

**INTERFAZ GRAFICA PARA LA SUPERVISION DE DATOS DE UN MOTOR MEDIANTE EL  
PROTOCOLO MODBUS TCP-IP**

**ALDASANIA CUELLO ROMERO  
NEVER LENIN MORA VERGARA**

**UNIVERSIDAD PONTIFICIA BOLIVARIANA  
BUCARAMANGA**

**2011**

**INTERFAZ GRAFICA PARA LA SUPERVISION DE DATOS DE UN MOTOR MEDIANTE EL  
PROTOCOLO MODBUS TCP-IP**

**ALDASANIA CUELLO ROMERO  
NEVER LENIN MORA VERGARA**

**Monografía de grado para optar al título de Especialista en Control e Instrumentación  
Industrial**

**Director  
Juan Carlos Villamizar  
MSc En Potencia Eléctrica**

**UNIVERSIDAD PONTIFICIA BOLIVARIANA  
BUCARAMANGA**

**2011**

## Lista de Figuras

Figura 1. Trama genérica del mensaje según el código empleado.....	8
Figura 2 Esquema de encapsulamiento en Modbus/TCP.....	14
Figura 3. Conexión del variador de velocidad en forma manual.....	21
Figura 4. Red implementada.....	21
Figura 5. Cable Modbus para la conexión PLC-Variador.....	22
Figura 6. Esquema de configuración de red para el variador y el PLC.....	22
Figura 7. Configuración de parámetros para la comunicación.....	23
Figura 8. Diagrama para la configuración del variador.....	24
Figura 9. Función de inicialización del variador.....	24
Figura 10. Configuración de parámetros iniciales.....	25
Figura 11. Definición de valores para el encendido, apagado y cambio en el sentido de giro.	25
Figura 12. Función para establecer la velocidad en sentido horario.....	25
Figura 13. Función para establecer la rotación en sentido anti horario.....	26
Figura 14. Función para detener el proceso.....	26
Figura 15. Función para establecer la velocidad.....	26
Figura 16. Función para establecer eliminar los errores del búfer.....	27
Figura 17. Lista del programa.....	27
Figura 18. Asignación de símbolos a entradas.....	28
Figura 19. Interfaz .....	29
<i>Figura 20. Diagrama de flujo de la interfaz.....</i>	<i>30</i>
<i>Figura 21. Diagrama de flujo Escribir Registro.....</i>	<i>31</i>
<i>Figura 22. Diagrama de flujo Leer Registro.....</i>	<i>32</i>
<i>Figura 23. Propiedades del elemento Winsock1.....</i>	<i>33</i>
<i>Figura 24. Propiedades del Timer.....</i>	<i>34</i>

### ***Lista de tablas***

Tabla 1. Estructura del prefijo de Modbus/TCP.....	11
Tabla 2. Estructura de mensajes en Modbus/TCP.....	12
Tabla 3. Funciones básicas y códigos de operación.....	12
Tabla 4. Datos configurados en el variador.....	24
Tabla 5. Constantes de estado definidas para winsock.....	33

## TABLA DE CONTENIDO

1. INTRODUCCIÓN .....	7
2.1 MODBUS.....	8
2.1.2 Estructura de la red .....	8
2.1.3 Protocolo .....	8
2.2 MODBUS TCP/IP .....	9
2.2.1 Características del protocolo Modbus TCP/IP.....	9
2.2.2 Estructura del protocolo Modbus TCP/IP .....	10
2.2.3 Esquema de encapsulamiento Modbus tcp/IP .....	14
2.3 VARIADOR DE VELOCIDAD .....	14
2.3.1 Ventajas de la utilización del Variador de Velocidad en el arranque de motores asíncronos. ....	15
2.3.2 Principales funciones de los variadores de velocidad electrónicos .....	15
2.3.3 Conexión del variador de velocidad. ....	20
3. IMPLEMENTACION DE LA RED .....	21
3.1 CONFIGURACION DE LA CONEXIÓN MODBUS ENTRE EL PLC Y EL VARIADOR DE VELOCIDAD.....	22
3.2 CONFIGURACION DEL VARIADOR PARA LA COMUNICACIÓN MODBUS.....	23
3.3 PROGRAMACION DEL CODIGO PARA LA RECEPCIÓN Y ENVIO DE DATOS .....	24
4. INTERFAZ GRAFICA EN VISUAL BASIC.....	28
4.1 CONCEPTOS PREVIOS .....	28
4.2 DESCRIPCIÓN DEL FUNCIONAMIENTO, ELEMENTOS Y CODIGO DE LA INTERFAZ CLIENTE MODBUS TCP/IP .....	29
4.2.1 Funcionamiento de la Interfaz .....	29
4.2.2 Descripción de elementos de la interfaz.....	33
4.2.3 Descripción de las funciones del código .....	35
CONCLUSIONES .....	39
BIBLIOGRAFIA.....	40

## RESUMEN GENERAL DE TRABAJO DE GRADO

**TITULO:** INTERFAZ GRAFICA PARA LA SUPERVISION DE DATOS DE UN MOTOR MEDIANTE EL PROTOCOLO MODBUS TCP-IP

**AUTORES:** Aldasania Cuello Romero  
Never Lenin Mora Vergara

**FACULTAD:** Esp. en Control e Instrumentación Industrial

**DIRECTOR:** Juan Carlos Villamizar

### RESUMEN

Esta monografía presenta una interfaz programada en Visual Basic para la supervisión de datos del plc TWD 07 de Schneider Electronics mediante el protocolo Modbus TCP/IP, Este a su vez se encarga de controlar un variador de velocidad conectado a un motor siemens de 0.45 KW. Con esta interfaz y el programa cargado al PLC se puede encender, apagar, ajustar la velocidad en el variador e invertir el sentido de giro del motor. El lenguaje en el cual está programada la interfaz es Visual Basic, el cual es sencillo y de amplio uso, esta se comunica con el PLC usando el protocolo Modbus TCP/IP donde se utiliza la biblioteca dinámica de funciones (DLL) winsock, esta dll soporta la comunicación TCP/IP mediante el envío y recepción de sockets, siendo sobre estos sockets donde se envía la trama Modbus. En la interfaz se dispone de controles sencillos para leer y escribir los registros del PLC, pudiendo así realizar las acciones de control básicas descritas en el párrafo anterior.

#### Key words:

Interfaz, Modbus TCP/IP, Variador de velocidad

#### THESIS DIRECTOR'S APPROVAL

## GENERAL SUMMARY OF THESIS

**TITLE:** INTERFAZ GRAFICA PARA LA SUPERVISION DE DATOS DE UN MOTOR MEDIANTE EL PROTOCOLO MODBUS TCP-IP

**AUTHORS:** Aldasania Cuello Romero  
Never Lenin Mora Vergara

**FACULTY:** Esp. en Control e Instrumentación Industrial

**DIRECTOR:** Juan Carlos Villamizar

## SUMMARY

This paper presents an interface programmed in Visual Basic for monitoring Schneider Electronics TWD07 plc's data, via Modbus TCP / IP. In addition, the plc is responsible for controlling a speed driver controller connected to a Siemens of 0.45 KW. With this interface and PLC's program loaded we can turn on/off, adjust the speed in the drive and reverse the direction of rotation of the motor.

The language in which the interface is programmed is Visual Basic, which is simple and widely used, it communicates with the PLC using the Modbus TCP / IP which uses the dynamic library of functions (DLL) winsock, this dll supports TCP / IP by sending and receiving sockets, being on these sockets which sends the Modbus frame. The interface has simple controls to read and write PLC's registers, and can perform basic control actions described above.

### Key words:

Interfaz, Modbus TCP/IP, Variador de velocidad

### THESIS DIRECTOR'S APPROVAL

## 1. INTRODUCCIÓN

En el campo de la instrumentación, es muy importante el buen funcionamiento de la comunicación entre los sensores, actuadores y controladores, la información que se trasmite entre estos debe llegar a su destino sin ser alterada. Para que esto ocurra en ese sentido se han desarrollado muchos protocolos de comunicación siendo el protocolo Modbus TCP/IP uno de ellos. Modbus TCP/IP es una extensión del protocolo Modbus que se utiliza sobre la capa de transporte TCP/IP. Este se basa en la arquitectura maestro esclavo, es un protocolo abierto de bajo costo y simplicidad, convirtiéndolo estas características en un estándar industrial de facto.

Actualmente la industria ofrece muchos tipos de programas supervisores aunque el acceso a estos en la mayoría de las veces es limitado para las pequeñas empresas debido a sus altos costos. En este trabajo de investigación se emplea una interfaz gráfica utilizando Visual Basic, la cual es una herramienta de programación generalizada. En este caso específico la supervisión de datos y control de un motor por medio de un PLC y un variador de velocidad.



## 2. MARCO TEORICO

### 2.1 MODBUS

Modbus es un protocolo de enlace, debido a esto puede implementarse con diversos tipos de redes físicas donde generalmente cada fabricante suele suministrar un software de aplicación propio. Este fue desarrollado por Modicom y en la actualidad es de acceso libre, una característica que lo ha convertido en un protocolo ampliamente utilizado.

Las características principales de Modbus se describen a continuación:

#### 2.1.2 Estructura de la red

*Medio Físico:* El medio físico de conexión puede ser un bus semidúplex (half duplex) (RS-485 o fibra óptica) o dúplex (full duplex) (RS-422, BC 0-20mA o fibra óptica). La comunicación es asíncrona y las velocidades de transmisión previstas van desde los 75 baudios a 19.200 baudios. La máxima distancia entre estaciones depende del nivel físico, pudiendo alcanzar hasta 1200 m sin repetidores.

*Acceso al Medio:* La estructura lógica es del tipo maestro-esclavo, donde el número máximo de esclavos es de 63 y 1 maestro para un total de 64 elementos.

Los intercambios de mensajes pueden ser de dos tipos:

- Intercambios punto a punto, que comparten siempre dos mensajes: una demanda del maestro y una respuesta del esclavo (puede ser simplemente un reconocimiento).
- Mensajes difundidos. Estos consisten en una comunicación unidireccional del maestro a todos los esclavos. Este tipo de mensajes no tiene respuesta por parte de los esclavos y se suelen emplear para mandar datos comunes de configuración, reset, etc.

