

Systemy wbudowane dla automatyki W03

dr inż. Krzysztof Urbański

Dodatkowe materiały do kursu

na prawach rękopisu

Źródła: materiały własne, publicznie dostępne dokumenty w dostępie otwartym oraz noty katalogowe i oficjalna dokumentacja prezentowanych rozwiązań.

Opuszczamy komfortowy świat formatu tekstowego
i przechodzimy do bitowego,
gdzie RS-485 nadal odpowiada za warstwę fizyczną,
ale protokołem łączą danych jest:

DMX512

albo

Modbus RTU

DMX512: 1986; United States Institute for Theatre Technology (USITT) (sic!)
The American National Standards Institute (ANSI) zatwierdziło DMX512 w 2004 r.
Niezbyt “inżynierskie” korzenie, ale może dzięki temu bardzo łatwy w implementacji.

DMX512 od strony sprzętowej

- EIA-485: 250000,8N2
- do 400 m, do 32 urządzeń
- złącza 5-pin XLR (opcja: XLR-3, RJ45)

Budowa ramki DMX512

- Break condition (BREAK, min. 88 us)
- Mark-After-Break (MAB, min. 8 us)
- Slot 0 = Start Code (u nas będzie to 0x00)
- Do 512 slotów danych, po 1B każdy

→ najdłuższa ramka trwa ok. 23 ms co daje 44 Hz, ale można skracać ramki.

