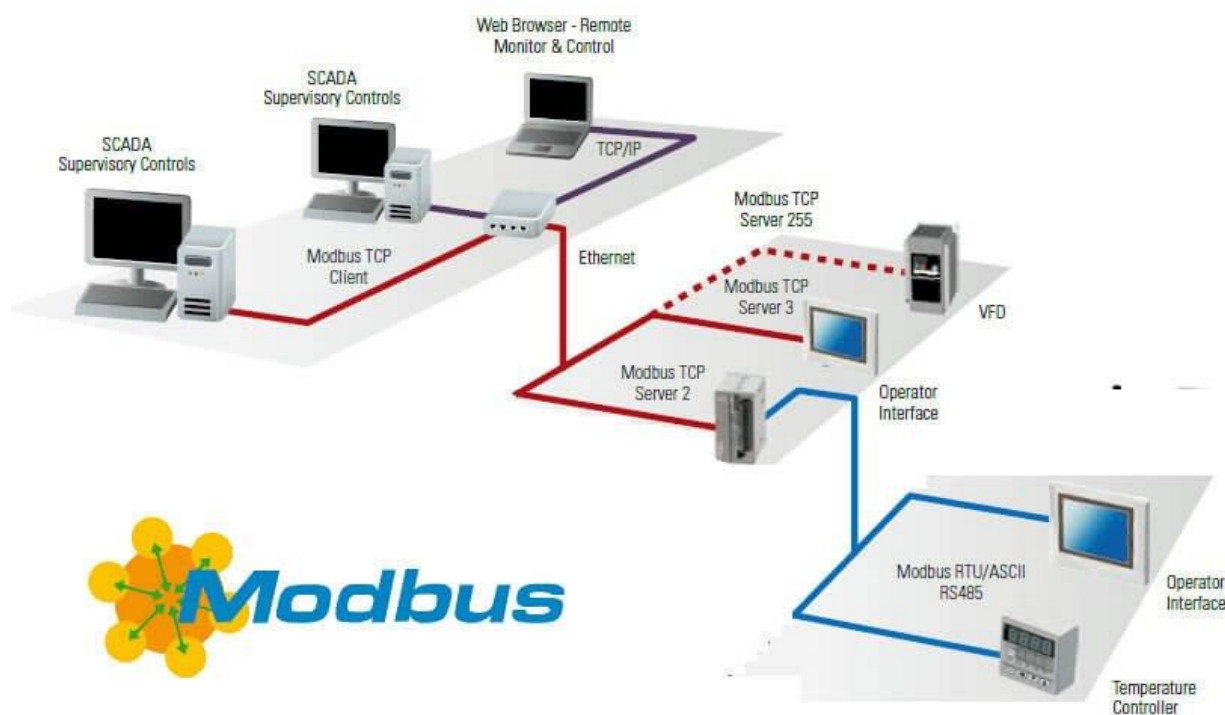


MODBUS là gì?

MODBUS là một chuẩn giao thức truyền thông công nghiệp được phát hành và phát triển bởi MODICON vào năm 1979, và chính thức thuộc về Schneider Electric vào năm 1996. MODBUS đã nhanh chóng trở thành tiêu chuẩn truyền thông trong các ngành công nghiệp tự động hóa bởi tính ổn định, dễ dàng, thuận tiện và đặc biệt hơn nữa là MIỄN PHÍ và hiện được duy trì bởi tổ chức “modbus.org”.

Nguyên tắc hoạt động của MODBUS

MODBUS hoạt động theo nguyên tắc “Master – Slave” hay còn gọi là “Chủ – Tớ”. Một Master có thể kết nối được với một hay nhiều “Slave”. “Master” thường là PLC, PC, DCS, RTU hay SCADA. “Slave” thường là các thiết bị cấp hiện trường. Nói một cách dễ hiểu, nó là một phương pháp được sử dụng để truyền thông tin qua đường dây nối tiếp giữa các thiết bị điện tử. Thiết bị yêu cầu thông tin được gọi là Modbus Master và thiết bị cung cấp thông tin là Modbus Slaves. Trong mạng Modbus tiêu chuẩn, có một Master và tối đa 247 Slave, mỗi Slave có một địa chỉ Slave duy nhất từ 1 đến 247. Master cũng có thể ghi thông tin vào các Slave.



