver*) del ADM485; permitiendo el envío de los datos hacia el bus. Mientras el computador no envía datos, la señal RTS se mantiene en 1, que es un 0 después de pasar por un *receiver* del MAX232, conllevando a la habilitación del *receiver* para la recepción de información del bus. (figura 33).

Figura 33. Diagrama del Conversor RS232/RS485.



3.2.4 Resistencias de bias

Las resistencias de bias (pull-up y pull-down) son necesarias para forzar el estado de las líneas a un nivel lógico estable, estas consisten en una resistencia de pull-up en la línea B (a +5V), y una resistencia de pull-down (a tierra) en la línea A. El valor de estas resistencias depende del numero de nodos y del valor de las resistencias de terminación.

Estas resistencias se pueden colocar en cualquier punto de la red o inclusive se pueden repartir en cualquiera de los nodos. Un valor típico de resistencias de bias utilizado en convertidores industriales es de 4.7KΩ adecuado para la mayoría de sistemas sin terminadores. No obstante este valor debe ser calculado ver anexo 3.

En la figura 34 que se muestra a continuación se detalla el circuito esquemático del conversor.

Figura 34. Circuito esquemático del conversor RS 232 a R S485.

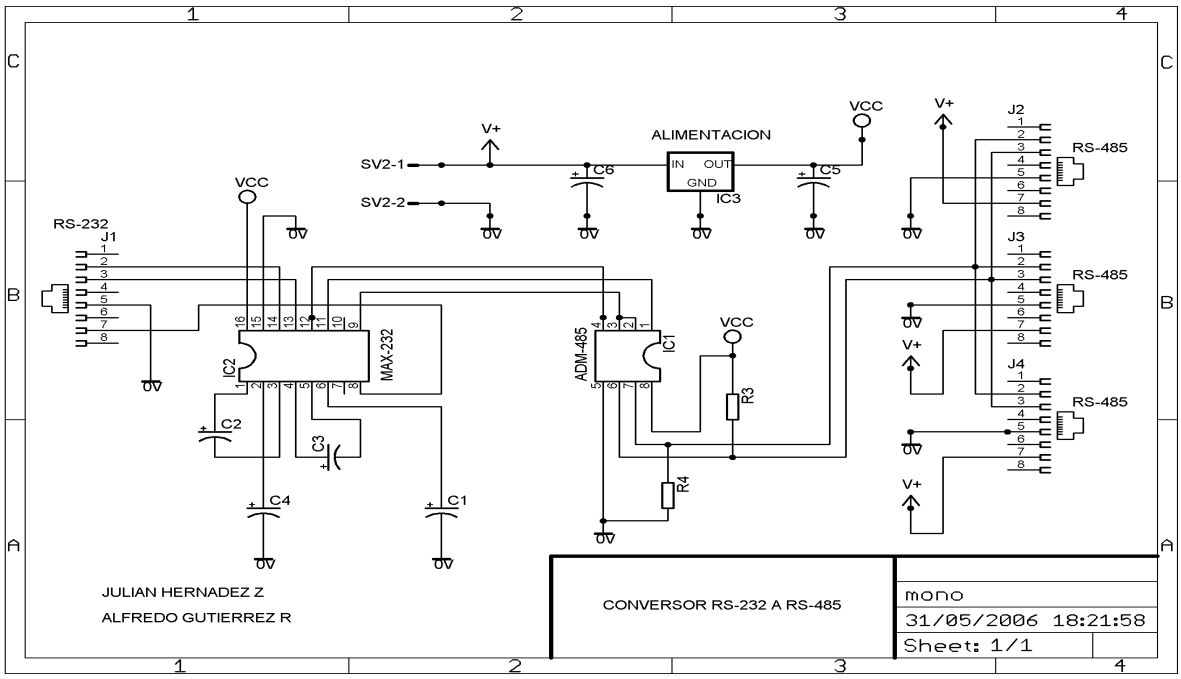
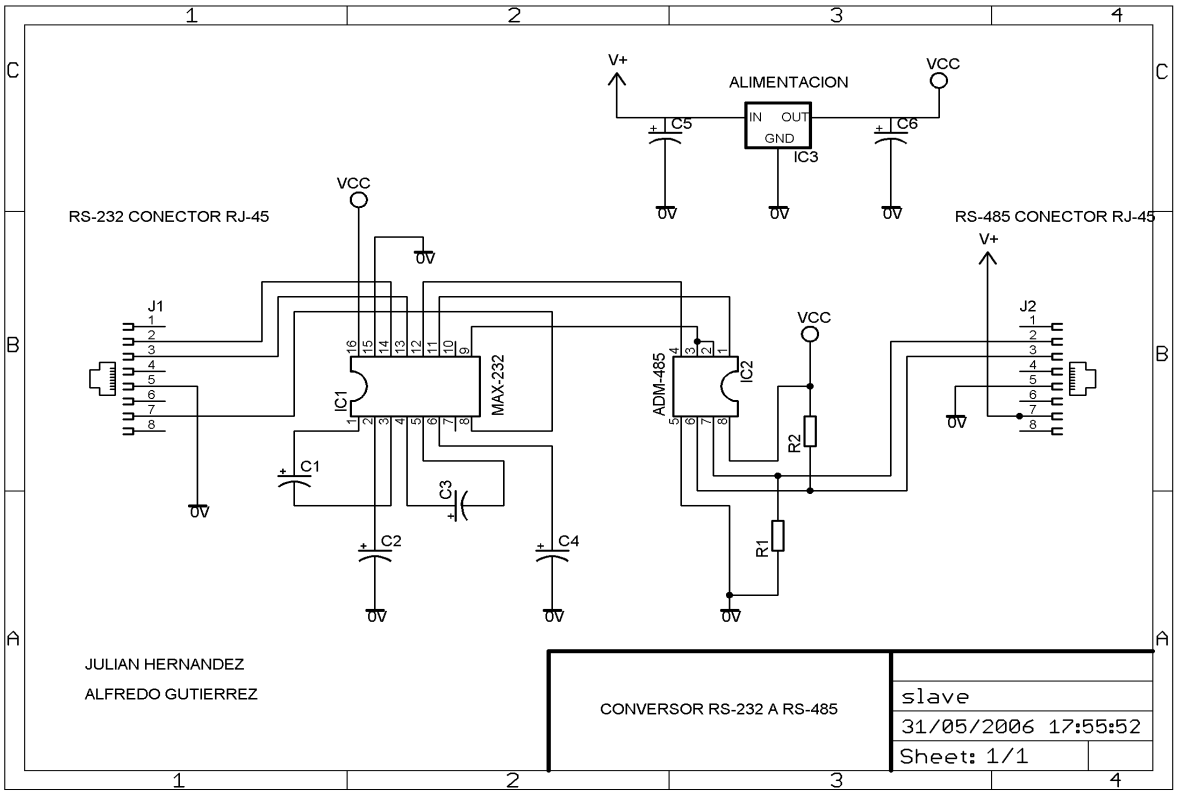