1 1.00V/

← 0.00s 50.0μs/

Sngl 1 STOP

START

START CODE IN SLOT 0
FOLLOWED BY UP TO 512 DATA CHANNELS

12
uS

MAB

IDLE

BREAK
100uS

SLOT 0
44uS

SLOT 1
44uS

SLOT 2
44uS

SLOT 3
44uS

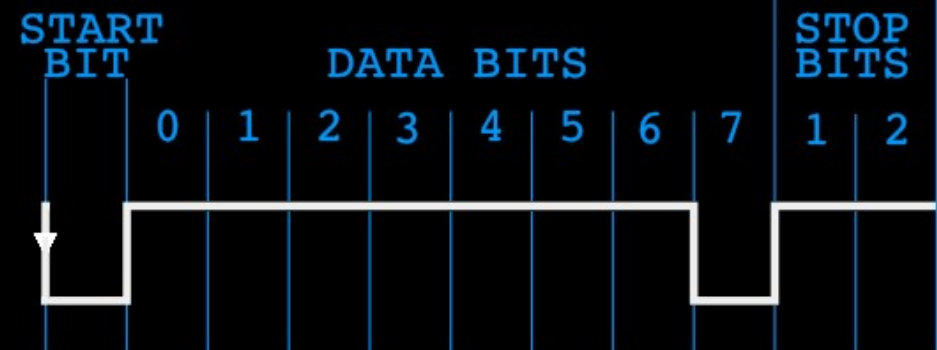
SLOT 4
44uS

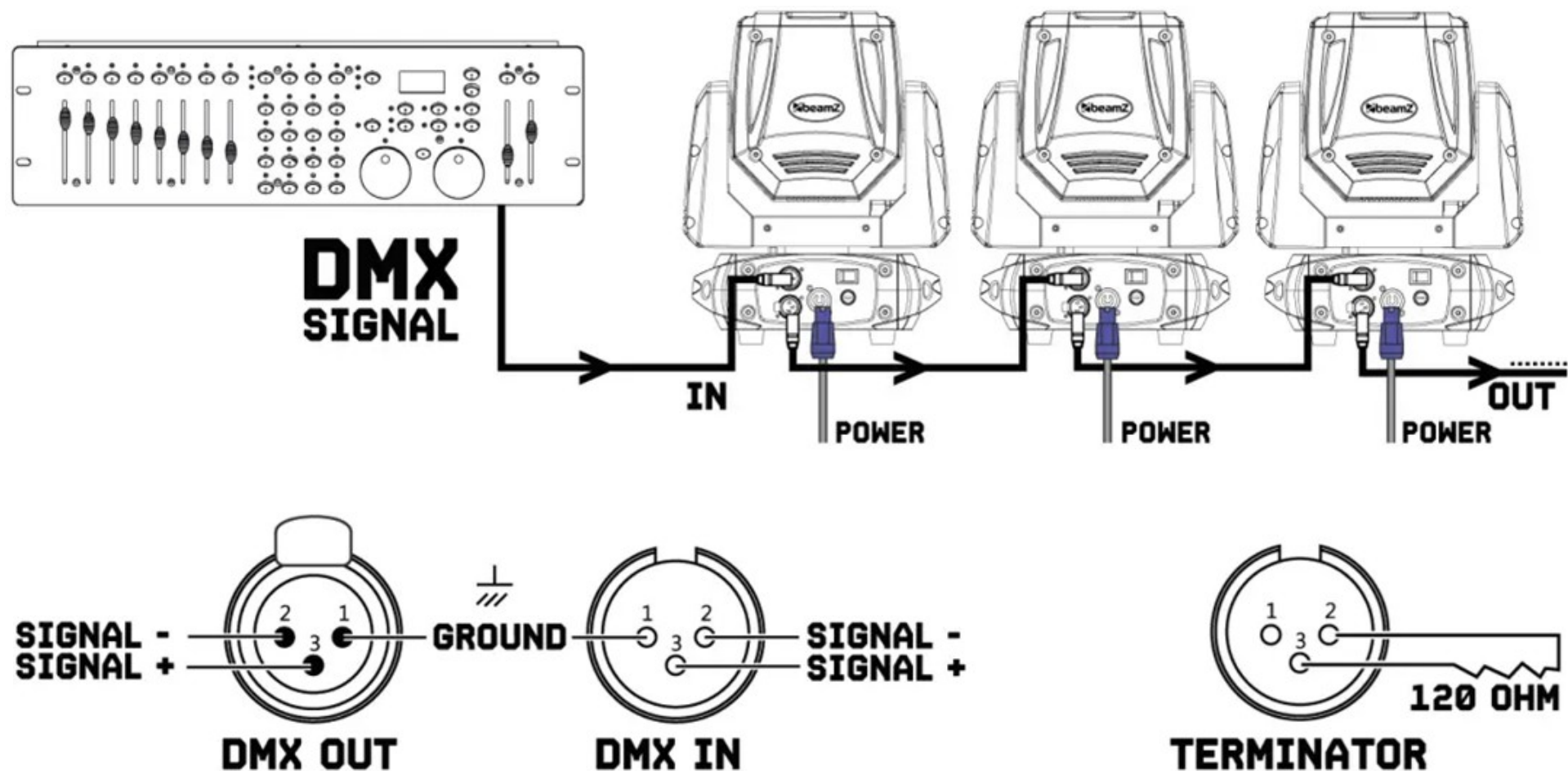
SLOT 5
44uS

SLOT 6
44uS

...

START CODE AND DATA CHANNELS
EACH 11 BITS OF 4 uSec





Branża rozrywkowa używa chętnie złączy XLR. Przewody audio (krótkie) zadziałają z DMX, ale nie jest to zalecane - baudrate 250000 znacznie wykracza poza pasmo gwarantowane okablowaniem audio.

Przykładowy kod w C#, Windows

```
byte[] frame = new byte[1 + 512];
```

```
uart.PortName = "COM10";
```

```
uart.BaudRate = 250000;
```

```
uart.StopBits = StopBits.Two;
```

```
uart.Open();
```

```
uart.BreakState = true; Thread.Sleep(10);
```

```
uart.BreakState = false; Thread.Sleep(1);
```

```
frame[0] = 0x00; # important
```

```
uart.Write(frame, 0, frame.Length);
```

```
dmx.set("dimm", 255); # common dimmer value for all light colors
```

```
dmx.set("red", 255);
```

→demonstracja na wykładzie

Przykładowy kod w C, Pico

```
#define DMX_FRAME_LENGTH (1 + 512)
uint8_t frame[DMX_FRAME_LENGTH];

void dmx_send_frame() {
    frame[0] = 0x00; # important
    gpio_put(RS485_DIR_PIN, RS485_DIR_TX);
    uart_set_break(uart, true); sleep_us(100);
    uart_set_break(uart, false); sleep_us(10);
    uart_write_blocking(uart, frame, DMX_FRAME_LENGTH);
    uart_tx_wait_blocking(uart);
    gpio_put(RS485_DIR_PIN, RS485_DIR_RX);
}

void dmx_set(int32_t base, int32_t reg, uint8_t val) {
    frame[baseaddr + reg - 1] = val;
}

dmx_set(dimm, 255); # common dimmer value for all light colors
dmx_set(red, 255);
```

Przykładowy kod w MicroPython, Pico

```
frame = bytearray(1 + 512)
uart = UART(1, tx=Pin(4), rx=Pin(5))
uart.init(baudrate=250_000, bits=8, parity=None, stop=2)

def dmx_send():
    frame[0] = 0x00 # important
    de_nre.value(1) # direction TX
    uart.init(baudrate=100_000) # work-around, slow-down baudrate for break signal synthesis
    uart.sendbreak() → this approach really sucks, see C or C# implementation
    uart.init(baudrate=250_000) #restore “production” baudrate
    uart.write(frame)
    uart.flush() # similar to uart_tx_wait_blocking() in C
    de_nre.value(0) # switch to RX

def dmx_set(reg, value):
    frame[baseaddr + reg - 1] = value

dmx_set(dimm, 255) # common dimmer value for all light colors
dmx_set(red, 255)
```


Modbus

1979-1997 firma Modicon → Schneider Electric (1997-2004) → od 2004 Modbus Organization, obecnie otwarty protokół.

RTU/ASCII (serial) oraz TCP:502

Na tych zajęciach interesuje nas Modbus RTU korzystający z warstwy fizycznej RS485 (ale mogą to być RS232 i wiele innych)

1 bit startu, 8 bitów danych, 1 bit parzystości (typowo E), 1 bit stopu.

Ramka RTU (remote terminal unit):

Slave address	Function Code	Data	CRC
1 byte	1 byte	0 – 252 bytes	2 bytes: 1 CRC low byte and 1 CRC high byte

Przy czym CRC-16-MODBUS jest opisane wielomianem $x^{16} + x^{15} + x^2 + 1$.

Adres 0 jest rozgłoszeniowy.

Odstępy między ramkami przynajmniej 3.5T.