Phân loại chuẩn MODBUS

Hiện nay, MODBUS được biết đến và sử dụng phổ biến trong công nghiệp gồm 3 chuẩn: MODBUS RTU, MODBUS TCP và MODBUS ASCII.

Modbus RTU

Dữ liệu được mã hóa theo hệ nhị phân, và chỉ cần một byte truyền thông cho một byte dữ liệu. Đây là giao thức truyền thông lý tưởng đối với RS232 hay RS485, tốc độ từ 1200 đến 115000 baud. Tốc độ phổ biến nhất là từ 9600 đến 19200 baud. MODBUS RTU là giao thức truyền thông công nghiệp được sử dụng rộng rãi nhất, do đó hầu như trong bài viết này sẽ tập trung đề cập đến MODBUS RTU.

Modbus TCP

MODBUS TCP là MODBUS qua Ethernet (RJ45). Với MODBUS TCP, dữ liệu MODBUS được tóm lược đơn giản trong một gói TCP/IP. Nói một cách đơn giản, đây như là một thông điệp của Modbus RTU được truyền bằng trình bao bọc TCP/IP và được gửi qua mạng thay vì các đường nối tiếp. Máy chủ không có SlaveID vì nó sử dụng địa chỉ IP.

Trong đó:

TCP (Transmission Control Protocol) là giao thức điều khiển đường truyền và IP (Internet Protocol) là giao thức Internet. Các giao thức này được sử dụng cùng nhau và là giao thức truyền tải cho internet. Khi thông tin modbus được gửi bằng các giao thức này, dữ liệu được chuyển tới TCP nơi thông tin bổ sung được đính kèm và cấp cho IP. IP sau đó đặt dữ liệu trong một gói (hoặc gói dữ liệu) và truyền nó.

TCP phải được thiết lập kết nối trước khi truyền dữ liệu, vì nó là giao thức dựa trên kết nối. Master (hoặc Client trong Modbus TCP) thiết lập kết nối với Slave (hoặc Server). Server chờ một kết nối đến từ Client. Sau khi kết nối được thiết lập, Server sẽ phản hồi các truy vấn từ Client cho đến khi Client ngắt kết nối.

Modbus ASCII

Mọi thông điệp được mã hóa bằng hexadecimal, sử dụng đặc tính ASCII 4 bit. Đối với mỗi một byte thông tin, cần có 2 byte truyền thông, gấp đôi so với MODBUS RTU hay MODBUS TCP. Tuy nhiên, MODBUS ASCII chậm nhất trong số 3 loại giao thức, nhưng lại phù hợp với modem điện thoại hay kết nối sử dụng sóng radio bởi ASCII sử dụng các tính năng phân định thông điệp. Nhờ tính năng phân định này, mọi rắc rối trong phương tiện truyền dẫn sẽ không làm thiết bị nhận dịch sai thông tin. Điều này rất quan trọng khi đề cập đến các giao thức truyền thông cho các modem cần độ chính xác thông tin cao, điện thoại di động, nhiều âm thanh hay các phương tiện truyền thông chuyên dụng khác.

Mã HEXA (HEXADECIMAL)

Khi quá trình khắc phục sự cố, khi xem dữ liệu thô thực tế đang được truyền. Các chuỗi dài gồm số 1 và số 0 rất khó đọc, vì vậy các bit được kết hợp và hiển thị ở dạng thập lục phân. Mỗi khối 4 bit được biểu diễn bằng một trong mười sáu ký tự từ 0 đến F.

0000 = 0	0100 = 4	1000 = 8	1100 = C
----------	----------	----------	----------

0001 = 1	0101 = 5	1001 = 9	1101 = D
0010 = 2	0110 = 6	1010 = A	1110 = E
0011 = 3	0111 = 7	1011 = B	1111 = F

Mỗi khối 8 bit (gọi là byte) được biểu diễn bằng một trong 256 cặp ký tự từ 00 đến FF.

