



Aplikacje sterowników przemysłowych 1

Protokół komunikacyjny Modbus



WEAil



PSk

Paweł Strączyński

pstraczynski@tu.kielce.pl

Katedra Urządzeń Elektrycznych i Automatyki

Protokoły komunikacji w przemyśle

Obecnie w środowisku przemysłowym wykorzystywanych jest wiele **protokołów** określających zasady wymiany informacji między urządzeniami. Konieczność stosowania skutecznych systemów wymiany informacji jest następstwem zdecentralizowanych systemów sterowania.

Aktualnie powszechnie używane w przemyśle urządzenia takie jak sterowniki PLC, panele HMI, falowniki, sterowniki serwomechanizmów itp. wymieniają między sobą informacje w oparciu o komunikację określoną wedle danego **protokołu komunikacji**.

Specyficzne wymagania systemów kontrolno-pomiarowych w przemyśle (niezawodność, szybka, częsta wymiana stosunkowo małych porcji informacji) spowodowały, że ze środowiska tego wyrosły różnego rodzaju **protokoły** które urosły do rangi przemysłowych **standardów**.

Protokoły komunikacji w przemyśle

Wśród dużej ilości protokołów komunikacyjnych wyróżnić można te oparte na **Ethernetcie** jak i wykorzystujące **komunikację szeregową** w oparciu o standardy **RS-422** oraz **RS-485**.

W oparciu o sieć **Ethernet** pracują m.in. bardzo popularne w przemyśle protokoły takie jak:

- ☐ Profinet,
- ☐ Modbus
- ☐ TCP,
- ☐ EtherCAT,
- ☐ Powerlink,
- ☐ DeviceNET,
- ☐ SERCOS III.

Do najbardziej popularnych protokołów przemysłowych opartych o **komunikację szeregową** należą:

- ☐ Profibus DP,
- ☐ Modbus RTU,
- ☐ CanOpen,
- ☐ MPI.

Protokół Modbus

Protokół Modbus stworzony został w 1979 roku przez firmę Modicon i pierwotnie dedykowany był do komunikacji między sterownikami PLC tego producenta.

Jego niezawodność i prostota sprawiły że jest on obecnie w automatyce uznanym standardem i jednym z najchętniej stosowanych protokołów wymiany informacji.

Uznanie zyskał ze względu na prostą zasadę wymiany informacji (*Master-Slave*), wbudowany system potwierdzania komunikatów (*Zapytanie-Odpowiedź*) oraz kontrole poprawności przesyłanych danych.

Wyróżnia się wersje protokołu oparte o łącze szeregowe (**Modbus RTU** oraz **Modbus ASCII**) oraz wersję protokołu opartą na Ethernetie - **Modbus TCP**.

Wszystkie odmiany tego protokołu mają wspólną funkcję i rodzaj organizacji pamięci. Różnią się nieznacznie budową ramek.

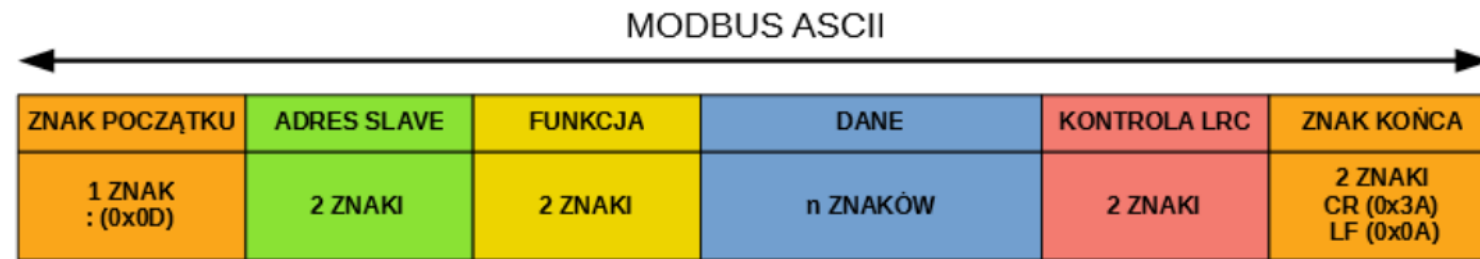
Protokół Modbus

Struktura ramek komunikacyjnych obu szeregowych odmian protokołu (**Modbus RTU** oraz **Modbus ASCII**) jest taka sama.