Figura 35. Circuito esquemático de las tarjetas esclavos.



Capitulo 4

APLICACIÓN EN LABVIEW

Para la implementación de un bus con protocolo de comunicación MODBUS, se desarrollaron dos aplicaciones en Labview. Una que realiza la función de Maestro de Bus, y una segunda que su función es de esclavo, ésta última será instalada en tres computadores diferentes, con el fin de simular un Bus con un computador principal (Maestro) y tres computadores (Esclavos). Para el desarrollo de estas dos aplicaciones, se utilizaron las librerías que son suministradas por la National Instrument, de uso exclusivo de aplicaciones en MODBUS.

4.1 Descripción de la librería modbus utilizada de National Instrument

INIT: Este VI se utiliza para darle las condiciones iniciales al puerto serial, para poder establecer comunicación.

Los parámetros que usa son:

- Selección del puerto
- Velocidad
- Control de flujo
- Paridad
- Time out
- Modo de transmisión

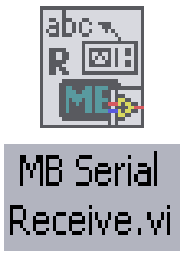
MB SERIAL TRANSMIT: Con este VI escribimos finalmente en el puerto, la trama que se ha desarrollado para cada función.

Parámetros que utiliza:

- Nombre del puerto
- Trama Modbus
- Parámetros del puerto
- Errores presentes

MB SERIAL RECEIVE: Con este VI, le damos las condiciones al puerto para recibir información del puerto.