ASCII là gì?

ASCII là viết tắt của “American Standard Code for Information Interchange”. Theo cách tương tự, cứ 4 bit có thể được kết hợp và biểu diễn bằng một trong mười sáu ký tự thập lục phân từ 0 đến F, cứ 8 bit (mỗi byte) có thể được kết hợp và biểu diễn bằng một trong 256 ký tự ASCII, bao gồm các ký tự bàn phím chung. Ví dụ: một số giá trị cho các ký tự ASCII là ...

decimal (base10)	binary (base2)	Hex (base16)	ASCII (base256)
0	0000 0000	00	null
1	0000 0001	01	“
34	0010 0010	22	#
35	0010 0011	23	\$
36	0010 0100	24	%
47	0010 1111	2F	/
48	0011 0000	30	0
49	0011 0001	31	1
56	0011 1000	38	8
57	0011 1001	39	9
58	0011 1010	3A	:
64	0100 0000	40	@
65	0100 0001	41	A
66	0100 0010	42	B

89	0101 1001	59	Y
90	0101 1010	5A	Z
91	0101 1011	5B	[
95	0101 1111	5F	—
96	0110 0000	60	`
97	0110 0001	61	a
122	0111 1010	7A	z
123	0111 1011	7B	{
174	1010 1110	AE	®
255	1111 1111	FF	

Địa chỉ dữ liệu và thanh ghi theo chuẩn MODBUS

Thông tin dữ liệu được lưu trữ trong thiết bị Slave được chia trong 4 khoảng giá trị khác nhau. Hai khoảng lưu trữ các giá trị rời rạc on/off (coils) và hai khoảng lưu trữ giá trị số (register – thanh ghi). Mỗi coils và register đều có khoảng biến chỉ đọc (read-only) và biến đọc và ghi (read-write).

- Mỗi khoảng có 9999 biến giá trị
- Mỗi coil hoặc contact là 1 bit và được gán một địa chỉ dữ liệu trong khoảng từ 0000 đến 270E
- Mỗi register là 1 word = 16 bit = 2 bytes và cũng được gán một địa chỉ dữ liệu từ 0000 đến 270E

Coil/Register Numbers	Data Addresses	Type	Table Name
1-9999	0000 to 270E	Read-Write	Discrete Output Coils
10001-19999	0000 to 270E	Read-Only	Discrete Input Contacts
30001-39999	0000 to 270E	Read-Only	Analog Input Registers
40001-49999	0000 to 270E	Read-Write	Analog Output Holding Registers

Coil/Register Numbers có thể được coi như tên vị trí vì chúng không xuất hiện trong các thông điệp thực tế. “Data Addressses” được sử dụng trong các thông điệp truyền tải (truy xuất dữ liệu).

Ví dụ: Holding Register có số là 40001, có “Data Address” là 0000. Sự khác biệt giữa hai giá trị này là độ lệch. Mỗi bảng có một độ lệch khác nhau. 1, 10001, 30001 và 40001.

Function code

Byte thứ hai được “Master” gửi đi là “Function code”. Con số này cho “Slave” biết được rằng, địa chỉ nào cần truy cập để đọc hay ghi giá trị.

Function Code	Action	Table Name
01 (01 hex)	Read	Discrete Output Coils
05 (05 hex)	Write single	Discrete Output Coil
15 (0F hex)	Write multiple	Discrete Output Coils
02 (02 hex)	Read	Discrete Input Contacts
04 (04 hex)	Read	Analog Input Registers
03 (03 hex)	Read	Analog Output Holding Registers
06 (06 hex)	Write single	Analog Output Holding Register
16 (10 hex)	Write multiple	Analog Output Holding Registers

Lệnh và phản hồi trong chuẩn MODBUS

Click vào các liên kết trong bảng để xem ví dụ về các yêu cầu và phản hồi

Data Addresses	Read	Write Single	Write Multiple
Discrete Output Coils 0xxxx	FC01	FC05	FC15
Discrete Input Contacts 1xxxx	FC02	NA	NA
Analog Input Registers 3xxxx	FC04	NA	NA
Analog Output Holding Registers 4xxxx	FC03	FC06	FC16

Kiểu dữ liệu

Ví dụ: FC03 cho thấy register 40108 chứa AE41 chuyển sang 16 bit là 1010 1110 0100 0001.