Składa się z:

- ❑ pola adresowego,
- ❑ pola danych,
- ❑ kontroli błędów
- ❑ oraz znaczników początku i końca ramki.

Różnice widoczne są w samym sposobie przesyłania tych informacji.



Protokół Modbus ASCII

Modbus ASCII to rodzaj protokołu w którym informacje przesyłane są w postaci znaków ASCII - dokładnie każdy bajt informacyjny to dwa znaki ASCII.

Protokół dopuszcza stosowanie dużego odstępu czasowego pomiędzy znakami - nawet do 1 sekundy.

Ramka komunikacyjna zawiera jasno zdefiniowane znaczniki początku i końca:

- ❑ zaczyna się znakiem dwukropka : (0x3A),
- ❑ kończy się znakami (Carriage Return - 0x0D) oraz LF (Line Feed - 0x0A).

Poprawność przesyłania ramki Modbus ASCII badana jest z wykorzystaniem kodu **LRC** (*Longitudinal Redundancy Check*).

Wadą transmisji w trybie ASCII jest nadmiarowość danych koniecznych do przesłania tej samej informacji względem trybu RTU.

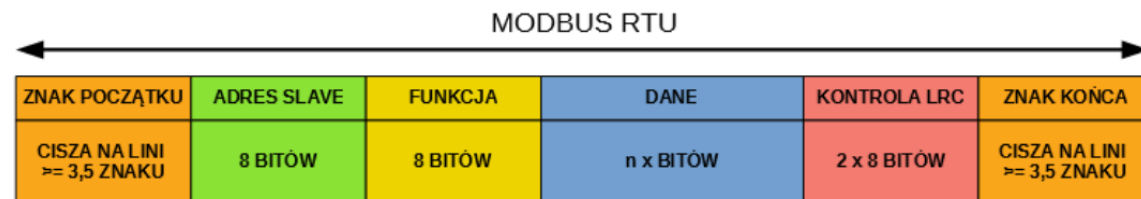
Protokół Modbus RTU

W protokole **Modbus RTU** informacje przesyłane są w postaci binarnej.

Ramki rozpoczyna się zwłoką czasową co najmniej **3,5 razy** dłuższą od czasu trwania pojedynczego znaku. Przerwy pomiędzy nadawaniem kolejnych znaków nie mogą być dłuższe niż **1,5 czasu trwania** każdego z nich znaku.

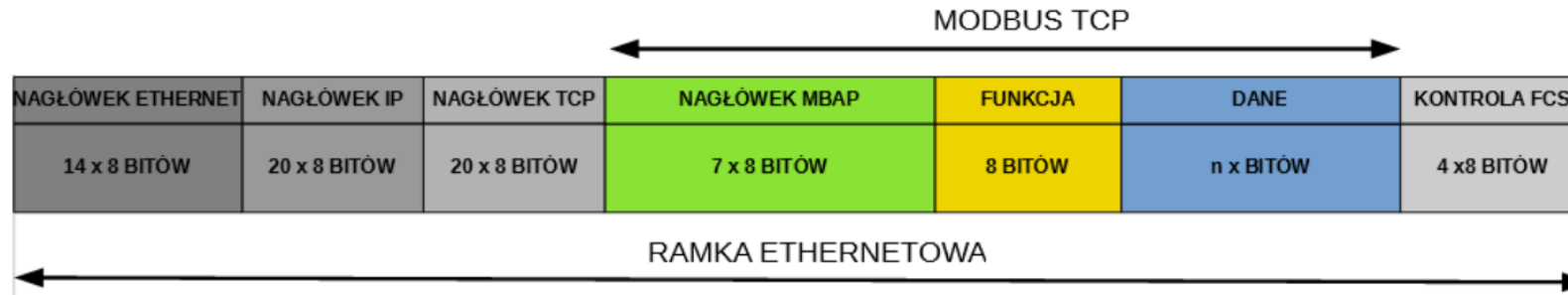
Do weryfikacji poprawności przesyłanych informacji wykorzystywana jest suma kontrolna **CRC** (*Cyclic Redundancy Check*) - **16-bitowe słowo**, przesyłane w postaci dwóch bajtów na końcu ramki.