#### 2.1.3 Protocolo

La codificación de datos dentro de la trama puede hacerse en modo ASCII o puramente binario, según el estándar RTU (Remote Transmission Unit). En cualquiera de los dos casos, cada mensaje obedece a una trama que contiene cuatro campos principales, según se muestra en la figura 1. La única diferencia estriba en que la trama ASCII incluye un carácter de encabezamiento («:»=3A) y los caracteres CR y LF al final del mensaje. Otra diferencia entre

estos dos es la forma en la cual se calcula la palabra de chequeo de errores (CRC), el formato RTU emplea una fórmula polinómica en vez de la simple suma en módulo 16 como lo hace el formato ASCII. A continuación muestra cada una de las dos tramas y la posición de los elementos en la misma:

: (3AH)	Nº Esclavo (00-3F <sub>H</sub> )	Código de Operación	Subfunciones, Datos	LRC(16) H L	CR (0D <sub>H</sub> )	LF (0A <sub>H</sub> )
------------	--	------------------------	---------------------	----------------	--------------------------	--------------------------

**Codificación ASCII**

Nº Esclavo (00-3F <sub>H</sub> )	Código de Operación	Subfunciones, Datos	CRC(P16) H L
--	------------------------	---------------------	-----------------

**Codificación RTU**

*Figura 1. Trama genérica del mensaje según el código empleado. ¡Error! No se encuentra el origen de la referencia. ¡Error! No se encuentra el origen de la referencia.<sup>1</sup>*

## 2.2 MODBUS TCP/IP

ModbusTCP/IP es un protocolo diseñado para comunicar equipos industriales sobre una red, utilizando la capa de transporte TCP/ IP. Fue diseñado para la supervisión y el control de equipo de automatización. Específicamente, el protocolo cubre el uso de mensajes MODBUS en un entorno internet o intranet usando los protocolos TCP/IP.

MODBUS® TCP/IP es un estándar industrial muy utilizado debido a su simplicidad, bajo coste, necesidades mínimas en cuanto a componentes de hardware, y sobre todo, que se trata de un protocolo abierto. Cualquier sistema, computador o microprocesador con pila de protocolos TCP/IP puede usar Modbus/TCP y desde allí se puede hacer monitoreo o reparaciones. Posee un alto desempeño, limitado generalmente por la capacidad del sistema operativo del computador para comunicarse.

### 2.2.1 Características del protocolo Modbus TCP/IP

- El protocolo ModBus TCP/IP está orientado a conexión. Las operaciones de programación esperan una comunicación orientada a la conexión, es decir, las máquinas de origen y de destino establecen un canal de comunicaciones antes de

<sup>1</sup> Comunicaciones Industriales - Universidad Politécnica de Cartagena  
[http://www.dte.upct.es/personal/manuel.jimenez/docencia/GD6\\_Comunic\\_Ind/pdfs/Tema%207.pdf](http://www.dte.upct.es/personal/manuel.jimenez/docencia/GD6_Comunic_Ind/pdfs/Tema%207.pdf).

transferir datos. Esta conexión, puede llevar múltiples transacciones independientes. En adición, TCP permite establecer un gran número de conexiones.

- concurrentes, de este modo el cliente (maestro) puede ya sea re-usar una conexión previamente establecida o crear una nueva, en el momento de realizar una transacción de datos.
- La codificación de datos ModBus usa el formato “big-endian”, en el cual el byte más significativo se encuentra primero.  
Por ejemplo: 16 bits 0x1234 seria, 0x12 0x34
- Para los códigos de función que llevan una cantidad variable de datos en la solicitud ó respuesta, la porción de datos estará precedida por un campo que representa el número de bytes que siguen. Cuando MODBUS es llevado sobre TCP, la información de longitud se adiciona en el prefijo (o encabezado), para permitir al receptor reconocer los límites del mensaje. Para los códigos de función que llevan una solicitud ó respuesta con una longitud fija, solo es suficiente el código de función.

### 2.2.2 Estructura del protocolo Modbus TCP/IP

La estructura general del protocolo MODBUS TCP/IP, o la forma de encapsular los datos por parte de este en caso de una petición o una respuesta Modbus llevada sobre una red MODBUS TCP/IP, se caracteriza por la manera en la que tanto el cuerpo de la pregunta como la respuesta, desde el código de función hasta el final de la porción de datos, poseen la misma disposición y el mismo significado que en las otras variantes del protocolo MODBUS como lo son:

- MODBUS serial con codificación ASCII
- MODBUS serial con codificación RTU
- MODBUS PLUS

Las únicas diferencias en esos otros casos son la especificación de los delimitadores inicial y final del mensaje (*framing*), el patrón de chequeo de error y la interpretación de la dirección.

Las solicitudes normalmente son enviadas en forma *half-duplex* (los datos pueden viajar en cualquier dirección, pero no en forma simultánea) sobre una conexión dada. Esto implica que,

no hay beneficio en enviar solicitudes adicionales sobre una única conexión mientras una respuesta está pendiente. Sin embargo, los dispositivos que desean obtener altas tasas de transferencia pueden establecer múltiples conexiones TCP al mismo destino.

Esta técnica de consulta/respuesta encaja perfectamente con la naturaleza Maestro/Esclavo de Modbus, añadido a la ventaja del determinismo que las redes Ethernet conmutadas ofrecen a los usuarios en la industria. El empleo del protocolo abierto Modbus con TCP proporciona una solución para la gestión desde unos pocos a decenas de miles de nodos.

El campo dirección esclavo de MODBUS es reemplazado por un byte identificador de unidad el cual puede ser usado para comunicar a través de dispositivos tales como puentes y *gateways*, los cuales usan una dirección IP única para soportar múltiples unidades terminales independientes.

Los mensajes de solicitud y respuesta en Modbus/TCP poseen un prefijo ó encabezado compuesto por seis bytes como se aprecia en la tabla 2.

Ref	Ref	00	00	len
-----	-----	----	----	-----

Tabla 1. Estructura del prefijo de Modbus/TCP.<sup>2</sup>

Los elementos “ref ref” iniciales son los dos bytes del campo referencia de transacción, un número que no tiene valor en el servidor, pero son copiados literalmente desde la solicitud de respuesta. Este campo se utiliza para que un cliente Modbus/TCP pueda establecer simultáneamente múltiples conexiones con diferentes servidores y pueda identificar cada una de las transacciones.

El tercer y cuarto campo del prefijo representa el identificador de protocolo, un número el cual debe ser establecido a cero. El término “len” especifica el número de bytes que siguen. La longitud es una cantidad de dos bytes, pero el byte alto se establece a cero ya que los mensajes son menores que 256.

---

<sup>2</sup> <http://www.monografias.com/trabajos75/red-modbus-tcp-ordenador/red-modbus-tcp-ordenador2.shtml>

Posición del Byte	Significado
Byte 0	Identificador de transacción. Copiado por el servidor - normalmente 0.
Byte 1	Identificador de transacción. Copiado por el servidor - normalmente 0.
Byte 2	Identificador de protocolo = 0.
Byte 3	Identificador de protocolo = 0.
Byte 4	Campo de longitud (byte alto) = 0. los mensajes son menores a 256.
Byte 5	Campo de longitud (byte bajo). Número de bytes siguientes.
Byte 6	Identificador de unidad (previamente Dirección esclavo).
Byte 7	Código de función MODBUS.
Byte 8 en adelante	Los datos necesarios

*Tabla 2. Estructura de mensajes en Modbus/TCP.<sup>3</sup>*

En la siguiente tabla se muestra un conjunto de funciones básicas las cuales conforman uno de los cuatro campos principales que conforman un mensaje. Este código de función se encuentra en el byte 7 de la estructura de mensaje Modbus.

<b>Función</b>	<b>Código</b>	<b>Tarea</b>
<b>0</b>	00 <sub>H</sub>	Control de estaciones esclavas
<b>1</b>	01 <sub>H</sub>	Lectura de <i>n</i> bits de salida o internos
<b>2</b>	02 <sub>H</sub>	Lectura de <i>n</i> bits de entradas
<b>3</b>	03 <sub>H</sub>	Lectura de <i>n</i> bits de palabras de salidas o internos

4	04 <sub>H</sub>	Lectura de $n$ palabras de entradas
5	05 <sub>H</sub>	Escritura de un bit
6	06 <sub>H</sub>	Escritura de una palabra
7	07 <sub>H</sub>	Lectura rápida de 8 bits
8	08 <sub>H</sub>	Control de contadores de diagnósticos número 1 a 8
9	09 <sub>H</sub>	No utilizado
10	0A <sub>H</sub>	No utilizado
11	0B <sub>H</sub>	Control del contador de diagnósticos número 9
12	0C <sub>H</sub>	No utilizado
13	0D <sub>H</sub>	No utilizado
14	0E <sub>H</sub>	No utilizado
15	0F <sub>H</sub>	Estructura de $n$ Bits
16	10 <sub>H</sub>	Escritura de $n$ palabras

Tabla 3. Funciones básicas y códigos de operación

Por ejemplo, la estructura del mensaje utilizada en el código de la interfaz propuesta en esta monografía es la siguiente:

```
StartLow = Val(Text2.Text) Mod 256
StartHigh = Val(Text2.Text) \ 256
LengthLow = Val(Text3.Text) Mod 256
LengthHigh = Val(Text3.Text) \ 256
```

```
MbusQuery(0) = 0 } Identificadores de transacción.
MbusQuery(1) = 0 }
MbusQuery(2) = 0 } Identificadores de protocolo
MbusQuery(3) = 0 }
MbusQuery(4) = 0 → Mensajes menores a 256
MbusQuery(5) = 6 → Numero de bytes siguientes
MbusQuery(6) = 1 → Identificador de unidad(esclavo 1)
MbusQuery(7) = 3 → Código de función Modbus
MbusQuery(8) = StartHigh }
MbusQuery(9) = StartLow } Cantidad de datos a leer o escribir
MbusQuery(10) = LengthHigh } Representados de la forma big- endian
MbusQuery(11) = LengthLow }
```

Donde Text2.Text es igual al valor de donde se empiezan a leer los registros yText3.Text es la cantidad de datos que deseo leer.

### 2.2.3 Esquema de encapsulamiento Modbus tcp/IP

Modbus/TCP simplemente encapsula una trama Modbus en un segmento TCP. Este a su vez proporciona un servicio orientado a conexión, lo que significa que toda consulta espera una respuesta. Esto se puede ilustrar en la siguiente figura.

