5 COMUNICACIONES: PROTOCOLO MODBUS TCP/IP

5.1 Protocolo de comunicación Modbus TCP/IP:

En las comunicaciones se ha usado el protocolo Modbus TCP/IP. Es un protocolo de mensajería sobre capas de aplicación, posicionado en el nivel 7 del modelo OSI, que proporciona la comunicación de cliente/servidor entre dispositivos conectados sobre tipos diferentes de buses o redes.

El protocolo Modbus TCP/IP consiste en una adaptación de Modbus a entornos de comunicación basados en redes TCP/IP Modbus. De este modo podemos decir que Modbus TCP/IP es también un protocolo que sigue el modelo Maestro/Esclavo en el que el maestro (cliente) requiere una serie de servicios de uno o varios esclavos (servidores); estos servicios consisten básicamente en la lectura y escritura de valores almacenados en registros. Los registros serán de 16 bits, con lo que si se tienen datos de tamaño superior a un registro, el maestro debe tener en cuenta esta situación para pedir el acceso a más de un registro, ya sea en lectura como en escritura.

La comunicación se establece por el puerto 502 sobre TCP/IP y especificando la dirección del esclavo y la IP.

En este sistema se utiliza el interfaz Ethernet empleando el protocolo estándar para entornos industriales de control Modbus para comunicaciones maestro-esclavo.

5.2 Variables que intervienen en la comunicación

El sistema Maestro pretende llevar la supervisión de trece variables, las cuales se leen directamente de determinados registros del esclavo. De estas variables diez son de sólo lectura y tres de lectura/escritura. En la tabla siguiente se muestra un listado de las variables especificando el número del registro/s que ocupa cada una, el formato, el registro o registros donde se encuentran y si son de lectura o lectura/escritura y:

Registro	Descripción del registro	Descripción de la variable
0 (R)	(float 32 bits) Parte alta de **Ib**	Corriente de salida de las baterías
1 (R)	Parte baja de **Ib**	
2 (R)	(float 32 bits) Parte alta de **Vo**	Tensión de DC-LINK
3 (R)	Parte baja de **Vo**	
4 (R)	(float 32 bits) Parte alta de **Vac**	Tensión de salida del Inversor (antes del transformador)
5 (R)	Parte baja de **Vac**	
6 (R)	(float 32 bits) Parte alta de **Vacs**	Tensión de salida del equipo (después del transformador)
7 (R)	Parte baja de **Vacs**	
8 (R)	(float 32 bits) Parte alta de **Vb**	Tensión de las baterías
9 (R)	Parte baja de **Vb**	
10 (R)	(int 16 bits) AlarmaIimax	Alarma de corriente de la Pila máxima
11(R)	(int 16 bits) AlarmaVi_mínima	Alarma de tensión de la Pila mínima
12(R)	(int 16 bits) AlarmaBUS	Alarma de tensión del BUS máxima
13 (R)	(int 16 bits) AlarmaBaterias	Alarma de baterías bajo mínimos
14 (R)	(int 16 bits) AlarmaTemperaturaMáx	Alarma de temperatura máxima del sistema y/o componentes
16 (R)	(int 16 bits) **fuente/Estado**	Estado de funcionamiento del sistema
17 (R)	(float 32 bits) Parte alta de **Ii**	Corriente de salida de la Pila
18 (R)	Parte baja de **Ii**	
19 (R)	(float 32 bits) Parte alta de **Pref**	Potencia de referencia de la carga
20 (R)	Parte baja de **Pref**	

Nota: Los registros de 16 bits almacenan variables booleanas (activado/desactivado) de tal forma que:

- 0x00: Activada
- 0x01: Desactivada

5.3 Estructura de los mensajes Modbus TCP/IP

La estructura de los mensajes Modbus TCP/IP es muy parecida a los de Modbus, se han añadido nuevos campos y otros han desaparecido (Chequeo de errores).

Se ha suprimido el campo de chequeo de errores porque la capa Ethernet ya realiza por sí misma un chequeo de la trama con un CRC de 32 bits, con lo que no hace falta que la capa Modbus haga otro chequeo.