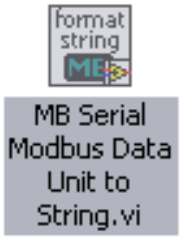
Parámetros que utiliza:



- Nombre del puerto
- Trama Modbus
- Parámetros del puerto
- Errores presentes
- Time Out
- Código de excepción

MB SERIAL DATA UNIT TO STRING: Convierte un formato de Unidad de datos Modbus a formato String, para luego visualizarlo en pantalla.

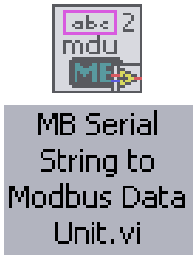
Parámetros que utiliza:



- Trama Modbus
- Parámetros del puerto

MB SERIAL STRING TO MODBUS DATA: Con éste VI podemos realizar un cambio de formato de Datos, String a Unidad de datos Modbus.

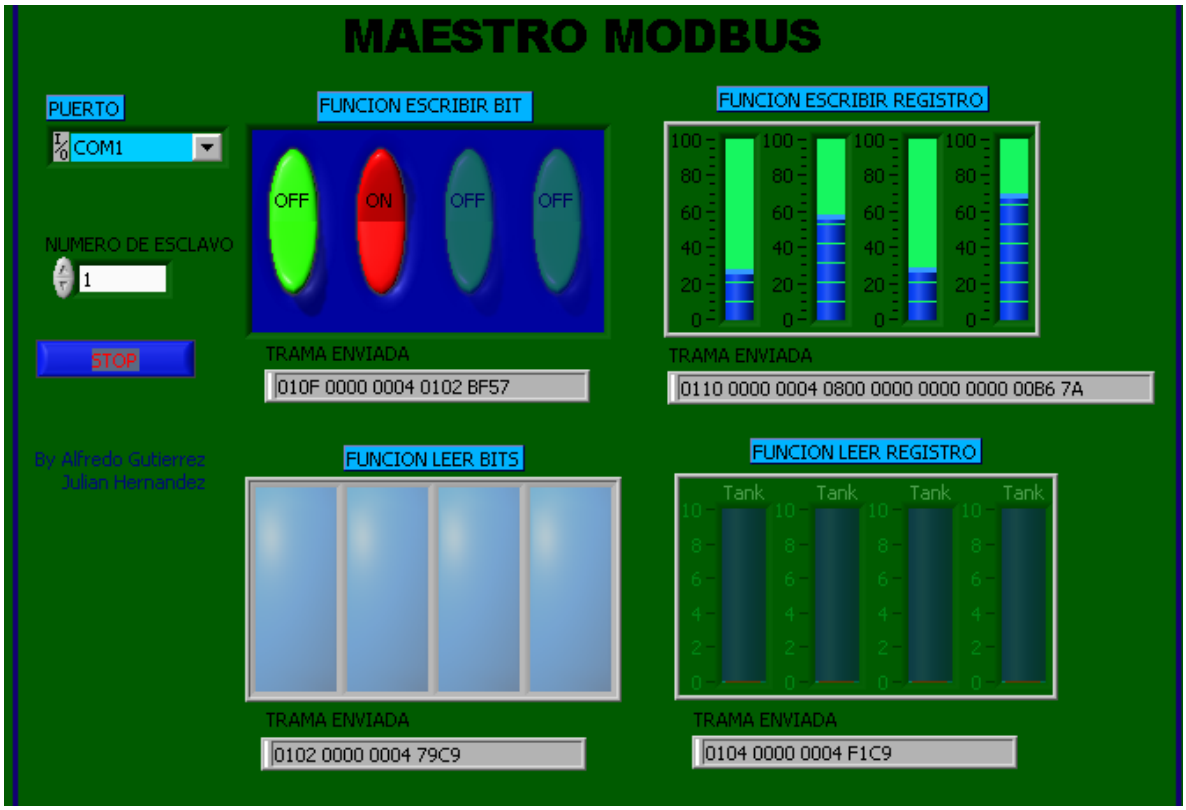
Parámetros que utiliza:



- Comando Modbus
- String

4.2 Maestro modbus

Figura 36. Aplicación Maestro Modbus.