W przypadku obu odmian protokołu słowo kontrolne jest wyliczane przez urządzenie nadawcze i odbiorcze oraz porównywane. Ewentualne rozbieżności świadczą o występujących przekłamaniach w transmitowanych danych.



Protokół Modbus TCP

W przypadku protokołu **Modbus TCP** ramka protokołu jest osadzona w standardowej ramce ethernetowej.



Rodzaje transakcji

Komunikacja w ramach **protokołu Modbus** zawsze inicjowana jest przez urządzenie typu **Master**. Transakcje można podzielić na:

- ❑ transakcja typu zapytanie - odpowiedź (Unicast),
- ❑ transakcja typu rozgłoszenie (Broadcast).

W transakcji typu zapytanie-odpowiedź urządzenie typu **Master** łączy się z konkretnym urządzeniem typu **Slave**.

W przypadku poleceń dotyczących odczytu stanu bitów lub rejestrów w odpowiedzi **Slave** odsyła odpowiednie informacje.

W przypadku wysłania przez urządzenie **Master** polecenia zapisu do rejestrów, **Slave** po wykonaniu operacji jako potwierdzenie odsyła ramkę do urządzenia typu **Master**.

Przy zapisie większej liczby informacji jest to ramka skrócona zawierająca ilość zapisywanych rejestrów oraz adres pierwszego z nich.

Transakcje typu broadcast

Drugi typ transakcji, a więc transakcja **typu rozgłoszeniowego** adresowana jest do wszystkich urządzeń na linii (zarezerwowanym adresem rozgłoszeniowym jest 0).

Slave'y po wykonaniu poleceń otrzymanych przez Master'a nie wysyłają ramki zwrotnej.

Funkcje i typy obszarów pamięci

Wymiana danych w ramach protokołu odbywa się według ściśle zdefiniowanych funkcji.

Te najczęściej stosowane przedstawiono w tabeli poniżej. Odnoszą się one bezpośrednio do odczytu i zapisu jednego lub wielu obszarów pamięci danego typu.

Kod funkcji	Rodzaj operacji
1	Odczyt wyjść binarnych <i>Read Coils</i>
2	Odczyt wejść binarnych <i>Read Discrete Inputs</i>
3	Odczyt n rejestrów wyjściowych - <i>Read Holding Register</i>
4	Odczyt n rejestrów wejściowych - <i>Read Input Register</i>
5	Zapis pojedynczego wyjścia binarnego <i>Write Single Coil</i>
6	Zapis pojedynczego rejestru wyjściowego - <i>Write Single Register</i>
7	Odczyt statusu - <i>Read Exception Status</i>
8	Test diagnostyczny - <i>Diagnostics</i>
15	Zapis n bitów - <i>Write Multiple Coils</i>
16	Zapis n rejestrów wyjściowych - <i>Write Multiple Register</i>

Funkcje i typy obszarów pamięci

Wymiana danych w ramach protokołu odbywa się według ściśle zdefiniowanych funkcji.