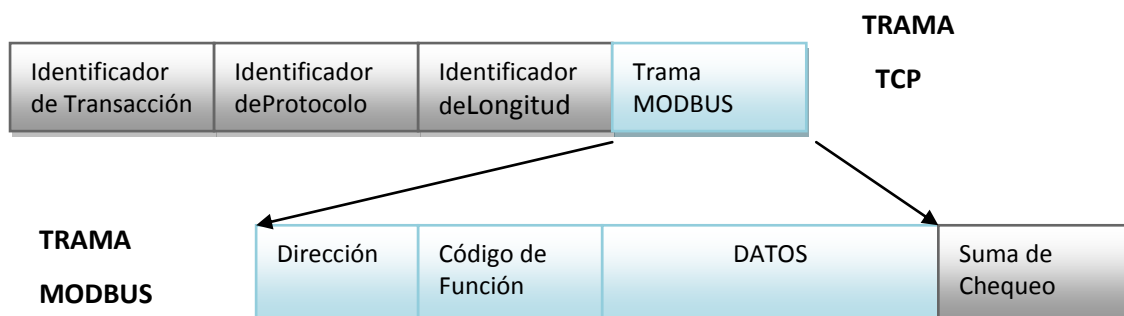


Figura 2. Esquema de encapsulamiento en Modbus/TCP.<sup>3</sup>

Esta técnica de consulta/respuesta encaja perfectamente con la naturaleza Maestro/Esclavo de Modbus.

## 2.3 VARIADOR DE VELOCIDAD<sup>4</sup>

Los variadores de velocidad son dispositivos que permiten variar la velocidad de los motores asíncronos trifásicos, convirtiendo las magnitudes fijas de frecuencia y tensión de red en magnitudes variables.

Los variadores se utilizan para:

- Dominio de par y la velocidad
- Regulación sin golpes mecánicos
- Movimientos complejos

<sup>3</sup> Vid nota 1

<sup>4</sup>La información sobre el variador de velocidad fue tomado de las guías de laboratorio de redes industriales UPB bucaramanga.

- Mecánica delicada

14

### **2.3.1 Ventajas de la utilización del Variador de Velocidad en el arranque de motores asíncronos.**

- El variador de velocidad no tiene elementos móviles, ni contactos.
- La conexión del cableado es muy sencilla.
- Permite arranques suaves, progresivos y sin saltos.
- Controla la aceleración y el frenado progresivo.
- Limita la corriente de arranque.
- Permite el control de rampas de aceleración y deceleración regulables en el tiempo.
- Consigue un ahorro de energía cuando el motor funcione parcialmente cargado, con acción directa sobre el factor de potencia
- Puede detectar y controlar la falta de fase a la entrada y salida de un equipo. Protege al motor.
- Puede controlarse directamente a través de un autómatas o microprocesador.
- Se obtiene un mayor rendimiento del motor.
- Nos permite ver las variables (tensión, frecuencia, r.p.m, etc...).

### **2.3.2 Principales funciones de los variadores de velocidad electrónicos**

#### ***Aceleración controlada***

La aceleración del motor se controla mediante una rampa de aceleración lineal o en «S».

Generalmente, esta rampa es controlable y permite por tanto elegir el tiempo de aceleración adecuado para la aplicación.



Una rampa de aceleración es la forma como un motor se arranca y se detiene. El tiempo en el cual el motor va de su estado de reposo a una velocidad determinada se llama aceleración, y el tiempo en el cual se va de una velocidad determinada hasta que el rotor se queda quieto se llama desaceleración. Hay varias formas para lograr este frenado y cada una de ellas tiene sus efectos sobre la carga.

Rampa lineal: este tipo de rampa acelera y desacelera de forma proporcional al tiempo transcurrido. Aparentemente es la más confortable pero al llegar al momento de velocidad constante se pueden presentar perturbaciones en cintas transportadoras.

Rampa en forma de U: para minimizar las perturbaciones al momento de adquirir velocidad constante se han implementado métodos en forma de U y S las cuales deben seleccionarse de acuerdo a la carga que se vaya a mover.

### ***Variación de velocidad***

Un variador de velocidad no puede ser al mismo tiempo un regulador. En este caso, es un sistema, rudimentario, que posee un mando controlado mediante las magnitudes eléctricas del motor con amplificación de potencia, pero sin bucle de realimentación: es lo que se llama «en bucle abierto».

La velocidad del motor se define mediante un valor de entrada (tensión o corriente) llamado consigna o referencia. Para un valor dado de la consigna, esta velocidad puede variar en función de las perturbaciones (variaciones de la tensión de alimentación, de la carga, de la temperatura). El margen de velocidad se expresa en función de la velocidad nominal.

Para variar la velocidad del motor lo que hace el variador es mantener una frecuencia constante. Si se quiere variar la velocidad el cambio no es brusco sino que cambia con un tiempo estipulado en los parámetros del variador.

### ***Regulación de la velocidad***

Un regulador de velocidad es un dispositivo controlado. Posee un sistema de mando con amplificación de potencia y un bucle de alimentación: se denomina, «bucle abierto».

La velocidad del motor se define mediante una consigna o referencia.

El valor de la consigna se compara permanentemente con la señal de alimentación, imagen de la velocidad del motor. Esta señal la suministra un generador tacométrico o un generador de impulsos colocado en un extremo del eje del motor.

Si se detecta una desviación como consecuencia de una variación de velocidad, las magnitudes aplicadas al motor (tensión y/o frecuencia) se corrigen automáticamente para volver a llevar la velocidad a su valor inicial.

Gracias a la regulación, la velocidad es prácticamente insensible a las perturbaciones.

La precisión de un regulador se expresa generalmente en % del valor nominal de la magnitud a regular.

### ***Deceleración controlada***

Cuando se desconecta un motor, su deceleración se debe únicamente al par resistente de la máquina (deceleración natural). Los arrancadores y variadores electrónicos permiten controlar la deceleración mediante una rampa lineal o en «S», generalmente independiente de la rampa de aceleración.

Esta rampa puede ajustarse de manera que se consiga un tiempo para pasar de la velocidad de régimen fijada a una velocidad intermedia o nula:

- Si la deceleración deseada es más rápida que la natural, el motor debe desarrollar un par resistente que se debe sumar al par resistente de la máquina; se habla entonces de frenado eléctrico, que puede efectuarse reenviando energía a la red de alimentación, o disipándola en una resistencia de frenado.

- Si la deceleración deseada es más lenta que la natural, el motor debe desarrollar un par motor superior al par resistente de la máquina y continuar arrastrando la carga hasta su parada.

### ***Inversión del sentido de marcha***

Al igual que con un funcionamiento normal para invertir el sentido de giro de un motor de inducción, con un variador electrónico se le debe invertir el sentido de rotación al campo magnético en el estator. Solo que aquí no se hace con contactores ni con switches externos, sino simplemente se le cambia la secuencia de encendido a los transistores del puente en H trifásico. La gran diferencia es que con contactores se le puede cambiar repentinamente el sentido de giro al motor, en un variador esto no se puede conseguir porque se dañan los transistores del puente en H. Para evitar el daño, los variadores deben primero aplicar uno de los frenados descritos anteriormente, esperar que la velocidad llegue a cero y después invertirle la secuencia a las fases del motor.

### ***Frenado***

De manera similar a como se realizan los arranques en los procesos industriales, los frenados en estos exigen que se hagan de manera muy suave. Imagínese que sucedería con la banda que lleva productos de vidrio si se hiciera un frenado rápido. En un variador de velocidad se pueden tener los siguientes tipos de frenados:

*Frenado inercial o natural:* consiste en dejar que la máquina desgaste su energía cinética en la carga, una buena aplicación para este tipo de frenado son las bombas de agua, que al quitársele la energía, al motor la bomba sigue trabajando por un pequeño tiempo. Este tipo de frenado se puede usar en procesos donde no se requiere demasiada precisión. Para hacer este tipo de frenado el variador de velocidad solo quita la energía y ningún proceso se hace luego.

*Frenado por rampa:* este frenado consiste en ir bajando la frecuencia desde un valor de trabajo hasta un frecuencia baja donde al motor se le aplica otro tipo de frenado como regenerativo, o el frenado con corriente continua.

*Frenado con corriente continua:* este método, consiste en inyectarle una corriente continua al estator del motor de inducción y este crea un flujo magnético constante que hace parar al motor. Para poder frenar el motor, este debe primero llegar a un velocidad que no sea peligrosa para el proceso (unas cuantas rpm) y después, a tensión reducida se le aplica una corriente continua al estator; se le deja un pequeño tiempo hasta que el motor está completamente quieto, luego se le debe quietar la corriente continua para que los devanados del estator no se sobrecalienten y se dañen. Para que al motor se le pueda aplicar el frenado con corriente continua, se debe aplicar antes el frenado inercial o el frenado por rampa.

*Frenado regenerativo:* este frenado precisa que la energía cinética que se obtiene en el frenado se gaste en una resistencia externa, y para tal fin algunos variadores de velocidad

necesitan una resistencia externa para su buen funcionamiento. Al igual que el frenado con c.c. se necesita que el motor haya perdido parte de su velocidad por un frenado inercial o por rampa. En este frenado el motor de inducción se convierte en generador bajando la frecuencia de alimentación del estator, y luego la energía se disipa en una resistencia externa.

### ***Protección integrada***

Los variadores modernos aseguran tanto la protección térmica de los motores como su propia protección. A partir de la medida de la corriente y de una información sobre la velocidad (si la ventilación del motor depende de su velocidad de rotación), un microprocesador calcula la elevación de temperatura de un motor y suministra una señal de alarma o de desconexión en caso de calentamiento excesivo.

Además, los variadores, y especialmente los convertidores de frecuencia, están dotados de protecciones contra:

- los cortocircuitos entre fases y entre fase y tierra,
- las sobretensiones y las caídas de tensión,
- los desequilibrios de fases,
- el funcionamiento en monofásico.