En esta aplicación, se encuentran implementadas 4 funciones del protocolo MODBUS, realizándose secuencialmente una tras otra y cíclicamente hasta que el usuario seleccione el botón de STOP. Además cuenta con la posibilidad de seleccionar a que esclavo dentro del bus se le están enviando órdenes y tiene una ayuda visual para verificar la trama enviada por el puerto serial.

Para mayor versatilidad se puede seleccionar el puerto por el cual se desea enviar los datos, ya que no en todos los computadores es el mismo puerto.
Descripción de una de las tramas enviadas:

Figura 37. Trama enviada Función 02H.

Al comparar la trama enviada con el esquema que se vio en la figura 9, podemos decir lo siguiente:

Petición del Maestro

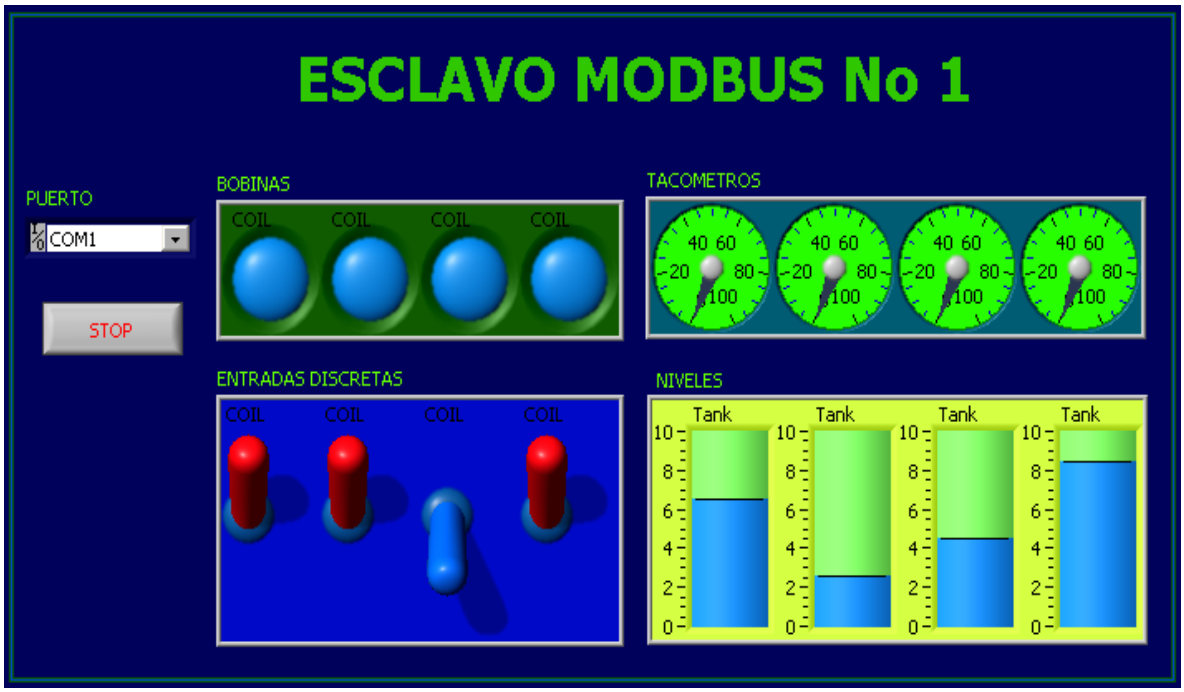
Trama enviada. Formato Hexadecimal.

0102 0000 0004 79C9

Nº esclavo: 01H, equivale al esclavo 1.
Función: 02H, esta especifica que la función es leer varias bobinas.
Dirección del 1er bit: 0000H
Número de Bits: 0004H, lo que nos indica que son 4 bits a leer.
CRC: 79C9, este valor es el resultado del calculo para la detección de errores.

4.3 Esclavo modbus

Figura 38. Aplicación esclavo Modbus.



Esta, aplicación nos representa un esquema básico de uno de los tres esclavos que se van a instalar en el bus. Presenta el estado de unas bobinas y unos registros que pueden ser modificados por el maestro y a la ves en este se pueden seleccionar estados de bobinas y valores de registros que solo el maestro los puede leer, estos pueden ser valores de procesos reales o simulados.