Kodowanie porządku bajtów: big-endian (bardziej znaczący bajt jest przesyłany wcześniej).

Przykład: wartość 0x1234 jest przesyłana jako bajt 0x12, a następnie 0x34.

Kody funkcji: 1..255, ale w praktyce jest ich niewiele, np.:

1: Read Coils

2: Read Discrete Inputs

5: Write Single Coil

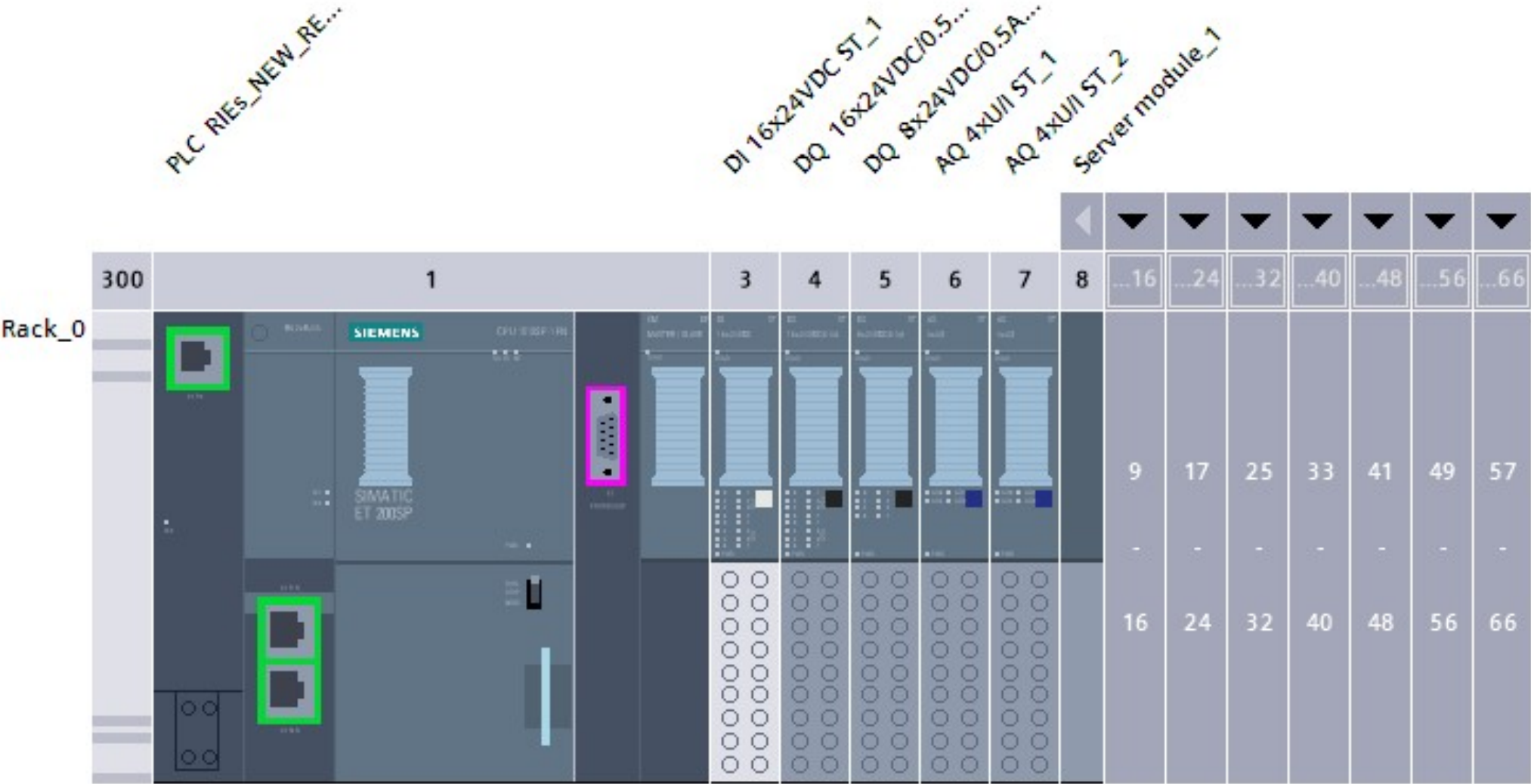
6: Write Single Holding Register

Modbus w automatyce

Realnie? Nie bardzo, są wygodniejsze (ale kosztowne) Profibus (dawniej), obecnie popularny w naszym kraju Profinet.

Mimo to Modbus jest rozwiązaniem wspieranym po stronie PC, PLC, MCU – niechętnie, ale jednak.