A continuación se describen los campos que forman la trama de este protocolo:

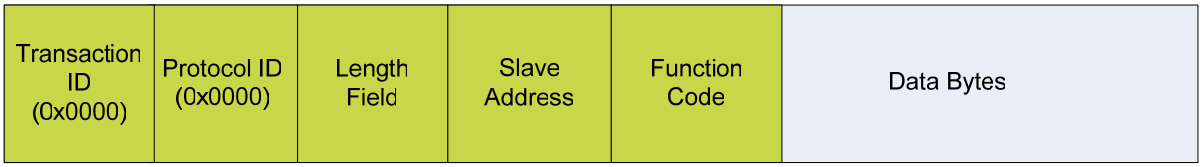

Figura 14: Estructura de los mensajes Modbus

Un último detalle es comentar que en Modbus TCP/IP se ha cambiado el nombre del campo “Slave Address” por “Unit Identifier”, pero salvo este detalle son iguales.

Transacción ID: este campo consta de 2 bytes, en principio su valor será 0x0000, aunque si al maestro le interesa, puede darle valor para llevar un mayor control sobre las operaciones. El esclavo simplemente replicará este valor.

Protocolo ID: este campo también consta de 2 bytes, su valor es 0x0000.

Length Field: consta de 2 bytes. Indica la longitud en bytes del resto de la trama, es decir la longitud de los campos Slave address, Function Code y Data Bytes.

Slave address: consta de 1 byte. Permite al esclavo saber a quién va dirigida una petición o bien al maestro saber qué esclavo está respondiendo a una petición anterior.

Function Code: consta de 1byte. Éste indica al esclavo la operación que se le está solicitando. Al maestro le indica a qué tipo de petición se le está respondiendo.

Los códigos de función que se han implementado para el maestro son lectura y escritura de varios registros.

-0x03: Lectura de múltiples registros
-0x10: Escritura de múltiples registros.

Cuando se leen las variables del esclavo se leen todas a la vez, ya que es más rápido, y cuando se tiene que escribir se escriben de una en una, aunque sigue haciendo falta usar la escritura en múltiples registros ya que dos de las tres variables de escritura ocupan dos registros cada una.

Data Bytes: es el campo de datos, tanto el esclavo como el maestro interpretarán los datos según el código de función

A continuación se detalla la construcción de las tramas de escritura y lectura utilizadas

5.3.1 Lectura:

Como ya se ha dicho, para mayo rapidez en la comunicación la lectura de los registros del esclavo se hace de una vez, es decir con una petición de lectura se leen todos sus registros. A continuación se describen las tramas usadas en lectura de registros, tanto en petición como en respuesta:

5.3.1.1 Trama de Petición de lectura:

1) “Function Code”: 0x03 (1byte)

2) Data bytes: el campo de datos está compuesto por 4bytes. A su vez se divide en dos subcampos de 2 bytes cada uno:

2-1) “Reference number”: Indica el registro de referencia a partir del cual se van a empezar a leer los registros. Para esta aplicación el registro de referencia es el cero por lo que el valor de este campo es 0x0000.

2-2) “Word count” (1-125): indica el número de registros a leer (los registros son de 16 bits). En esta aplicación se tienen un total de 20 registros que en hexadecimal es 0x0014

3)”Lenght Field”: son 6 bytes, uno de dirección del esclavo, otro del código de función y cuatro del campo de datos. Su valor es 0x0006.

La trama completa de petición es:

Transaction ID	Protocolo ID	Length Field	Slave address	Function code	Referente number	Word Count
0000	0000	0006	01	03	0000	0014
En hexadecimal						

5.3.1.2 Trama de respuesta a lectura:

1)"Function Code": 0x03 (1byte)

2) "Data bytes": está compuesto por dos subcampos.

2-1) Byte count (1byte): indica el número de bytes que tienen los datos. Son 40 bytes (20 registros de 2 byte cada uno) que en hexadecimal es 0x28.

2-2) Register values: son los datos leídos.

3) Length Field: Son 43 bytes en total (uno de dirección del esclavo, otro del código de función, otro de Byte count y por último 40 del campo de datos). En hexadecimal 0x002B

La trama completa de respuesta es:

Transaction ID	Protocolo ID	Length Field	Slave address	Function code	Byte Count	Register Values
0000	0000	002B	01	03	28	Datos (40bytes)
En hexadecimal						

5.3.1.3 Excepciones en trama de lectura:

En caso de que el esclavo en la trama de petición no admita el código de función o la dirección del dato sea incorrecta responderá con una trama de error. El código de función de esta trama es el resultado de sumarle ochenta en hexadecimal al código de función original. El campo de datos estará constituido por un byte con valor cero o uno dependiendo del tipo de error:

En caso de que el esclavo no admita el código de función se le conoce como error tipo 1 y su estructura es:

Transaction ID	Protocolo ID	Length Field	Slave address	Function code	Exception Code
0000	0000	0003	01	83	01
En hexadecimal					

El caso de que el error sea por no admitir la dirección del dato se le conoce como error tipo 2 y su estructura es:

Transaction ID	Protocolo ID	Length Field	Slave address	Function code	Exception Code
0000	0000	0003	01	83	02
En hexadecimal					

5.3.2 Escritura:

Al contrario que en lectura, la escritura se hace variable por variable. El hecho de que algunas de las variables de escritura ocupen más de un registro obliga a usar la escritura de múltiples registros de este protocolo. A continuación se describen las tramas usadas en escritura de registros, tanto en petición como en respuesta:

5.3.2.1 Trama de Petición de escritura:

1)” Function Code” (1byte): 0x10

2) “Data bytes”: el campo de datos está compuesto por cuatro subcampos que se describen a continuación:

2-1) “Referente number” (2bytes): Indica el registro de referencia a partir del cual se van a empezar a escribir en los registros. Dependiendo de la variable a escribir se tendrá una referencia u otra.

2-2) “Word Count” (1-100) (2bytes): Indica el número de registros a escribir. En nuestro caso serán uno o dos registros, ya que de las tres variables a escribir dos de ellas ocupan dos registros y la otra uno.

2-3)”Byte Count” (1byte): Indica el número de bytes a escribir. Su valor es 2x (Word Count).

2-4) “Register Values”: Valor de las variables a escribir.

3) “Length Field”: este valor oscila entre 9 y 11 dependiendo de la variable a escribir, ya que tienen distinto tamaño.

-Variable Estado (2bytes): 0x0009

-Variables Intensidad de Pila y Potencia de referencia (4bytes): 0x000B

Las tramas completas para las tres variables de escritura son:

Intensidad de pila

Transaction ID	Protocolo ID	Length Field	Slave address	Function code	Referente number	Word Count	Byte Count	Register Values
0000	0000	000B	01	10	0010	0002	04	Datos (4bytes)
En Hexadecimal								

Potencia de referencia:

Transaction ID	Protocolo ID	Length Field	Slave address	Function code	Referente number	Word Count	Byte Count	Register Values
0000	0000	000B	01	10	0012	0002	04	Datos (4bytes)
En Hexadecimal								

Estado:

Transaction ID	Protocolo ID	Length Field	Slave address	Function code	Referente number	Word Count	Byte Count	Register Values
0000	0000	0009	01	10	000F	0001	02	Datos (2bytes)
En Hexadecimal								

5.3.2.2 Trama de respuesta a escritura

1)” Function Code” (1byte): 0x10

2) “Data bytes”: Esta compuesto por 2 subcampos:

2-1) “Reference number” (2bytes): Indica el registro de referencia a partir del cual se ha empezado a escribir.

2-2)”Word Count” (2bytes): Indica el número de registros que se han escrito.

3) Length Field: Son 6 bytes.

Las tramas completas para las tres variables de escritura son:

Intensidad de Pila:

Transaction ID	Protocolo ID	Length Field	Slave address	Function code	Referente number	Word Count
0000	0000	0006	01	10	0010	0002
En hexadecimal						

Potencia de referencia:

Transaction ID	Protocolo ID	Length Field	Slave address	Function code	Referente number	Word Count
0000	0000	0006	01	10	0012	0002
En hexadecimal						

Estado:

Transaction ID	Protocolo ID	Length Field	Slave address	Function code	Referente number	Word Count
0000	0000	0006	01	10	000F	0001
En hexadecimal						

5.3.2.3 Excepciones en trama de escritura

Al igual que en las tramas de lectura se pueden producir errores tipo 1 y tipo 2. También, el código de función de esta trama es el resultado de sumarle ochenta en hexadecimal al código de función original. El campo de datos estará igualmente constituido por un byte con valor cero o uno dependiendo del tipo de error:

Error tipo 1

Transaction ID	Protocolo ID	Length Field	Slave address	Function code	Exception Code
0000	0000	0003	01	90	01
En hexadecimal					

Error tipo 2:

Transaction ID	Protocolo ID	Length Field	Slave address	Function code	Exception Code
0000	0000	0003	01	90	02
En hexadecimal					