También permite seleccionar el puerto para establecer comunicación con el maestro.

Capitulo 7

CONCLUSIONES

- Se realizo el estudio del protocolo Modbus, mediante la implementación de cuatro funciones :
    - Función 01: Leer bits
    - Función 04: Leer registros
    - Función 05: Forzar bits
    - Función 06: Escribir registros

  Estas funciones permiten la manipulación de bits y de registros de 16 bits, con lo cual es posible su utilización para el monitoreo y control de variables en un proceso industrial, tal como se evidencia en la aplicación.

- Se desarrollo una tarjeta de conversión bidireccional de niveles de voltajes entre los estándares RS-232 a RS485, con control de flujo de datos, por medio del pin del puerto serial RTS. Este control de flujo, se implementó con el fin de evitar colisiones.

- Se realizo un montaje físico de un Bus de campo con arquitectura maestro-esclavo, utilizando un computador con la aplicación Maestro Modbus y tres computadores con la aplicación Esclavo Modbus. Si el estudiante quiere realizar este montaje vea el anexo 4.

- Se realizaron pruebas en la red física, y se obtuvieron resultados satisfactorios, comunicación estable, y funcionamiento correcto de cada una de las funciones implementadas.

- Algunos de los valores obtenidos en las pruebas son:

Tabla 6. Pruebas.

	NIVELES DE VOLTAJE	
Pin	**Salida del Puerto(volt)**	**Salida del receiver del MAX(volt)**
RTS	12.2	5
Tx	12.2	5
Rx	12.2	5

- Para lograr una comunicación estable se debió calcular los tiempos de envió de las tramas, por ejemplo.

  Sabiendo que la velocidad de transmisión es de 9600 bps, se tiene que el tiempo de transmisión de un bit es de

  0.10416ms. 0102 0000 0004 79C9

La trama Modbus que se quiere enviar es: Teniendo en cuenta que los datos se envían con 1 bit de comienzo, 8 bits de datos y 1 bit de parada (que es 1).

Se tiene que:

01(hex) →1dec. → **0**000000001

02(hex) →2dec. → **0**010000001

00(hex) →0dec. → **0**000000001

00(hex) →0dec. → **0**000000001

00(hex) →0dec. → **0**000000001

04(hex) →4dec. → **0**001000001

79(hex) →121dec. → **0**100111101

C9(hex) →201dec. → **0**100100111Como son 10 bits por carácter enviar los 10 toma 1.0416ms, y como son 8 caracteres tomaría 8.33ms.