### ***Arranques con los variadores de velocidad***

Los métodos de arranque convencionales como arranque con resistencias, devanados parciales, a tensión reducida, etc, tienen el gran defecto de necesitar altas corrientes en el arranque, o en su defecto necesitan varios pasos de arranque para llevar un motor sin que se sobrecaliente demasiado hasta unas condiciones de trabajo. Este efecto no puede suceder en los variadores electrónicos de velocidad, ya que los transistores de potencia del puente en H se dañan (no permiten que las corrientes entre el emisor y el colector excedan demasiado la corriente de trabajo) o en su defecto estos transistores deberían ser demasiado grandes (que soporten la corriente de arranque del motor), lo que encarecería el variador demasiado y lo sobredimensionaría. En su defecto el motor debe arrancarse con una rampa de aceleración, en

la cual se van variando simultáneamente la tensión y la frecuencia de la señal alterna que alimenta el motor.

Para estipular el tiempo de aceleración del variador de velocidad, en aplicaciones industriales, las características del motor ya no influyen tanto como las características del proceso. Por ejemplo, si se necesita acelerar una carga, tan crítica, como lo es una cinta transportadora que lleva embases de vidrio llenos de algún líquido; no se le puede practicar los arranques convencionales, ya que estos tienen un torque de arranque que es alto e incontrolable y haría que los embases se cayeran de la banda. Cuando se arranque esta cinta con un variador electrónico el motor se debe acelerar lo más despacio posible para evitar que los productos se dañen, pero para que el proceso sea lo más rentable posible la banda se debe llevar a la máxima velocidad.

### ***Arranque con variadores Electrónicos***

Para arrancar el variador de velocidad solo hay que cerrar el contacto del switch SW que una el terminal LI1 y la fuente de 24V. Inmediatamente el motor deberá ir hasta la velocidad programada en el potenciómetro. El arranque del motor no se hace a una frecuencia nominal de 60Hz sino la frecuencia de salida del variador va aumentando de forma paulatina y lineal desde cero al punto de funcionamiento. Con esto se evita:

- a) Que las corrientes de arranque sean grandes, esta se limita a la corriente nominal.
- b) Las aceleraciones en el arranque sean altas.
- c) Hay pocos trastornos en la producción.

El tiempo de arranque se puede programar desde 0.25 segundos hasta varias horas. Lo que hace aplicable al variador a un amplio rango de trabajos.

### **2.2.3 Conexión del variador de velocidad.**

Un variador de velocidad se conecta entre la fuente de alimentación y el motor a controlar. Si el variador es de alta potencia se deben conectar las tres fases y si es de baja potencia solo es necesario tener conectada dos de sus fases como se muestra en la figura 3 izquierda. Para que el variador funcione correctamente se deben conectar varios elementos internos mínimos de funcionamiento como se muestra en la figura 3 derecha.

Los elementos mínimos son un potenciómetro para poder variar la velocidad y una llave selectora para invertir el sentido de giro. Con estos ajustes mínimos y un correcto ajuste de sus parámetros el variador está listo para funcionar.

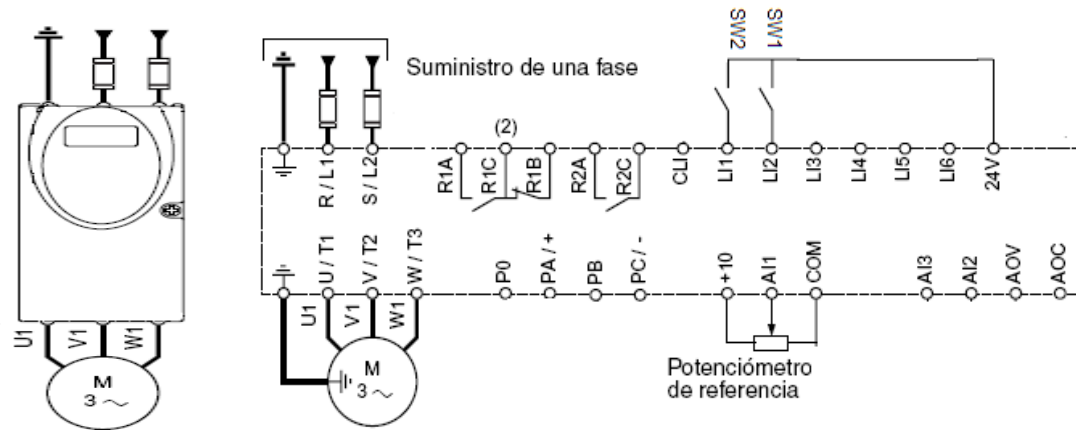


Figura 3. Conexión del variador de velocidad en forma manual.<sup>5</sup>

### 3. IMPLEMENTACION DE LA RED

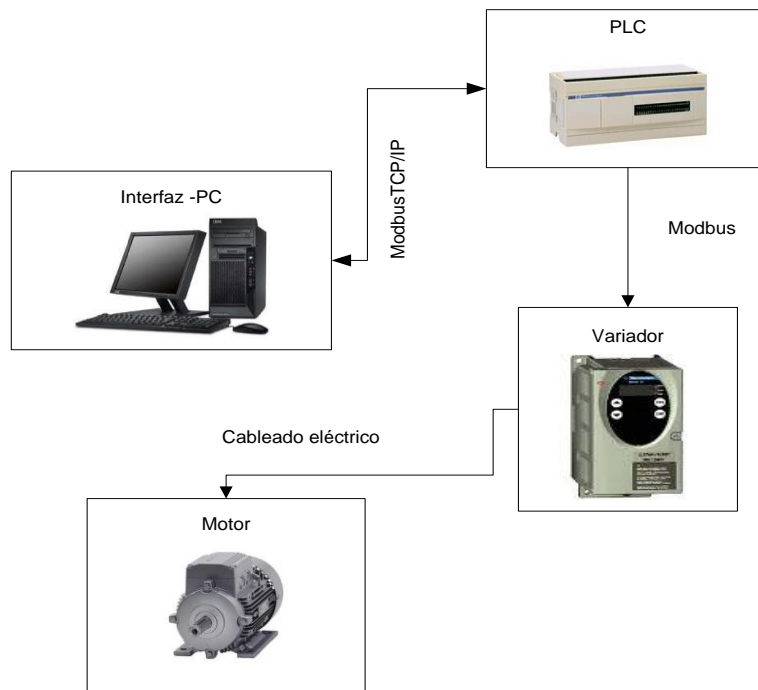


Figura 4. Red implementada

<sup>5</sup> <http://www.alamedaelectric.com/Modicon%20Documents/AC%20Drive%20ATV31%20Start-Up%20Guide.pdf>

Los elementos de la figura 4 representan la red montada, así como el tipo de conexión utilizada entre cada uno de ellos. En la conexión Modbus TCP/IP se utiliza un cable RS 485, mientras que para la conexión entre el PLC y el variador se utiliza un cable Modbus. El cable Modbus se construye con un conector RS 485 y un conector minidim como se muestra en la siguiente figura:

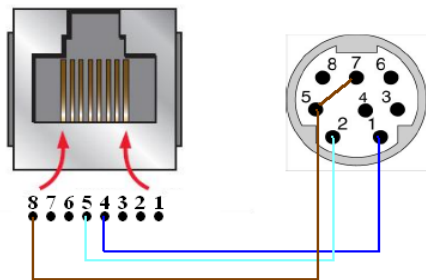


Figura 5. Cable Modbus para la conexión PLC-Variador<sup>6</sup>

### 3.1 CONFIGURACION DE LA CONEXIÓN MODBUS ENTRE EL PLC Y EL VARIADOR DE VELOCIDAD

Para crear una red ModBus con el variadores de velocidad configuramos en Twido lo siguiente: Se selecciona un PLC de referencia TWDLCAE40DRF y un variador de velocidad de referencia ATV31 y se configura el PLC como maestro y el variador como esclavo, quedando de esta forma:

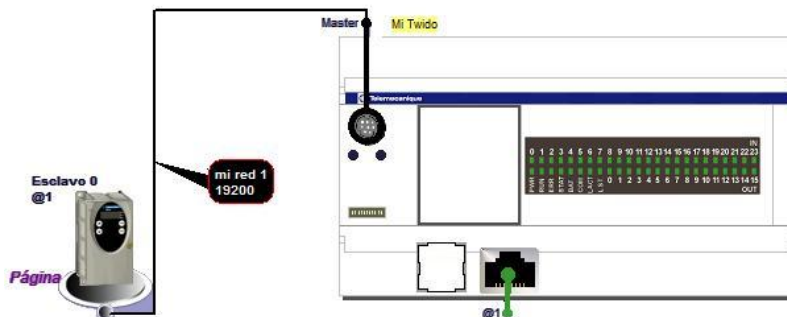


Figura 6. Esquema de configuración de red para el variador y el PLC

<sup>6</sup> Tomado de la guía de laboratorio Redes Modbus, UPB Bucaramanga

Posteriormente se selecciona la velocidad y los datos de la transmisión. Haciendo doble click en el cable de conexión de la red.

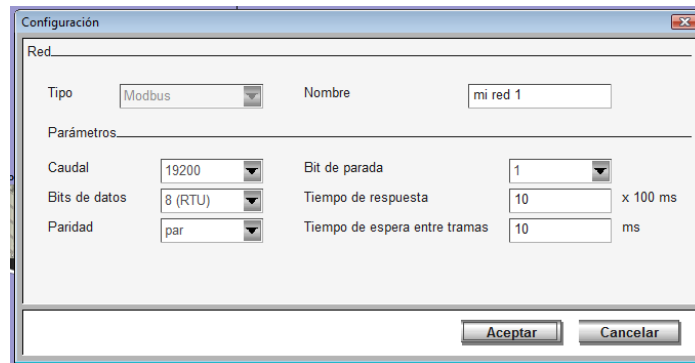


Figura 7. Configuración de parámetros para la comunicación

### 3.2 CONFIGURACION DEL VARIADOR PARA LA COMUNICACIÓN MODBUS

Después de realizar la conexión física entre el PLC y el variador, se configura el variador para realizar la comunicación ModBus, los parámetros a modificar y la forma en la cual se accede se muestra a continuación (practica redes industriales –red Modbus):