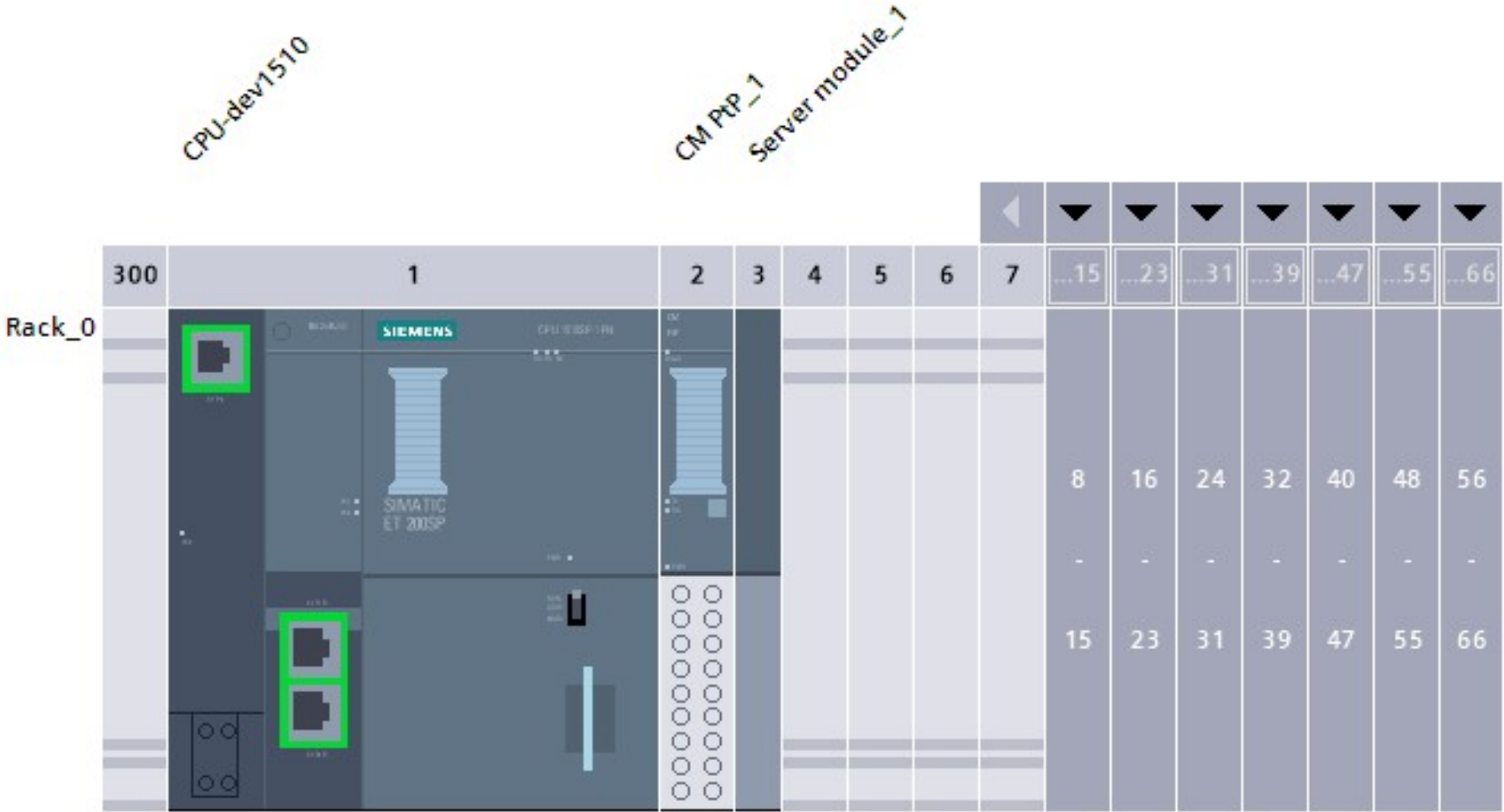
Siemens S7 1500 – typowy moduł PLC



Siemens S7 1500 – typowy moduł RTU



Siemens S7 1500 PLC modułem CM PtP (RS232, RS422, RS485)



Jak Modbus wygląda od strony PLC?

Niezbyt przyjemnie, ale żeby mieć o tym jakieś wyobrażenie...

Interface

Specification of the operating mode

Specification of the operating mode

☐ RS232C

☐ Full duplex (RS422) 4-wire operation (point-to-point)

☐ Full duplex (RS 422) four wire mode (multipoint master)

☐ Full duplex (RS 422) four wire mode (multipoint slave)

☒ Half duplex (RS485) 2-wire operation

Receive line initial state

☐ None

☒ Signal R(A)=0 V, Signal R(B)=5 V

Port configuration

Protocol

Protocol: Freeport/Modbus

The Modbus protocol is configured in the user program using the Modbus_Comm_Load instruction