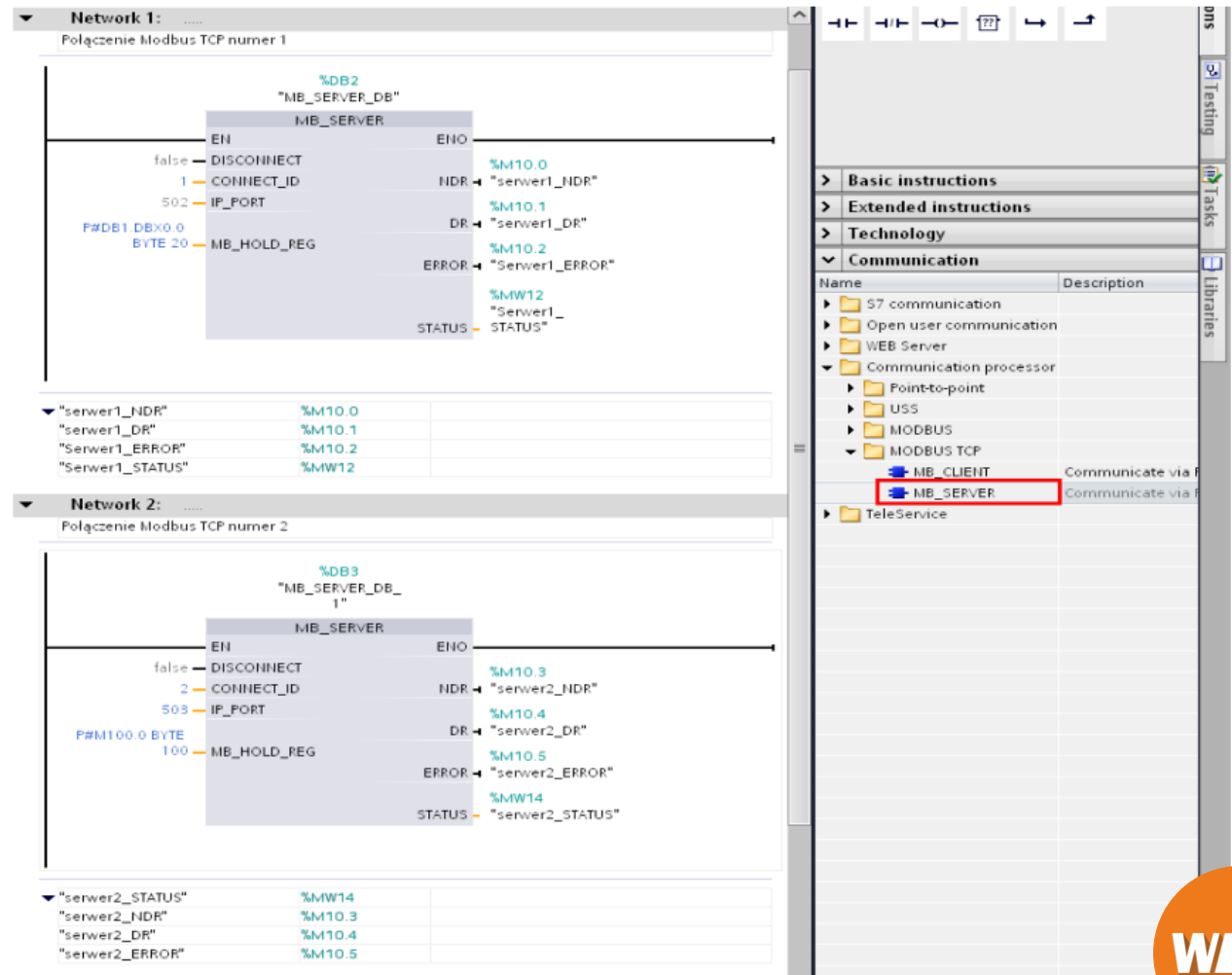
Te najczęściej stosowane przedstawiono w tabeli poniżej. Odnoszą się one bezpośrednio do odczytu i zapisu jednego lub wielu obszarów pamięci danego typu.

Typ zmiennej	Rodzaj	Dostęp	Zastosowanie
Discretes Inputs	1-bit	Odczyt	Dwustanowe wejście
Coils	1-bit	Odczyt/Zapis	Dwustanowe wyjście
Input Register	16-bitowe słowo	Odczyt	Rejestr wejściowy
Holding Register	16-bitowe słowo	Odczyt/Zapis	Rejestr wyjściowy

Serwer Modbus TCP w sterowniku S7-1200

...TCP ▶ Modbus_Server [CPU 1212C AC/DC/Rly] ▶ Program blocks ▶ Data_block_1 [DB1]