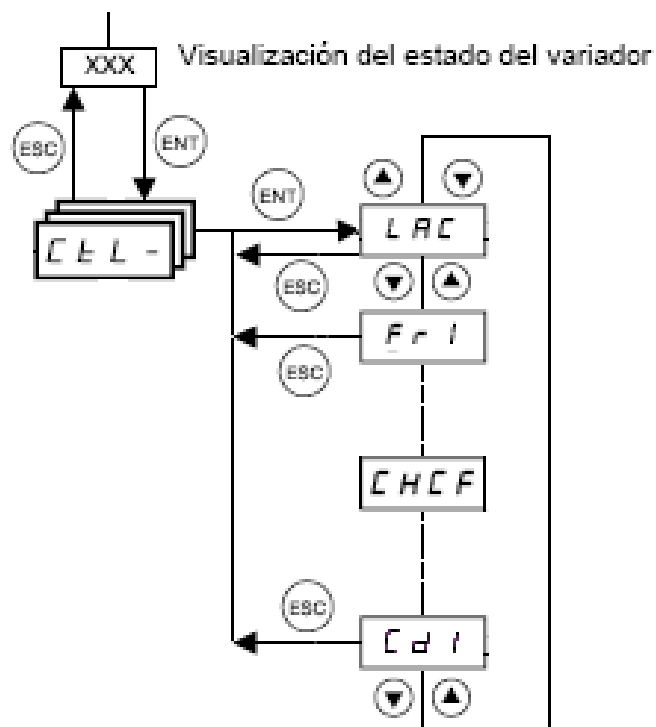


Figura 8. Diagrama para la configuración del variador.<sup>7</sup>

<sup>7</sup> <http://www.alamedaelectric.com/Modicon%20Documents/AC%20Drive%20ATV31%20Prog%20Manual.pdf>



Código	Descripción	Ajuste para modo Modbus
<b>LAC</b>	<b>L3: Acceso a las funciones avanzadas y gestión de los modos de control mixtos.</b>	<b>L3</b>
<b>Fr1</b>	Ndb: Consigna por Modbus	<b>Ndb</b>
<b>CHCF</b>	<b>SEP: Separados</b>	<b>SEP</b>
<b>Cd1</b>	Ndb: Control a través de Modbus	<b>Ndb</b>

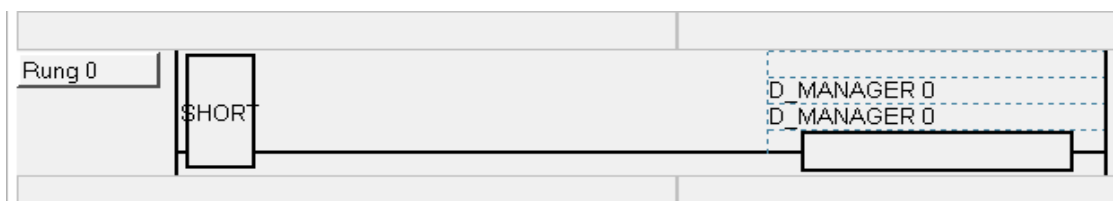
*Tabla 4. Datos configurados en el variador*

Esta configuración debe ser realizada antes de realizar la conexión con el PLC, ya que de lo contrario se pueden presentar daños tanto en el PLC como en el variador.

### 3.3 PROGRAMACION DEL CODIGO PARA LA RECEPCIÓN Y ENVIO DE DATOS

Después de configurar el variador, se programa en twidoSuite el código para la recepción y envío de datos del variador, así como para ejecutar las acciones de mando enviadas desde nuestra interfaz, luego este se transfiere al PLC. Este código se muestra a continuación:

Funcion D\_MANAGER\_0: Esta función debe activarse al principio de cada ciclo de funcionamiento del Twido y antes de activar cualquier otra función de unidad para garantizar el funcionamiento correcto de la ATV.



*Figura 9. Función de inicialización del variador*

Se configura el variador en modo consigna ( $D\_SETPOINT\_MODE=0$ ) y posteriormente se introduce el valor de la velocidad ( $D\_SETPOINT\_0$ ).

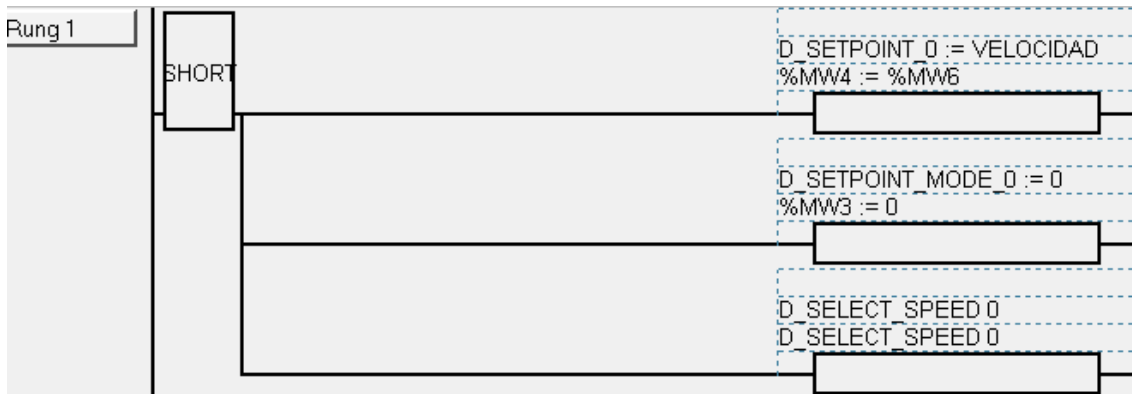


Figura 10. Configuración de parámetros iniciales

Condiciones utilizadas para encender, apagar y cambiar de giro al motor.

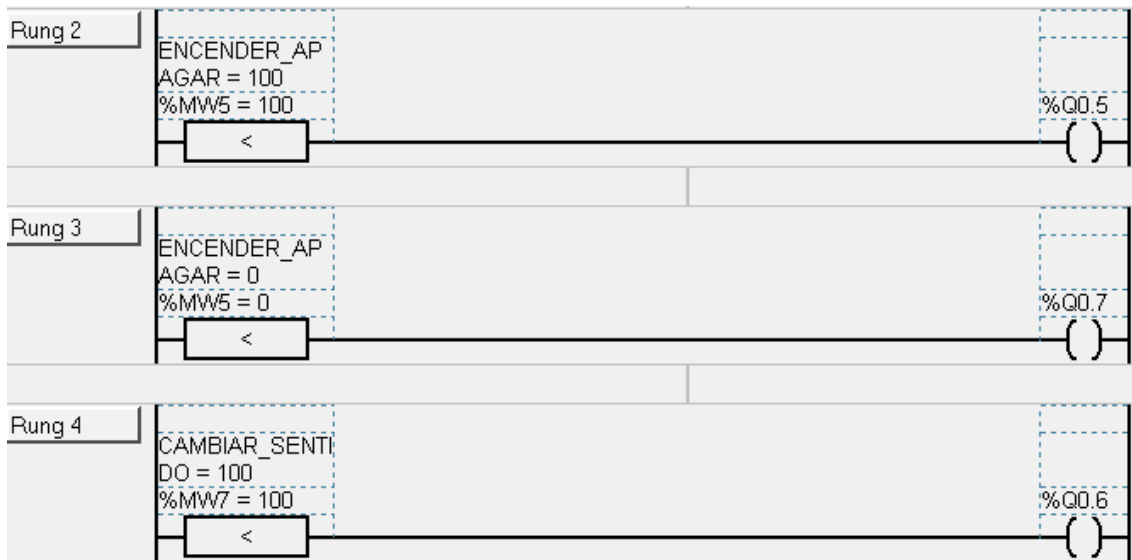


Figura 11. Definición de valores para el encendido, apagado y cambio en el sentido de giro

Esta línea guía a la unidad de velocidad ATV en sentido horario. La velocidad debe establecerse con D\_SELECT\_SPEED antes de activar esta macro.



Figura 12. Función para establecer la velocidad en sentido horario

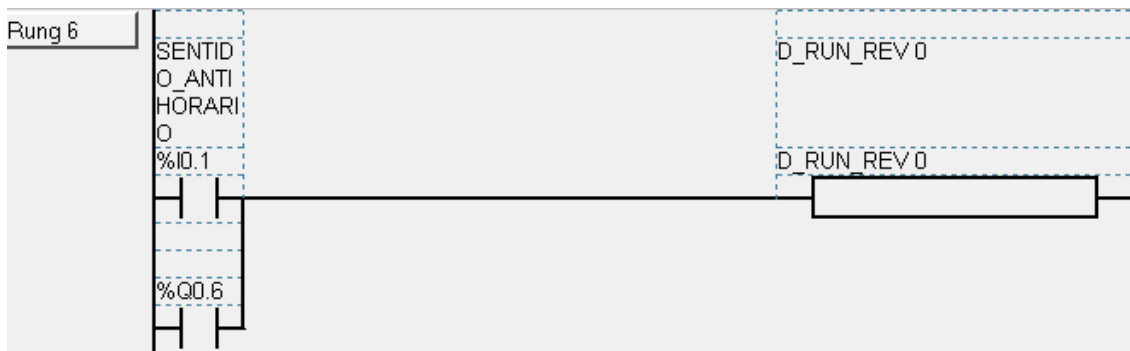


Figura 13. Función para establecer la rotación en sentido anti horario

La velocidad debe establecerse con D\_SELECT\_SPEED antes de activar estas dos macros.

Se guía a la unidad de velocidad ATV en la aplicación de la secuencia detener para el proceso.



Fig 14. Función para detener el proceso

Se programa la velocidad máxima del motor que será introducida en RPM como se muestra a continuación:



Figura 15. Función para establecer la velocidad

Con estas líneas se permite al usuario borrar los errores almacenados en el búfer (Es la ubicación en la memoria de un instrumento).