☐ Performance optimized for many short frames

Port parameters

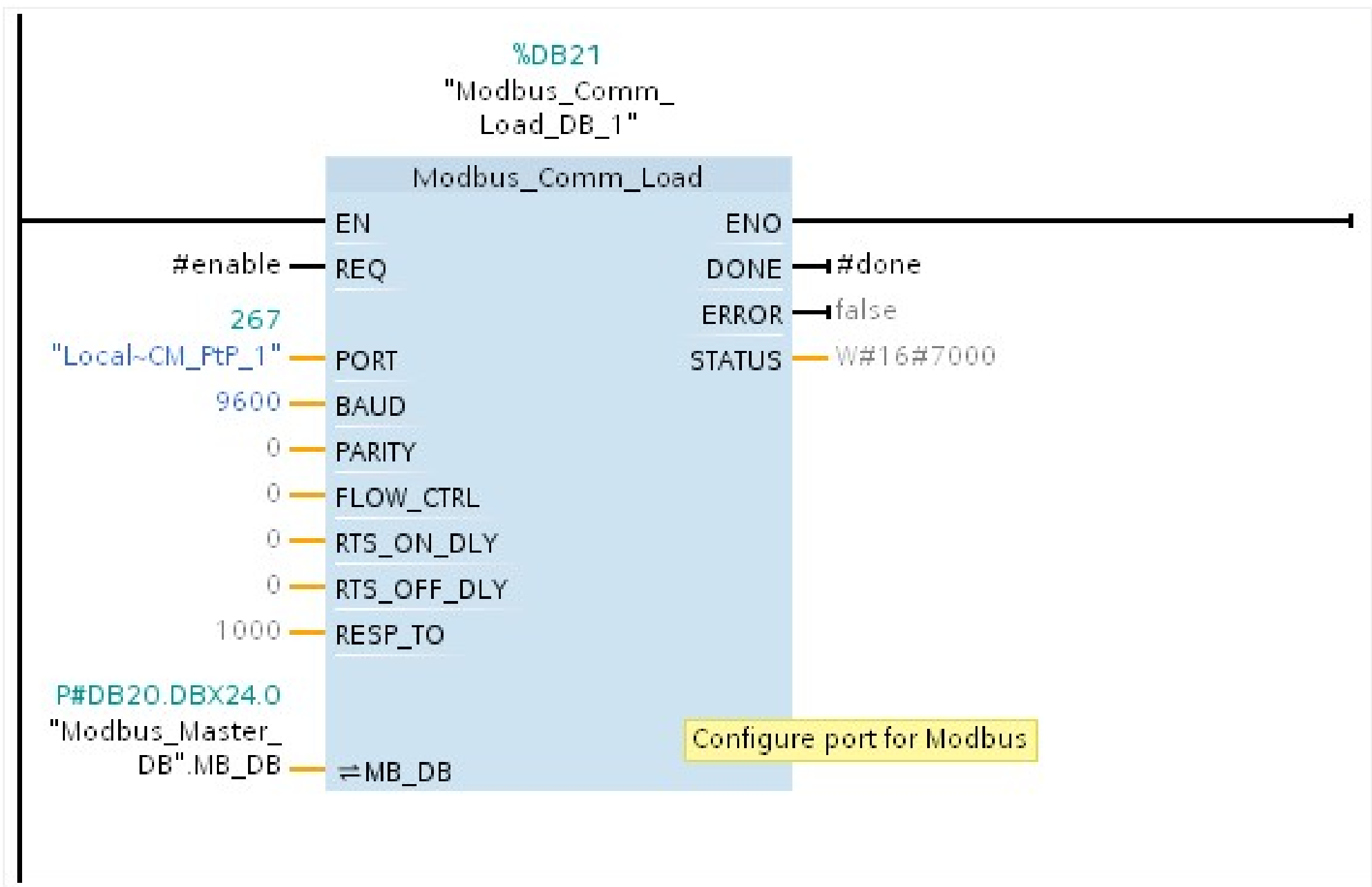
Data transmission rate: 9600

Parity: None

Data bits: 8 bits

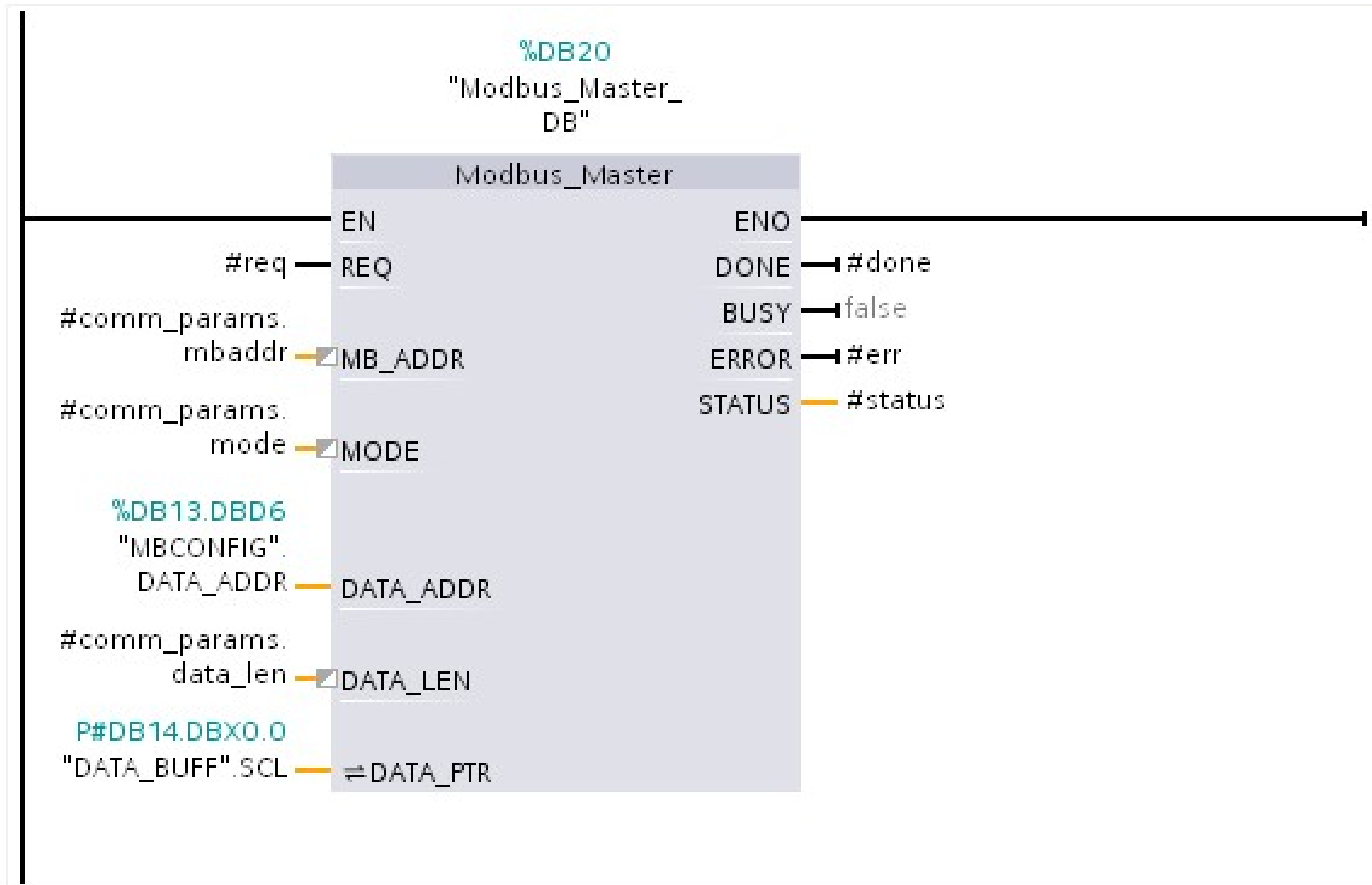
Stop bits: 1

























Comment
















▼ **Network 2:** Modbus Communication Block

Comment

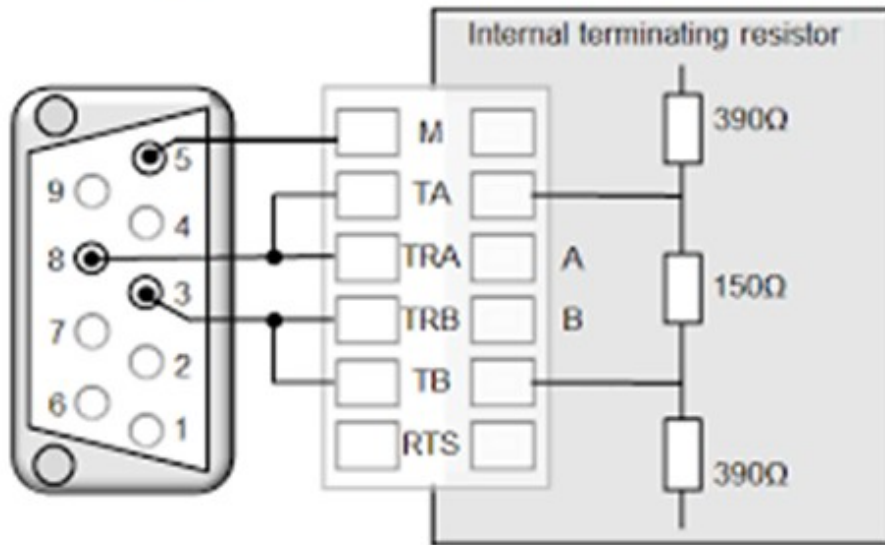


MBCONFIG					
		Name	Data type	Offset	Start value
1		▼ Static			
2		■ REQ	Bool 	0.0	false
3		■ MB_ADDR	UInt	2.0	3
4		■ Static_1	USInt	4.0	1
5		■ DATA_ADDR	UDInt	6.0	40001
6		■ DATA_LEN	UInt	10.0	8
7		■ DONE	Bool	12.0	false
8		■ BUSY	Bool	12.1	false
9		■ ERROR	Bool	12.2	false
10		■ STATUS	Word	14.0	16#0
11		■ AGAIN	Bool	16.0	false
12		■ DONE2	Bool	16.1	false
13		■ BUSY2	Bool	16.2	false
14		■ ERROR2	Bool	16.3	false
15		■ MB_ADDR2	UInt	18.0	3
16		■ MODE2	USInt	20.0	1
17		■ DATA_ADDR2	UDInt	22.0	40050
18		■ DATA_LEN2	UInt	26.0	2
19		■ STATUS2	Word	28.0	16#0
20		■ REQ2	Bool	30.0	false
21		■ count	Bool	30.1	false
22		■ VAL1	Bool	30.2	false
23		■ VAL2	Bool	30.3	false

MBLOAD					
		Name	Data type	Offset	Start value
1		▼ Static			
2		■ BAUD	UDInt 	0.0	19200
3		■ PARITY	UInt	4.0	0
4		■ DONE	Bool	6.0	false
5		■ ERROR	Bool	6.1	false
6		■ STATUS	Word	8.0	16#0
7		■ BAUD2	UDInt	10.0	19200
8		■ PARITY2	UInt	14.0	0
9		■ DONE2	Bool	16.0	false
10		■ ERROR2	Bool	16.1	false
11		■ STATUS2	Word	18.0	16#0
12		■ REQ	Bool	20.0	true