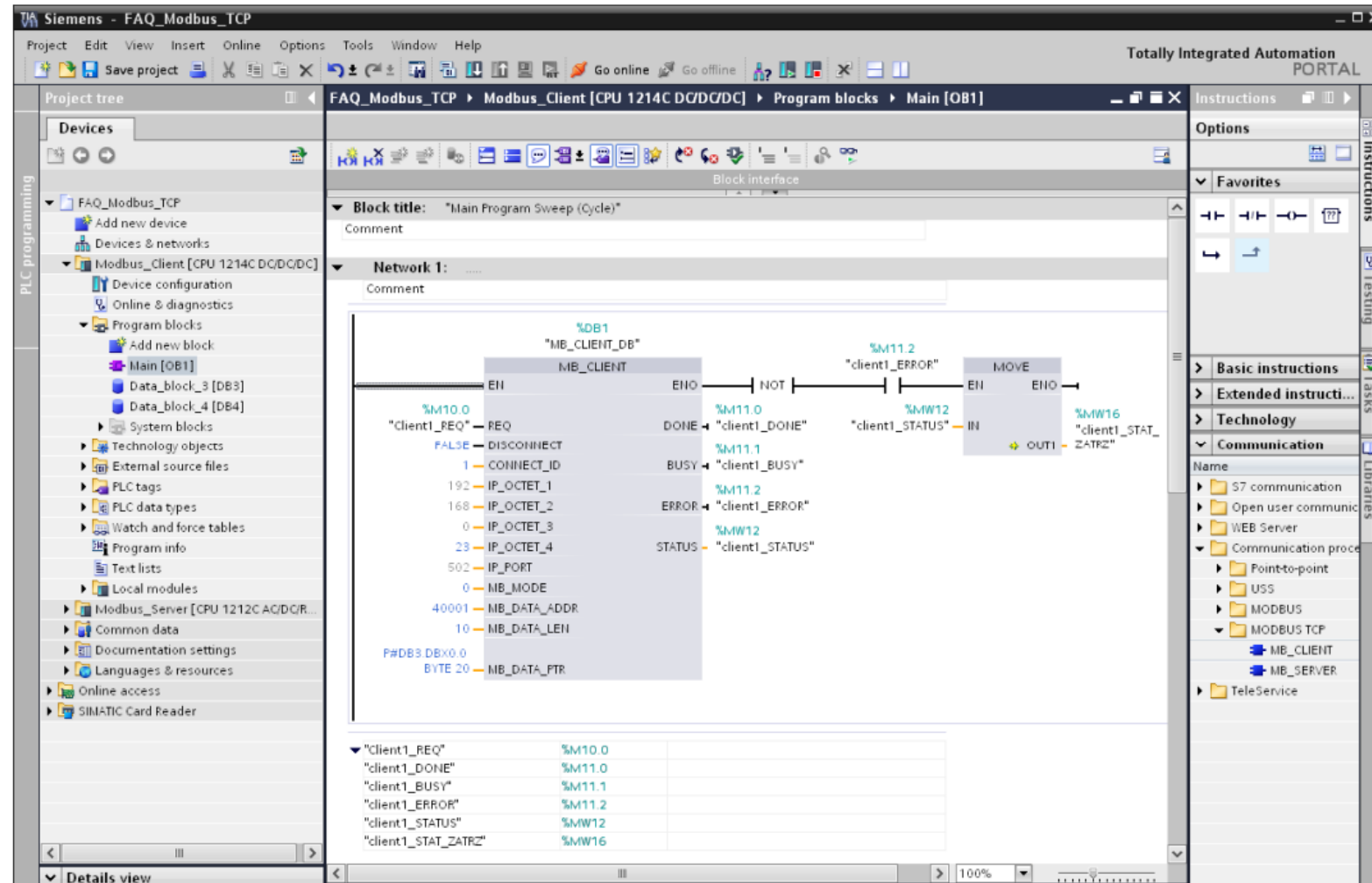
	Name	Data type	Offset	Start value	Retain	Visible in ...	Comm
1	Static						
2	MB_1	Int	0.0	0		☒	
3	MB_2	Int	2.0	0		☒	
4	MB_3	Int	4.0	0		☒	
5	MB_4	Int	6.0	0		☒	
6	MB_5	Int	8.0	0		☒	
7	MB_6	Int	10.0	0		☒	
8	MB_7	Int	12.0	0		☒	
9	MB_8	Int	14.0	0		☒	
10	MB_9	Int	16.0	0		☒	
11	MB_10	Int	18.0	0		☒	
12	<Add new>					☐	