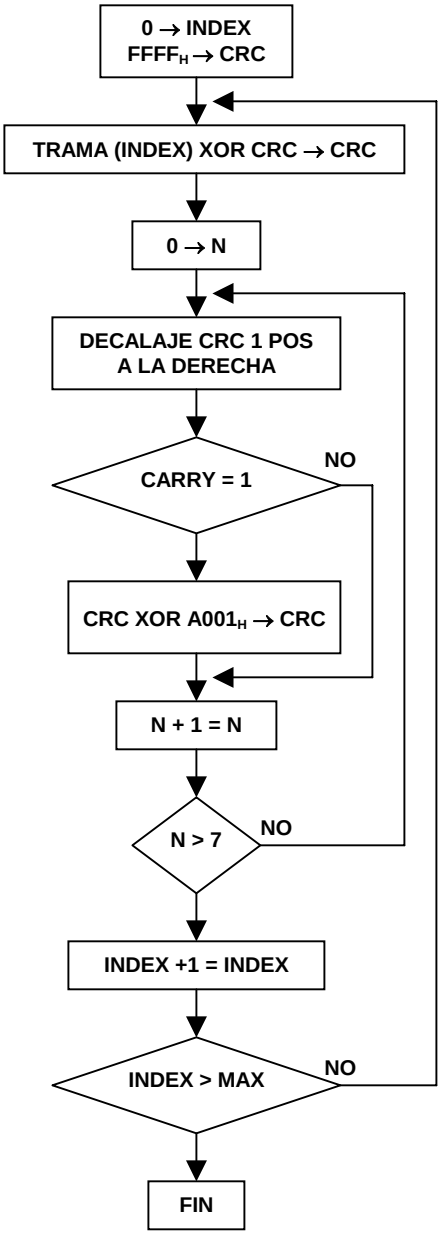
BIBLIOGRAFIA

- ARDILA, Pedro, CARREÑO, Sinle M. Modbus. Monitoreo de la red empleando Labview. Tesis de grado. Universidad Industrial de Santander. 2005.
- BALCELLS, J., ROMERAL, J.L., “Autómatas programables”, Marcombo. Serie Mundo Electrónico. Barcelona. 1997.
- CARO, D., “Automation network Selection”. ISA- The Instrumentation, Systems, and Automation Society. 2004.
- DUQUE, Jorge. “Procesador de comunicación Modbus. Implementación con microcontrolador”. Tesis de Maestría. Universidad Industrial de Santander 2006.

ANEXOS

ANEXO 1

Figura 39. Algoritmo de calculo del CRC.



Cálculo del Chequeo de Redundancia Cíclica CRC: El procedimiento para la generación del CRC es:

- *Paso 1*: Se cargar un registro de 16-bit con FFFF **h**. Llamando a éste el registro CRC.

- *Paso 2:* Se realiza una OR Exclusiva entre el primer byte del mensaje con el byte menos significativo del registro CRC, poniendo este resultado en el registro CRC.
- *Paso 3:* El registro CRC se desplaza hacia la izquierda una posición y se rellena con cero la posición del bit mas significativo.
- *Paso 4:* Se examina el bit que salió, si es igual a cero se repite el paso 3, si es igual a uno, se realiza una OR exclusiva entre el registro CRC y el valor fijo A001 **h** (1010 0000 0000 0001).
- *Paso 5:* Se repite el paso tres y cuatro hasta realizar 8 desplazamientos.
- *Paso 6:* Se repiten los pasos 2….5 para el siguiente byte del mensaje. Esto continúa para todos los bytes del mensaje.

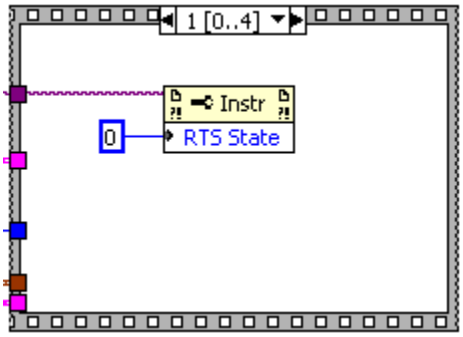
El resultado final del registro CRC contiene el valor del CRC.

- *Paso 7:* Cuando el CRC es colocado dentro del mensaje, se debe tener en cuenta que se transmite primero el byte de menor orden seguido del byte de mayor orden.

ANEXO 2

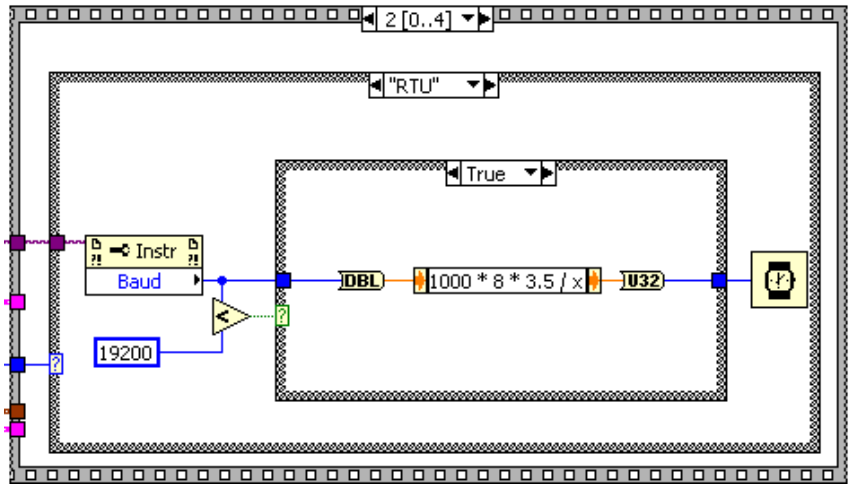
CAMBIO DE ESTADO DE LA LÍNEA RTS EN EL PROGRAMA

Figura 40. Estado de la función Instr. En 0.