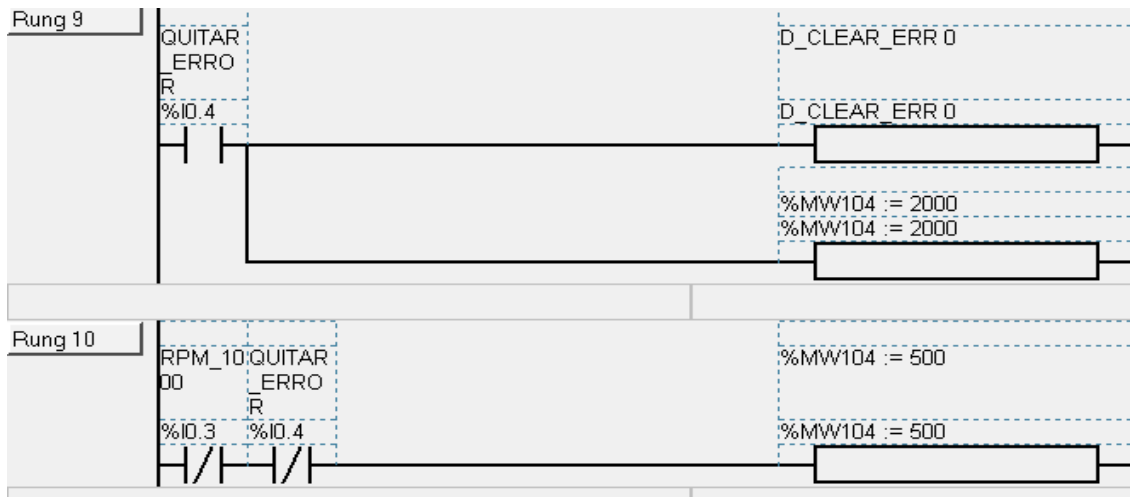


Figura 16. Función para eliminar los errores del búfer

En la siguiente imagen se muestra el programa de la aplicación en Lista:

```

1 IL
0   LD    1
1   [ D_MANAGER 0 ]
2   LD    1
3   [ D_SETPOINT_0 := VELOCIDAD ]
4   [ D_SETPOINT_MODE_0 := 0 ]
5   [ D_SELECT_SPEED 0 ]
6   LD    [ ENCENDER_APAGAR = 100 ]
7   ST    %Q0.5
8   LD    [ ENCENDER_APAGAR = 0 ]
9   ST    %Q0.7
10  LD    [ CAMBIAR_SENTIDO = 100 ]
11  ST    %Q0.6
12  LD    %IO.0
13  OR    %Q0.5
14  [ D_RUN_FWD 0 ]
15  LD    SENTIDO_ANTIHORARIO
16  OR    %Q0.6
17  [ D_RUN_REV 0 ]
18  LD    PARAR_MOTOR
19  OR    %Q0.7
20  [ D_STOP 0 ]
21  LD    RPM_1000
22  [ %MW104 := 1000 ]
23  LD    QUITAR_ERROR
24  [ D_CLEAR_ERR 0 ]
25  [ %MW104 := 2000 ]
26  LDN   RPM_1000
27  ANDN  QUITAR_ERROR
28  [ %MW104 := 500 ]
29  END

```

Figura 17. Lista del programa

Used	Address	Symbol	Comment
<input checked="" type="checkbox"/>	%MW7	CAMBIAR_SENTIDO	
<input checked="" type="checkbox"/>	%MW1	D_CANSTATE_0	Altivar CANOpen state
<input checked="" type="checkbox"/>	%MW2	D_ERROR_0	Altivar error code
<input checked="" type="checkbox"/>	%MW28	D_MODBUS_INIT_PHA	Modbus initialisation phase running Bit0
<input checked="" type="checkbox"/>	%MW17	D_SELECT_SPEED_VA	Control effort on the Altivar
<input checked="" type="checkbox"/>	%MW4	D_SETPOINT_0	Altivar set-point
<input checked="" type="checkbox"/>	%MW3	D_SETPOINT_MODE_0	Altivar set-point mode
<input checked="" type="checkbox"/>	%MW0	D_STATE_0	Altivar state
<input checked="" type="checkbox"/>	%MW5	ENCENDER_APAGAR	
<input checked="" type="checkbox"/>	%I0.2	PAPAR_MOTOR	
<input checked="" type="checkbox"/>	%I0.4	QUITAR_ERROR	
<input checked="" type="checkbox"/>	%I0.3	RPM_1000	
<input checked="" type="checkbox"/>	%I0.1	SENTIDO_ANTIHORAR	
<input checked="" type="checkbox"/>	%MW6	VELOCIDAD	

Figura 18. Asignación de símbolos a entradas

Mediante este programa al pulsar %I0.0 o escribir 100 en %MW5 el motor girara en el sentido de las manecillas del reloj, al pulsar %I0.1 o escribir 100 en %MW7 cambiara el sentido de giro y si se pulsa %I0.2 0 cero(0) en %MW5 se detiene el motor. La escritura de estos valores en los registros se realizara con la interfaz.

## 4. INTERFAZ GRAFICA EN VISUAL BASIC

### 4.1 CONCEPTOS PREVIOS

- **Visual Basic:** Visual Basic es un lenguaje de programación orientado a eventos. la versión utilizada en este proyecto es la 6.
- **ProtocoloTCP/IP:** Internet usa el protocolo TCP/IP que se encarga de recibir paquetes de información y redirigirlos al usuario final que los solicitó. Este protocolo es muy utilizado ya que puede verificar que el paquete de información ha llegado con éxito al destinatario final, concretando así la transacción
- **Winsock:** Winsock es un control utilizado en VB para hacer conexiones. Este Envía y recibe mensajes en un puerto y una dirección IP específicos. Estas son algunas propiedades o códigos utilizados para realizar la comunicación:

*Name:* Éste es el nombre que das a ese objeto particular en VB.

*Localport:* Aquí especificas el puerto en el cual el objeto del Winsock debe escuchar.

*protocol*: protocolo que se va a utilizar, TCP/IP o UDP. para este caso TCP/IP

*RemoteHost*: dirección IP al cual se van a enviar los datos, en este proyecto al PLC.

*RemotePort*: Puerto donde se recibirán los datos.

*Close*: cierra toda conexión en el winsock especificado.

*Connect*: Conecta a un determinado host y puerto.

*State*: Muestra el estado de conexión.

*SendData*: Envía Datos con winsock a un server remoto.