Serwer Modbus TCP w sterowniku S7-1200

Parametr	Typ danych	Opis
DISCONNECT	Bool	„MB_SERVER” próbuje nawiązać pasywne połączenie z partnerem. Oznacza to, że serwer jest w trybie pasywnym i nasłuchuje wywołania. Jeżeli „DISCONNECT” = 0 i połączenie jeszcze nie jest nawiązane, wtedy nowe połączenie może zostać zainicjowane. Jeżeli „DISCONNECT” = 1 i połączenie jest nawiązane, wtedy połączenie jest przerywane. Pozwala to programowi kontrolować połączenie po jego nawiązaniu. Gdy parametr ma wartość „1”, nie można zainicjować połączenia.
CONNECT_ID	UInt	Parametr ten identyfikuje unikalne połączenia wewnątrz PLC. Każdy unikalny blok danych funkcji „MB_CLIENT” lub „MB_SERVER” musi posiadać unikalny numer ID połączenia.
IP_PORT	UInt	Wartość domyślna = 502: Numer portu IP identyfikuje port IP który będzie monitorowany w celu wykrycia zapytania od klienta Modbusa. Poniższe numery TCP portu nie są dozwolone dla pasywnego połączenia „MB_SERVER”: 20, 21, 25, 80, 102, 123, 5001, 34962 oraz 34964.
MB_HOLD_REG	Variant	Wskaźnik do rejestru pamięci serwera: Rejestr pamięci musi być blokiem danych o standardowym dostępie lub obszarem pamięci M. Ten obszar pamięci wykorzystywany jest do przechowywania danych, do których będzie miał dostęp klient Modbusa używając funkcji 3 (read), 6 (write) oraz 16 (write).

Klient Modbus TCP w sterowniku S7-1200



Kody błędów

MB_SERVER

- 8187 – Nieprawidłowy wskaźnik dla parametru „MB_HOLD_REG”: obszar pamięci jest zbyt mały
- 818C - Parametr „DATA_PTR” wskazuje na zoptymalizowany blok danych (musi być standardowy blok DB lub obszar pamięci M)
- 8381 – Nieobsługiwany kod funkcji
- 8382 – Błąd długości danych
- 8383 – Błąd adresu danych lub próba dostępu poza granice określone przez „MB”HOLD_REG”
- 8384 – Błąd wartości danych
- 8385 – Nieobsługiwana wartość kodu diagnostycznego (kod funkcji 08).

MB_SLAVE

- 80C8 – Serwer nie odpowiada w określonym czasie
- 8188 – Nieprawidłowa wartość parametru “MODE”
- 8189 – Nieprawidłowa wartość parametru “DATA_ADDR”
- 818A – Nieprawidłowa wartość parametru „DATA_LEN”
- 818B – Nieprawidłowy wskaźnik obszaru pamięci „DATA_PTR”. Błąd może być spowodowany niepoprawną kombinacją parametrów „MB_DATA_ADDR” oraz „MB_DATA_LEN”
- 818C – Parametr „DATA_PTR” wskazuje na zoptymalizowany blok danych (musi być standardowy blok DB lub obszar pamięci M)
- 8200 – Port jest zajęty przetwarzaniem innego żądania połączenia
- 8380 – Otrzymana ramka Modbusa jest zniekształcona lub otrzymano zbyt dużą ilość bajtów
- 8387 – Przypisany numer ID połączenia jest różny od ID z poprzedniego zapytania. Może być tylko jeden numer ID połączenia użyty wewnątrz przypisanego do „MB_CLIENT” bloku danych DB
- 8388 – Serwer Modbus zwrócił ilość danych inną niż był odpytany. Tyczy się tylko funkcji 15 i 16.