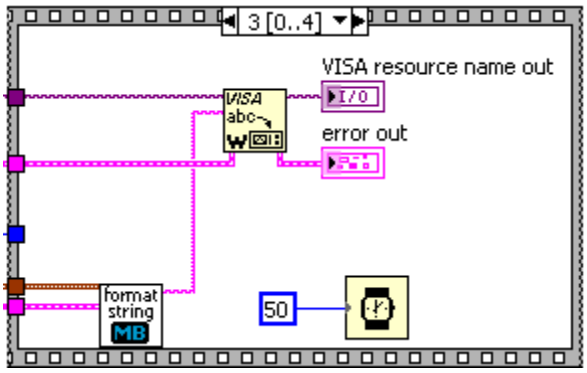
La función que se utiliza esta en la siguiente ubicación: clic derecho en el block diagram / all functions / Instrument I/O / Visa / Visa Advanced / Property Node. Y se configura con los parámetros que se muestra (estado línea RTS) haciendo clic derecho sobre la función, y por ultimo se agrega una constante con el valor deseado.

Figura 41. Retardo para escribir en el puerto.



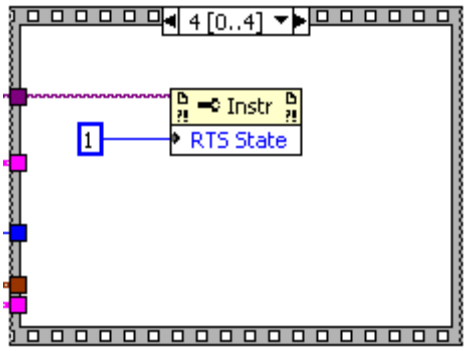
En esta secuencia se agrega un retardo antes de escribir al puerto igual a 3.5 el tiempo de un carácter ASCII.

Figura 42. Escritura de datos en el puerto.



En esta secuencia se escribe en el puerto la trama que se va a enviar.

Figura 43. Estado de la función Instr. En 1.



En esta secuencia se coloca la función en 1, que corresponde a un voltaje positivo en la línea RTS del puerto RS 232.

ANEXO 3

EJEMPLOS DE CALCULO DE RESISTENCIAS DE BIAS

Ejemplo 1.

Una red de 10 nodos con dos terminadores de 120Ω.

Como cada nodo tiene una impedancia de entrada de 12KΩ (Ver Tabla 5), los diez nodos en paralelo tendrán una impedancia equivalente de 1200Ω. Las dos resistencias de terminación en paralelo añaden una carga de 60Ω, lo que resulta en una carga total (todas en paralelo) 57Ω. Con lo que se ve claramente que los terminadores son los responsables de la mayor parte de la carga. Para mantener una tensión mínima de 200mV entre las líneas A y B, necesitamos una corriente de bias a través de la carga de unos 3.5mA. Para crearla a partir de los 5V de alimentación, necesitamos una resistencia total de 1428Ω o menos, si le restamos los 57Ω que ya están presentes en el circuito como parte de la carga, nos quedan 1371Ω. Poniendo la mitad de este valor como resistencia de Pull up y la otra mitad como resistencia de Pull down, nos quedan resistencias de bias con un valor máximo de 685.5Ω.

Ejemplo 2.

Una red de 32 nodos sin terminadores.

Cada nodo de la red tiene una impedancia de 12KΩ, luego los 32 nodos en paralelo dan una impedancia de 375Ω, Para mantener una tensión de 200mV a través de 375Ω, se necesita una corriente de 0.53mA, Para generar este valor de corriente a partir de los 5V de alimentación necesitamos una resistencia de 9375Ω como máximo, como los 375Ω de los driver ya están presentes restaran unos 9000Ω o menos para las resistencias de bias. Obsérvese que cuando no hay terminadores la corriente de bias necesaria es muy pequeña.