Register 40108 có thể được hiểu là bất kỳ kiểu dữ liệu 16 bit nào sau đây:

1. Một số nguyên không dấu 16 bit (một số nguyên không dấu nằm từ 0 đến 65535): Register 40108 chứa AE41 = 44,609 (chuyển từ hệ lục phân sang hệ thập phân)
2. Một số nguyên có dấu 16 bit (một số nguyên có dấu nằm từ -32768 đến 32767): AE41 = -20,927 (chuyển đổi từ hệ lục phân sang hệ thập phân không lặp lại, nếu nó quá 32767 thì trừ đi 65535)
3. Một chuỗi ASCII hai ký tự (2 chữ cái đã nhập): AE41 = ® A
4. Một giá trị on/off rời rạc (giá trị này hoạt động giống như số nguyên 16 bit có giá trị 0 hoặc 1. Dữ liệu hex sẽ là 0000 hoặc 0001)

Register 40108 cũng có thể được kết hợp với 40109 để tạo thành bất kỳ kiểu dữ liệu 32 bit nào sau đây:

1. Một số nguyên không dấu 32 bit (một số từ 0 đến 4,294,967,295): 40108.40109 = AE41 5652 = 2.923.517.522
2. Một số nguyên có dấu 32 bit (một số từ -2,147,483,648 đến 2,147,483,647) AE41 5652 = -1,371,449,774
3. Một số dấu phẩy động IEEE 32 bit. Đây là một công thức toán học cho phép bất kỳ số thực nào (một số có dấu thập phân) được biểu diễn bằng 32 bit với độ chính xác khoảng bảy chữ số. AE41 5652 = -4.395978 E-11
4. Một chuỗi ASCII bốn ký tự (4 chữ cái đã nhập) AE41 5652 = ® A V R. Nhiều Register hơn có thể được kết hợp để tạo thành chuỗi ASCII dài hơn. Mỗi Register được sử dụng để lưu trữ hai ký tự ASCII (2 byte)

Byte và Word trong chuẩn MODBUS

Chuẩn Modbus không xác định chính xác cách dữ liệu được lưu trữ trong các thanh ghi. Do đó, một số nhà sản xuất đã triển khai tích hợp chuẩn modbus trong thiết bị của họ để lưu trữ và truyền byte cao hơn đầu tiên sau đó là byte thấp hơn. (AE trước 41). Ngoài ra, cũng có những nhà sản xuất khác lưu trữ và truyền byte thấp hơn trước và sau đó là byte cao hơn (41 trước AE).

Tương tự, khi các thanh ghi được kết hợp để sử dụng các kiểu dữ liệu 32 bit. Một số thiết bị lưu trữ và truyền 16 bit cao hơn (word cao) trong thanh ghi đầu tiên và word thấp hơn trong thanh ghi thứ hai (AE41 trước 5652) trong khi những thiết bị khác làm ngược lại (5652 trước AE41)

Không quan trọng thứ tự byte hoặc word được gửi đi, miễn là thiết bị nhận biết được là làm cách nào để hoạt động chính xác theo yêu cầu.

Ví dụ: nếu số 2.923.517.522 được gửi dưới dạng số nguyên 32 bit không dấu, nó có thể được sắp xếp theo bất kỳ cách nào trong số bốn cách này.

1. AE41 5652
2. 5652 AE41
3. 41AE 5256
4. 5256 41AE

Mở rộng thanh ghi (Register) trong chuẩn MODBUS

Phạm vi của các thanh ghi giữ đầu ra tương tự là 40001 đến 49999, nó ngụ ý rằng không thể có nhiều hơn 9999 thanh ghi. Mặc dù điều này thường là đủ cho hầu hết các ứng dụng, nhưng có những trường hợp cần thêm nhiều địa chỉ thanh ghi nữa.