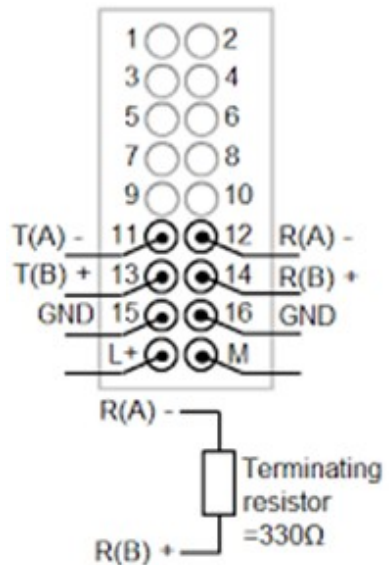
W S7-1200 spotkacie taki moduł CM1241:

9-pin Sub-D-socket CB 1241
(Terminal assignment for RS485 communication)

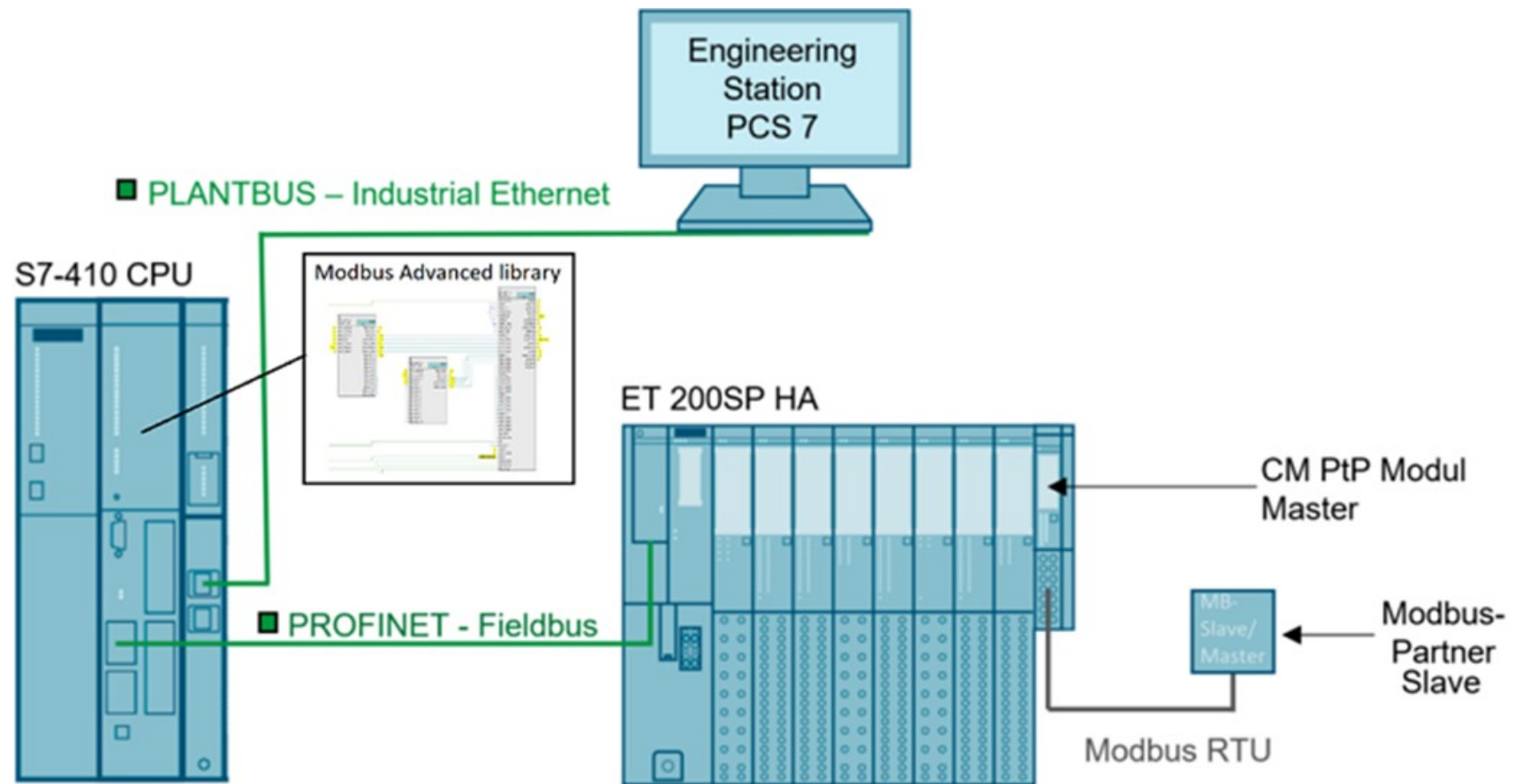


A z kolei RS422 (full-duplex) na CM PtP łączymy w ten sposób:

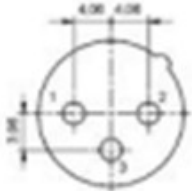
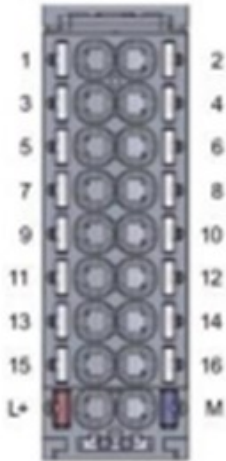
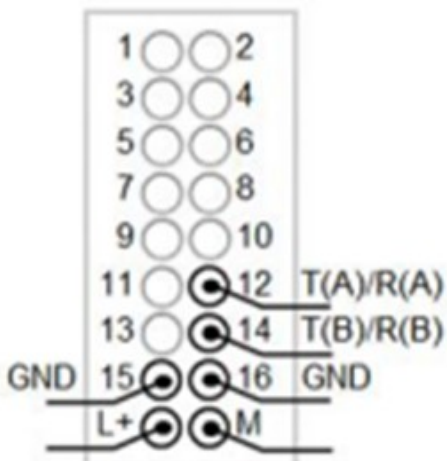

ET 200SP CM PtP
(Terminal assignment for RS422 communication)



Typowa struktura sytemu sterowania: PLC, wyspa (ET200) z modułem CM PtP, Modbus Slave

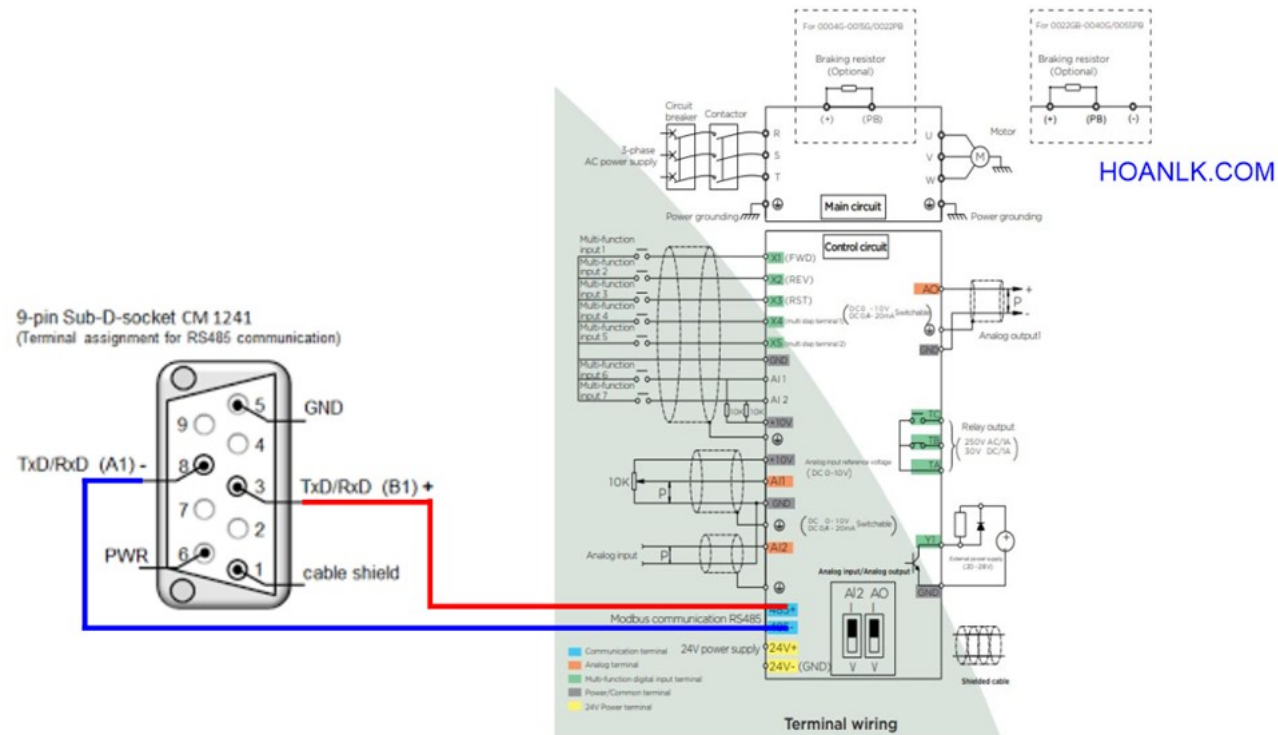


Ciekawostka – PLC może też sterować urządzeniami DMX512:

XLR-Type	XLR-Connector	ET 200SP BaseUnit	Pin assignment for RS485
3-polig	 <p>1: Ground (GND) 2: Signal inv. (DMX-) 3: Signal (DMX+)</p>		 <p>12: T(A)/R(A) Signal inv. (DMX-) 14: T(B)/R(B) Signal (DMX+) 15: Ground (GND) 16: Ground (GND)</p>
5-polig	 <p>1: Ground (GND) 2: Signal inv. (DMX-) 3: Signal (DMX+) 4: optional Data 2 - 5: optional Data 2 +</p>	<p>12: Signal inv. (DMX-) 14: Signal (DMX+) 15: Ground (GND) 16: Ground (GND)</p>	

Spotykane zastosowanie Modbus w automatyce: falownik + PLC

Dzisiaj raczej niechętnie używane – łatwiejsze w obsłudze (i droższe) są falowniki np. z ProfiNet.



Ten niepozorny moduł to właśnie CM PtP, dodający komunikację RS232/RS422/RS485 do PLC Siemens serii np. S7-1200, S7-1500