ANEXO 4

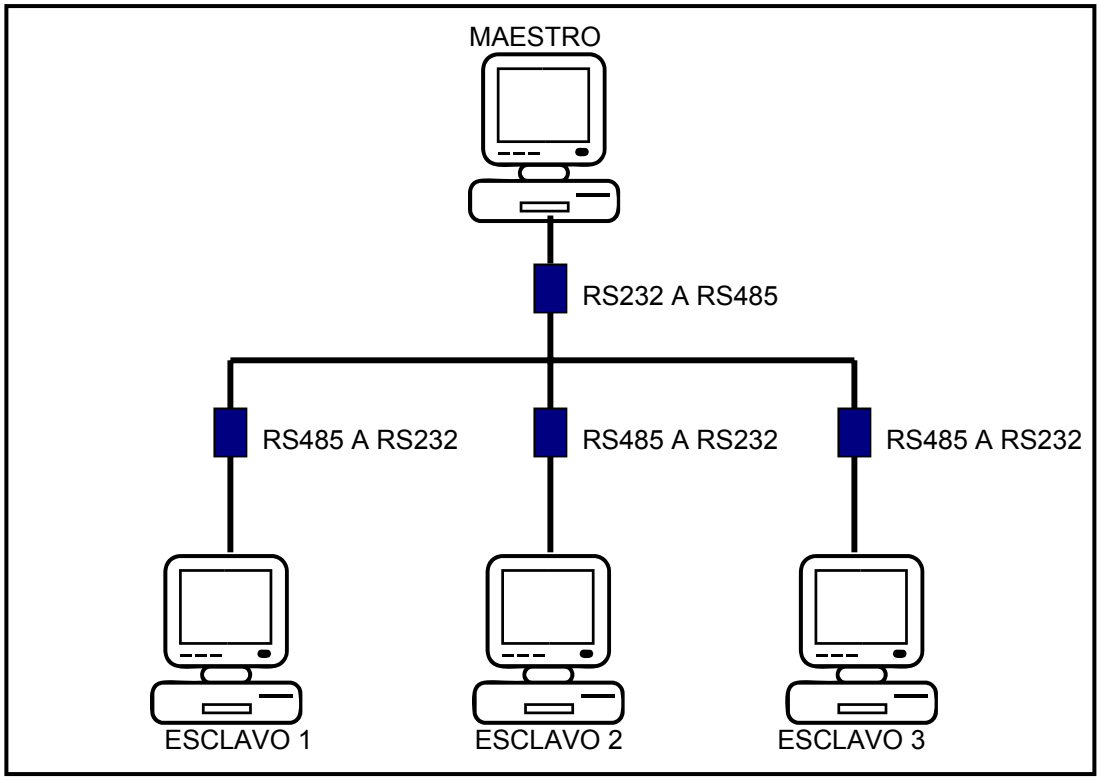
CONEXIÓN DE LA RED MODBUS

Recursos:

- Programa Maestro en Labview.
- Tres programas esclavos en Labview.
- Cables UTP con conectores RJ45.
- Cuatro conectores de RS232 a RJ45.
- Cuatro conversores de RS232 a RS485.
- Librerías Modbus de la *National Instruments*.

Para la conexión de la red Modbus usted debe contar con mínimo dos PCs, si cuenta con los cuatro PCs usted podrá conectar el respectivo maestro y los tres esclavos de lo contrario usted solo podrá conectar el maestro y uno de los esclavos. El esquema de la conexión que usted va a implementar se muestra a continuación en la figura 44.

Figura 44. Conexión de la red Modbus.



Para la conexión de la red verifique que sus PCs tengan Labview 7.1, ya que las librerías requieren de esta versión de Labview.

- Primero instale la librería Modbus de la National Instrument en cada uno de los computadores.
- En el PC que ha elegido como Maestro abra el archivo del Maestro que se encuentra en las librerías.
- En el PC o PCs restantes abra los archivos de los Esclavos.
- Conecte la tarjeta Maestro (la que contiene la mol de alimentación principal) al PC maestro, verifique que este puesta del lado correcto (esto es de RS232 a RS485)

- Por medio de los cables conecte las tarjetas restantes de modo correcto (esto es de RS485 a RS232), como se muestra en la figura 31.
- Antes de correr los programas verifique que su computador tenga el puerto configurado (COM1).
- En el programa Maestro y Esclavo seleccione el puerto de comunicaciones que usted previamente configuro en sus PCs y seleccione el esclavo en el cual usted desea escribir (Bobina o registro).
- Ponga en marcha los programas y verifique cada una de las funciones
- Por último analice cada una de las tramas Modbus.

Nota: Para visualizar las funciones de lectura de Bits y registros en el maestro, recuerde que debe cambiar estos parámetros en el esclavo.