Các thanh ghi 40001 đến 49999 tương ứng với địa chỉ dữ liệu từ 0000 đến 270E. Nếu chúng ta sử dụng các địa chỉ dữ liệu còn lại từ 270F đến FFFF, thì số thanh ghi có sẵn sẽ gấp hơn sáu lần, tổng cộng là 65536. Điều này sẽ tương ứng với các địa chỉ thanh ghi từ 40001 đến 105536.

Chú ý: nhiều trình điều khiển phần mềm sử dụng modbus (PC Master) được viết với giới hạn địa chỉ từ 40001 đến 49999 và không thể truy cập các địa chỉ thanh ghi mở rộng trong các thiết bị "Slave". Và nhiều thiết bị "Slave" không hỗ trợ sử dụng các địa chỉ thanh ghi mở rộng. Mặt khác, một số thiết bị "Slave" hỗ trợ các thanh ghi mở rộng này và một số phần mềm "Master" có thể truy cập được nó, đặc biệt nếu phần mềm được viết trên nền tảng có thể tùy chỉnh.

Địa chỉ "Slave" 2-byte

Bởi một byte đơn thường được sử dụng để xác định địa chỉ "Slave" và mỗi "Slave" trên mạng yêu cầu một địa chỉ duy nhất, nên số "Slave" trên mạng được giới hạn ở 256. Giới hạn được xác định trong modbus thậm chí còn thấp hơn và ở mức 247.

Để vượt qua giới hạn này, có thể thực hiện một sửa đổi đối với giao thức để sử dụng hai byte cho địa chỉ. "Master" và "Slave" đều phải được hỗ trợ sửa đổi này. Khi đó có thể chỉ định mở rộng giới hạn về số lượng "Slave" trong mạng lên 65535 bởi hai byte cho địa chỉ.

Theo mặc định, đa số phần mềm hỗ trợ Modbus sử dụng địa chỉ 1 byte. Khi một địa chỉ lớn hơn 255 được nhập vào, phần mềm sẽ tự động chuyển sang định địa chỉ 2 byte và giữ nguyên ở chế độ này cho tất cả các địa chỉ cho đến khi tắt định địa chỉ 2 byte theo cách thủ công.

Ứng dụng chuẩn MODBUS

Modbus là một giao thức mở, điều này có nghĩa là các nhà sản xuất hoàn toàn có thể tích hợp chuẩn Modbus vào thiết bị của họ miễn phí mà không phải trả tiền bản quyền. Nó đã trở thành một tiêu chuẩn giao thức truyền thông trong công nghiệp và đang là phương tiện phổ biến nhất hiện có để kết nối các thiết bị điện tử công nghiệp. Nó được sử dụng rộng rãi bởi nhiều nhà sản xuất trong nhiều ngành công nghiệp. Modbus thường được sử dụng để truyền tín hiệu từ thiết bị đo đạc và điều khiển trở lại bộ điều khiển chính hoặc hệ thống thu thập dữ liệu, ví dụ như hệ thống đo nhiệt độ và độ ẩm và truyền kết quả tới máy tính. Modbus thường được sử dụng để kết nối máy tính giám sát với thiết bị đầu cuối từ xa (RTU) trong hệ thống điều khiển giám sát và thu thập dữ liệu (SCADA). Các phiên bản của giao thức Modbus phổ biến cho các đường nối tiếp (Modbus RTU và Modbus ASCII) và cho Ethernet (Modbus TCP).