*Get Data*: Obtiene los datos llegados a winsock

## 4.2 DESCRIPCIÓN DEL FUNCIONAMIENTO, ELEMENTOS Y CODIGO DE LA INTERFAZ CLIENTE MODBUS TCP/IP

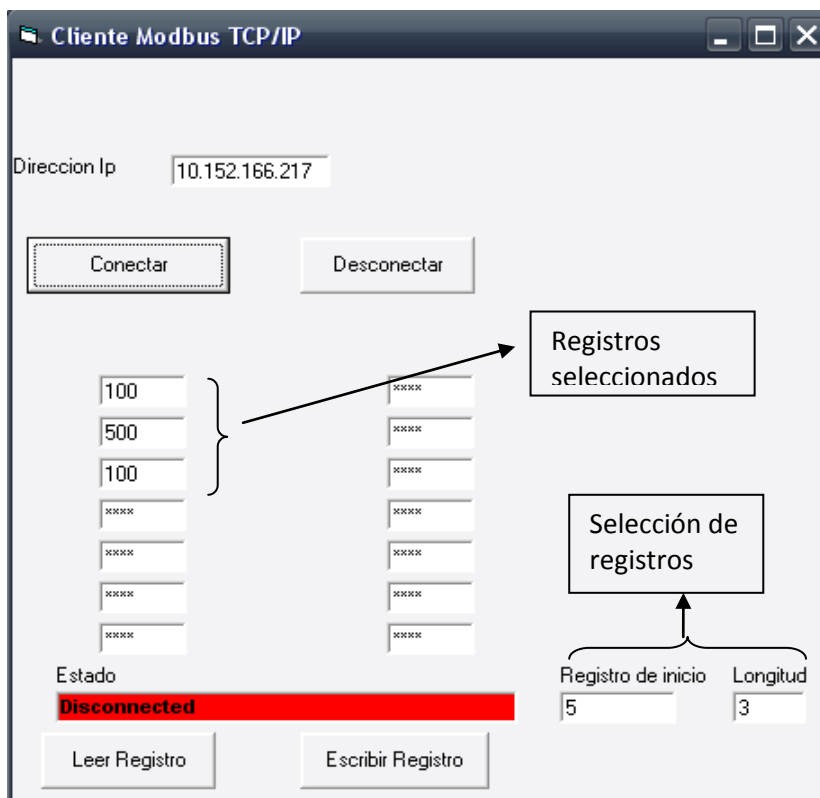


Figura 19. Interfaz

#### 4.2.1 Funcionamiento de la Interfaz

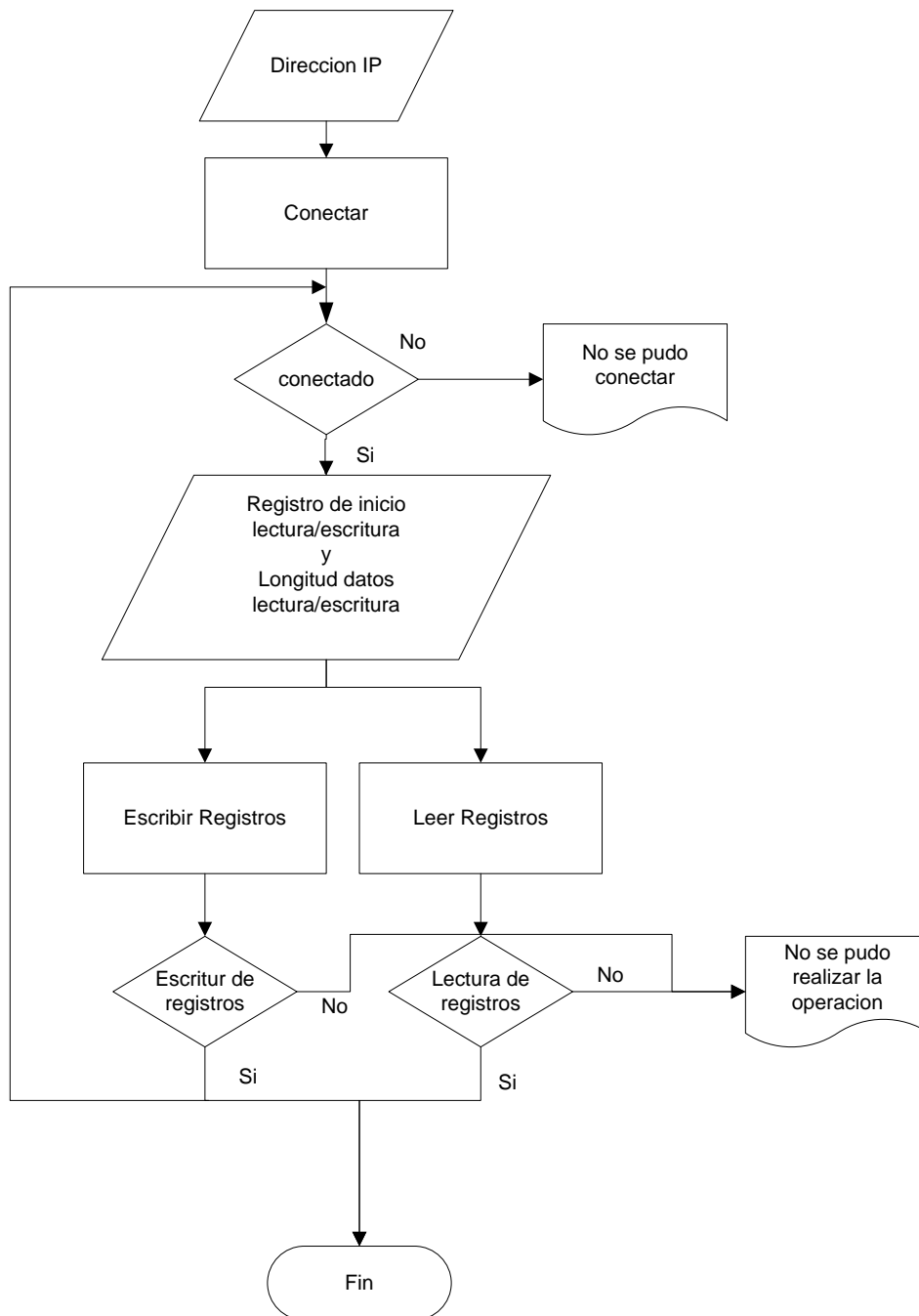


Figura 20. Diagrama de flujo de la interfaz

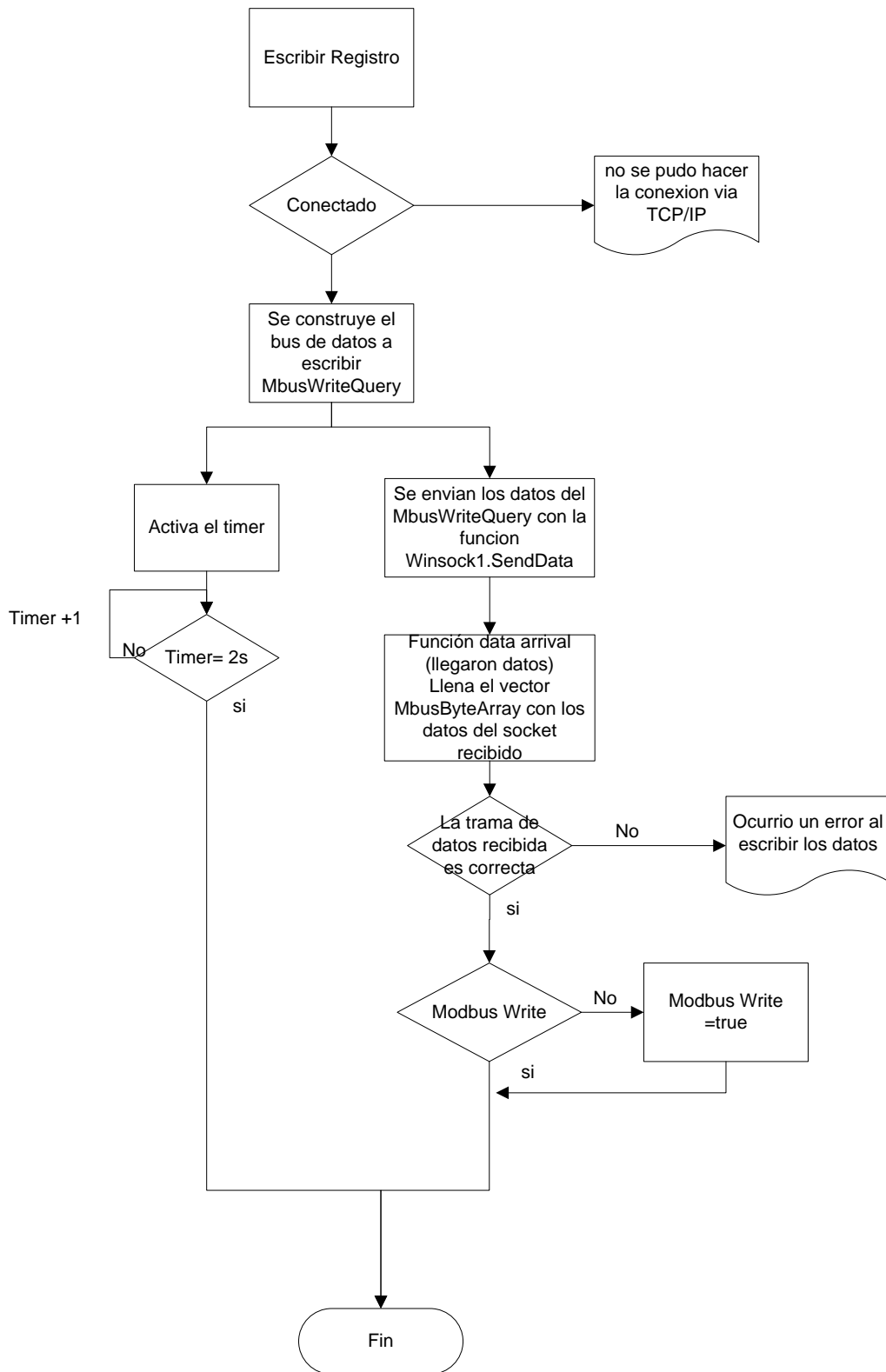


Figura 21. Diagrama de flujo Escribir Registro



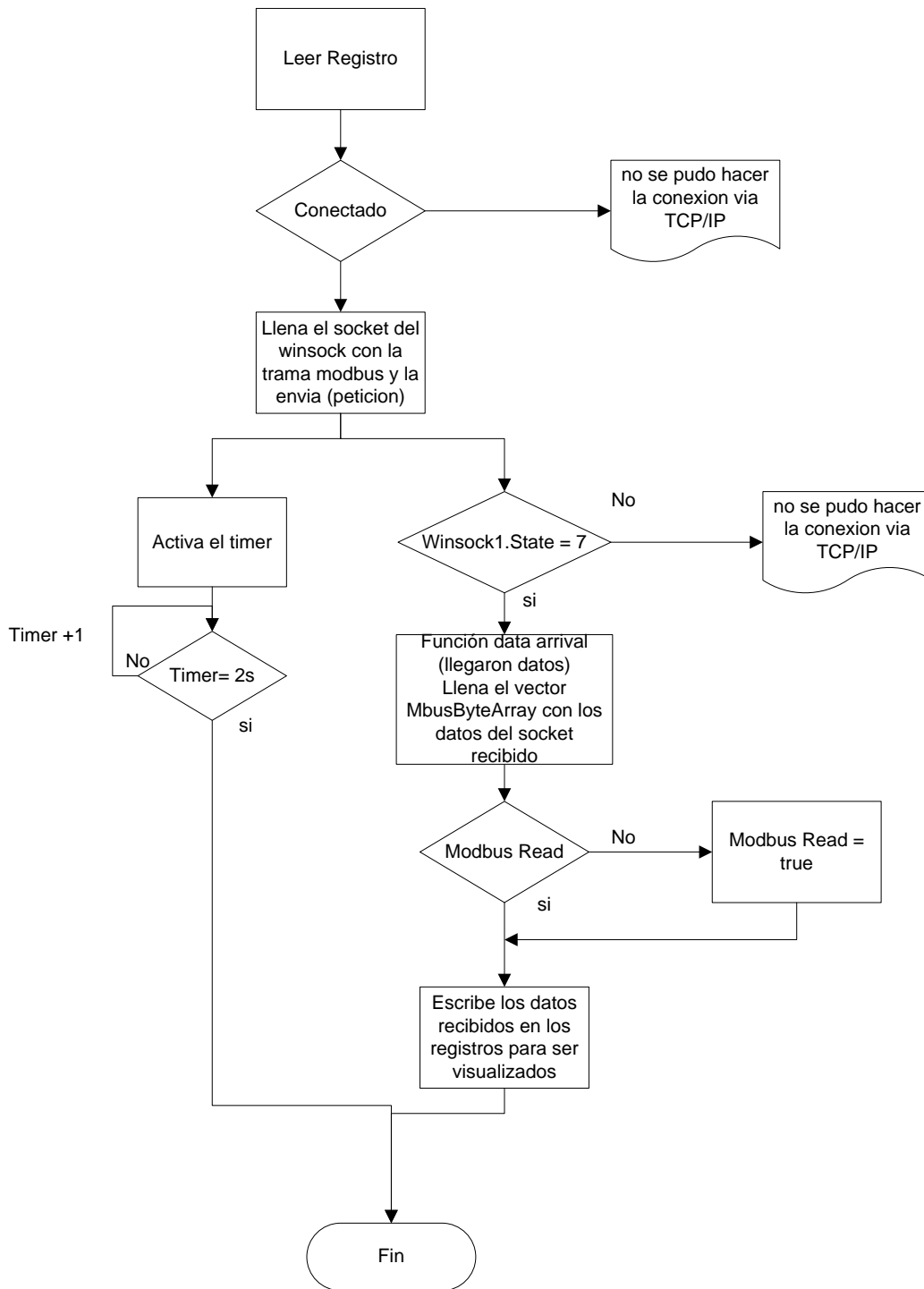


Figura 22. Diagrama de flujo Leer Registro

Para realizar la conexión Modbus TCP/IP se ingresa la dirección IP del PLC al cual se va a comunicar en el campo “Dirección IP”, Luego, damos click en el botón “conectar”. Si la conexión es exitosa el estado será “conectado”.

Para leer o escribir los datos del PLC escribimos la cantidad o longitud de datos que se quiere leer o escribir en el campo “Longitud” y la posición desde el cual se quiere empezar a ejecutar

la opción escogida en el campo “Registro de Inicio” y presionamos el botón “leer Registro” o “escribir Registro” respectivamente.


Según la asignación de símbolos a entradas (figura 15) en la figura 16, las variables velocidad, sentido, apagado y encendido (%w6,%w5,%w7) deben ser definidas de la siguiente manera:

Para encender el motor se escribe en la casilla 1 100 y para apagar “0”.

En la casilla 2 está la velocidad deseada en el motor (0 a 2000).

Para cambio de giro la casilla 3= debe ser igual 100.