Sự khác biệt giữa RTU và TCP

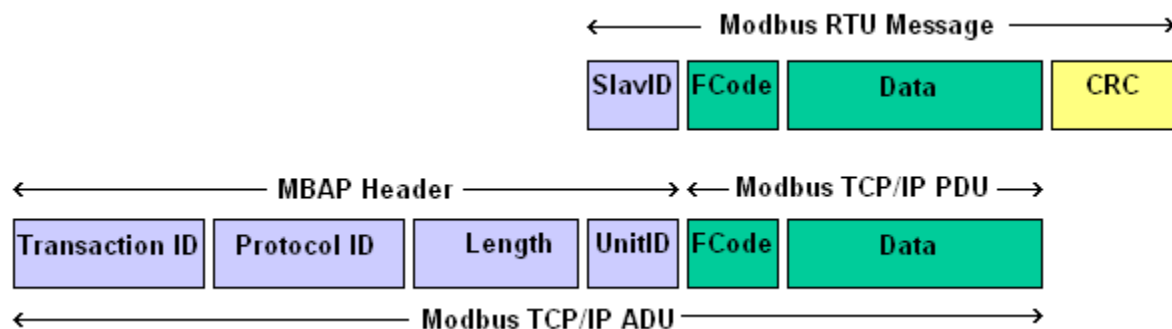
Một tiêu đề 7 byte mới được gọi là MBAP Header (Modbus Application Header) được thêm vào đầu thư. Tiêu đề này có dữ liệu sau:

Mã định danh (Transaction ID): 2 byte do Client đặt để nhận dạng từng yêu cầu duy nhất. Các byte này được lặp lại bởi Server vì các phản hồi của nó có thể không được nhận theo thứ tự như các yêu cầu.

Định dạng giao thức (Protocol ID): 2 byte do Client đặt, luôn luôn = 00 00

Độ dài (Length): 2 byte xác định số byte trong thông điệp cần theo dõi.

Định dạng đơn vị (UnitID): 1 byte được đặt bởi Client và được Server lặp lại để xác định một Slave từ xa được kết nối trên đường truyền nối tiếp hoặc trên các bus khác.



Yêu cầu tương đương với ví dụ về Modbus RTU này:

11 03 006B 0003 7687

Trong đó:

- 11: Địa chỉ SlaveID (17 = 11 hex)

- 03: Function code (đọc thanh ghi giữ đầu ra tương tự (Analog Output Holding Registers))
- 006B: địa chỉ dữ liệu của thanh ghi đầu tiên được yêu cầu. ($40108 - 40001 = 107 = 6B \text{ hex}$)
- 0003: tổng số thanh ghi được yêu cầu. (đọc 3 thanh ghi 40108 đến 40110)
- 7687: CRC (kiểm tra dự phòng theo chu kỳ) để kiểm tra lỗi.

Modbus TCP tương đương sẽ là:

0001 0000 0006 11 03 006B 0003

Trong đó:

- 0001: mã định danh
- 0000: định dạng giao thức
- 0006: độ dài tin nhắn (6 byte)
- 11: định dạng đơn vị ($17 = 11 \text{ hex}$)
- 03: Function code (đọc thanh ghi giữ đầu ra tương tự (Analog Output Holding Registers))
- 006B: địa chỉ dữ liệu của thanh ghi đầu tiên được yêu cầu. ($40108 - 40001 = 107 = 6B \text{ hex}$)
- 0003: tổng số thanh ghi được yêu cầu. (đọc 3 thanh ghi 40108 đến 40110)

Sự khác biệt giữa ASCII và RTU

Ví dụ: cùng phản hồi yêu cầu hiển thị các byte dữ liệu của Registers 40108 đến 40110 từ địa chỉ 17 của Slave

11 03 00 6B 00 03

Yêu cầu từ Modbus ASCII hoàn chỉnh được thực hiện bằng cách thêm các ký tự phân định thông điệp trước. Dấu hai chấm được thêm vào đầu thông điệp, LRC, ký tự xuống dòng và nguồn cấp dữ liệu dòng được thêm vào cuối thông điệp như sau:

: 1 1 0 3 0 0 6 B 0 0 0 3 7 E CR LF

Mỗi ký tự bây giờ được coi là một ký tự ASCII và được thay thế bằng giá trị hex của nó để đưa ra thông điệp cuối cùng.

3A 3131 3033 3030 3642 3030 3033 3745 0D 0A

Kích thước yêu cầu Modbus ASCII này là 17 byte (170 bit)

Thông điệp Modbus RTU tương đương sẽ là:

11 03 00 6B 00 03 76 87

Kích thước yêu cầu Modbus RTU này là 8 byte (80 bit)