#### 4.2.2 Descripción de elementos de la interfaz

 Winsock ( ): Control Utilizado Para la Comunicación Vía TCP/IP. (Control oculto)

Constantes de estado definidas para winsock		
Valor	Nombre	Descripción
0	sckClosed	connection closed
1	sckOpen	open
2	sckListening	listening for incoming connections
3	sckConnectionPending	connection pending
4	sckResolvingHost	resolving remote host name
5	sckHostResolved	remote host name successfully resolved
6	sckConnecting	connecting to remote host
7	sckConnected	connected to remote host
8	sckClosing	connection is closing
9	sckError	error ocured

Tabla 5. Constantes de estado definidas para winsock

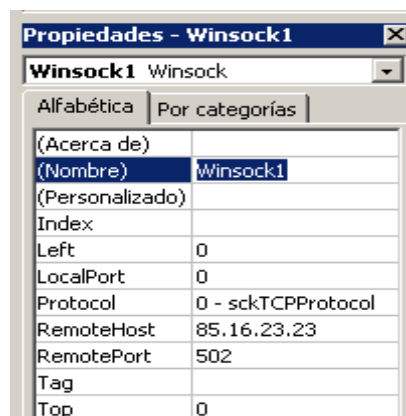


Figura 23. Propiedades del elemento Winsock1

Timer (🕒): Esta función de timer, permite evaluar los tiempos de respuesta en las peticiones realizadas con el winsock para determinar si esta fue exitosa o no (Control oculto).



Figura 24. Propiedades del Timer

*DireccionIP*: Se escribe la dirección IP a la cual el PC se va a comunicar. DirecciónIP del PLC.

*Conectar*: Al hacer click sobre este botón se realiza la conexión entre el PC y el PLC.

*Desconectar*: Al presionar este botón la comunicación se interrumpe, es decir, se desconecta.

*Texto*: en el texto se puede leer o escribir los datos que deseamos que sean leídos o escritos en el PLC.

*Estado*: Muestra el estado de la conexión, ya sea que esté conectado, no conectado o se presente algún problema en la conexión.

*Registro de Inicio*: Registro inicial desde donde quiero que se muestren los registros.

*Longitud*: cantidad de registros que deseo ver o escribir.

*Leer registro*: lee los registros del PLC

*Escribir Registro*: Escribe registros en el PLC

### 4.2.3 Descripción de las funciones del código

#### **Función Conectar**

```

PrivateSub conectar_Click()
Dim StartTime
} Esta función se ejecuta al dar click sobre el
} botón de conectar

If (Winsock1.State <> sckClosed) Then
Winsock1.Close()
EndIf
} cierra la conexión en caso no estar
} cerrada
Winsock1.RemoteHost = Text1.Text
} Asigna el valor escrito en "dirección ip"
} al host
Winsock1.Connect()
} Conecta al host previamente establecido

StartTime = Timer
DoWhile ((Timer < StartTime + 2) And (Winsock1.State <> 7))
DoEvents()
Loop
} Se inicializa el timer
} y realiza los eventos
} necesarios para
} establecer la
} comunicación

If (Winsock1.State = 7) Then
Text5.Text = "Conectado"
Text5.BackColor = &HFF00&
Else
Text5.Text = "no pudo conectar a " + Text1.Text
Text5.BackColor = &HFF
EndIf
EndSub

```

#### **Función Leer Registro**

```

PrivateSub leer_registro_Click()
Dim StartLowAsByte
Dim StartHighAsByte
Dim LengthLowAsByte
Dim LengthHighAsByte
} Declaración de variables

If (Winsock1.State = 7) Then
MbusQuery(0) = 0
MbusQuery(1) = 0
} Identificadores de transacción.
MbusQuery(2) = 0
MbusQuery(3) = 0
} Identificadores de protocolo
MbusQuery(4) = 0
MbusQuery(5) = 6
MbusQuery(6) = 1
MbusQuery(7) = 3
} Mensajes menores a 256
} Numero de bytes siguientes
} Identificador de unidad(esclavo 1)
} Código de función Modbus
MbusQuery(8) = StartHigh
MbusQuery(9) = StartLow
MbusQuery(10) = LengthHigh
MbusQuery(11) = LengthLow
} Cantidad de datos a leer o escribir
} Representados de la forma "big- endian"
MbusRead = True
MbusWrite = False
Winsock1.SendData(MbusQuery)
ModbusWait = True
ModbusTimeOut = 0
Timer1.Enabled = True
} Actualización de punteros
} y envío de la trama de datos

```

```

Else
MsgBox("no se pudo hacer la conexionvia TCP/IP")
EndIf
EndSub

```

### ***Función Escribir Registros***

```

PrivateSubescribir_registro_Click()
DimMbusWriteCommandAsString
DimStartLowAsByte
DimStartHighAsByte
DimByteLowAsByte
DimByteHighAsByte
Dim i AsInteger
If (Winsock1.State = 7) Then
StartLow = Val(Text2.Text) Mod 256
StartHigh = Val(Text2.Text) \ 256
LengthLow = Val(Text3.Text) Mod 256
LengthHigh = Val(Text3.Text) \ 256

```

```

MbusWriteQuery = Chr(0) + Chr(0) + Chr(0) + Chr(0) + Chr(0) + Chr(7 + 2 *
Val(Text3.Text)) + Chr(1) + Chr(16) + Chr(StartHigh) + Chr(StartLow) + Chr(0) +
Chr(Val(Text3.Text)) + Chr(2 * Val(Text3.Text))

```

Trama Modbus/TCP para enviar datos

```

For i = 0 ToVal(Text3.Text) - 1
ByteLow = Val(Text4(i).Text) Mod 256
ByteHigh = Val(Text4(i).Text) \ 256
MbusWriteQuery = MbusWriteQuery + Chr(Val(Text4(i).Text) Mod 256) +
Chr(Val(Text4(i).Text) \ 256)
Next i

```

} Tramamodbus +  
datos

```

MbusRead = False
MbusWrite = True
Winsock1.SendData(MbusWriteQuery) } Envió de datos
ModbusWait = True
ModbusTimeOut = 0
Timer1.Enabled = True
Else
MsgBox("no se pudo hacer la conexión via TCP/IP")
EndIf
EndSub

```

### ***Funcion Data Arrival***

```
Private Sub Winsock1_DataArrival(ByVal datalength As Long)
Dim b As Byte

For i = 1 To datalength
    Winsock1.GetData b
    MbusByteArray(i) = b
Next

If MbusRead Then
    For i = 10 To MbusByteArray(9) + 9 Step 2
        Text4(j).Text = Str((MbusByteArray(i) * 256) + MbusByteArray(i + 1))

        Next i
    Text5.Text = "Registers read"
    Text5.BackColor = &HFF00&

    ModbusWait = False
    ModbusTimeOut = 0
    Timer1.Enabled = False

End If
If MbusWrite Then
If (MbusByteArray(8) = 16) And (MbusByteArray(12) = Val(text3,text)) Then
    Text5.Text = "Registers written"
    Text5.BackColor = &HFF00&
    ModbusWait = False
    ModbusTimeOut = 0
    Timer1.Enabled = False
Else
    Text5.Text = "Ocurrio un error al escribir los datos"
    Text5.BackColor = &HFF
End If

End If

End Sub
```

### ***Función Timer***

```
Private Sub Timer1_Timer()
ModbusTimeOut = ModbusTimeOut + 1
If ModbusTimeOut > 2 Then
ModbusWait = False
ModbusTimeOut = 0
Text5.Text = "Modbus Time Out"
Text5.BackColor = &HFF
Timer1.Enabled = False
End If
End Sub
```

### ***Función Desconectar***

```
PrivateSub desconectar_Click()  
  
If (Winsock1.State <> sckClosed) Then  
Winsock1.Close()  
EndIf  
DoWhile (Winsock1.State <> sckClosed)  
DoEvents()  
Loop  
  
Text5.Text = "Desconectado"  
Text5.BackColor = &HFF  
EndSub
```

Esta función se encarga de cerrar la conexión

## CONCLUSIONES

- Con el uso del protocolo Modbus/TCP en el software supervisor de Visual Basic, se evidenció la facilidad y flexibilidad de este protocolo y por ende la razón de su alta difusión en entornos industriales.
- El uso de protocolo Modbus/TCP nos brinda la posibilidad de implementar fácilmente software supervisor sobre plataformas de desarrollo comunes, al necesitar solamente que estas posean librerías para el protocolo TCP/IP, además, la información que se maneja puede ser fácilmente llevada al internet; Esto puede facilitar desde la reparación y supervisión de equipos hasta el control de procesos, evitando el desplazamiento hasta el lugar de la instalación.
- Mediante el software desarrollado se puede escribir y leer registros del plc, los cuales nos brindan la posibilidad de controlar diferentes acciones, en este caso la velocidad, arranque y parada de un motor asíncrono.
- La librería Winsock de Visual Basic posee las herramientas necesarias para realizar la conexión y envío de datos vía TCP/IP, que fue de gran utilidad para el desarrollo de la interfaz, presentándose como una buena opción a la hora de desarrollar un software supervisor para una red ModBus.



## BIBLIOGRAFIA

- <http://read.pudn.com/downloads126/sourcecode/app/534835/MbusTCPTest/Form1.htm>
- [http://www.dte.upct.es/personal/manuel.jimenez/docencia/GD6\\_Comunic\\_Ind/pdfs/Tema%207.pdf](http://www.dte.upct.es/personal/manuel.jimenez/docencia/GD6_Comunic_Ind/pdfs/Tema%207.pdf)
- <http://www.rtaautomation.com/modbustcp/>
- <http://www.monografias.com/trabajos75/red-modbus-tcp-ordenador/red-modbus-tcp-ordenador2.shtml>
- [http://www.infopl.net/Descargas/Descargas\\_Schneider/Des\\_Schneider\\_Files/infoPLC\\_net\\_ALTIVAR\\_31\\_COMUNICACION\\_MODBUS.html](http://www.infopl.net/Descargas/Descargas_Schneider/Des_Schneider_Files/infoPLC_net_ALTIVAR_31_COMUNICACION_MODBUS.html)
- <http://www.alamedaelectric.com/Modicon%20Documents/AC%20Drive%20ATV31%20Prog%20Manual.pdf>
- Twidoprogramable controllers Software Reference Guide
- Manual de usuario PLC TSX ETZ de Telemecanique