


Przemysłowe Sieci Informatyczne (PSI)

Protokół MODBUS

Politechnika Gdańska
Wydział Elektrotechniki i Automatyki
Kierunek: Automatyka i Robotyka
Studia stacjonarne I stopnia: rok II, semestr IV



Opracowanie: dr inż. Tomasz Rutkowski
Katedra Inżynierii Systemów Sterowania

Czym jest MODBUS ?

- ▶ Protokół MODBUS został opracowany przez firmę Modicon – 1979 rok
 - ▶ Modicon został przejęty przez Schneider Automation
 - ▶ Schneider Electric która aby zapewnić otwartość protokołu Modbus, w 2004 r. przeniosła prawa autorskie do niedochodowej organizacji Modbus-IDA
 - ▶ Protokół MODBUS jest protokołem otwartym (*inne firmy bez dodatkowych opłat mogą implementować go w swoich urządzeniach*)
 - ▶ MODBUS został przyjęty jako standard inżynierski przez wielu producentów urządzeń automatyki systemowej
-

Czym jest MODBUS ?

- ▶ MODBUS jest protokołem komunikacyjnym
- ▶ MODBUS wedle specyfikacji lokuje się w następujących warstwach, warstwowego modelu ISO/OSI:
 - ▶ warstwa siódma (warstwa aplikacji MODBUS)
 - ▶ warstwy druga i pierwsza (łącza danych i fizyczna)
- ▶ W MODBUS wykorzystywana jest reguła wymiany danych typu master-slave (nadrzędny-podrzędny)
- ▶ MODBUS wykorzystywany jest do znakowej wymiany informacji pomiędzy urządzeniami systemów automatyki przemysłowej

Co zdecydowało o popularności MODBUS w zastosowaniach przemysłowych

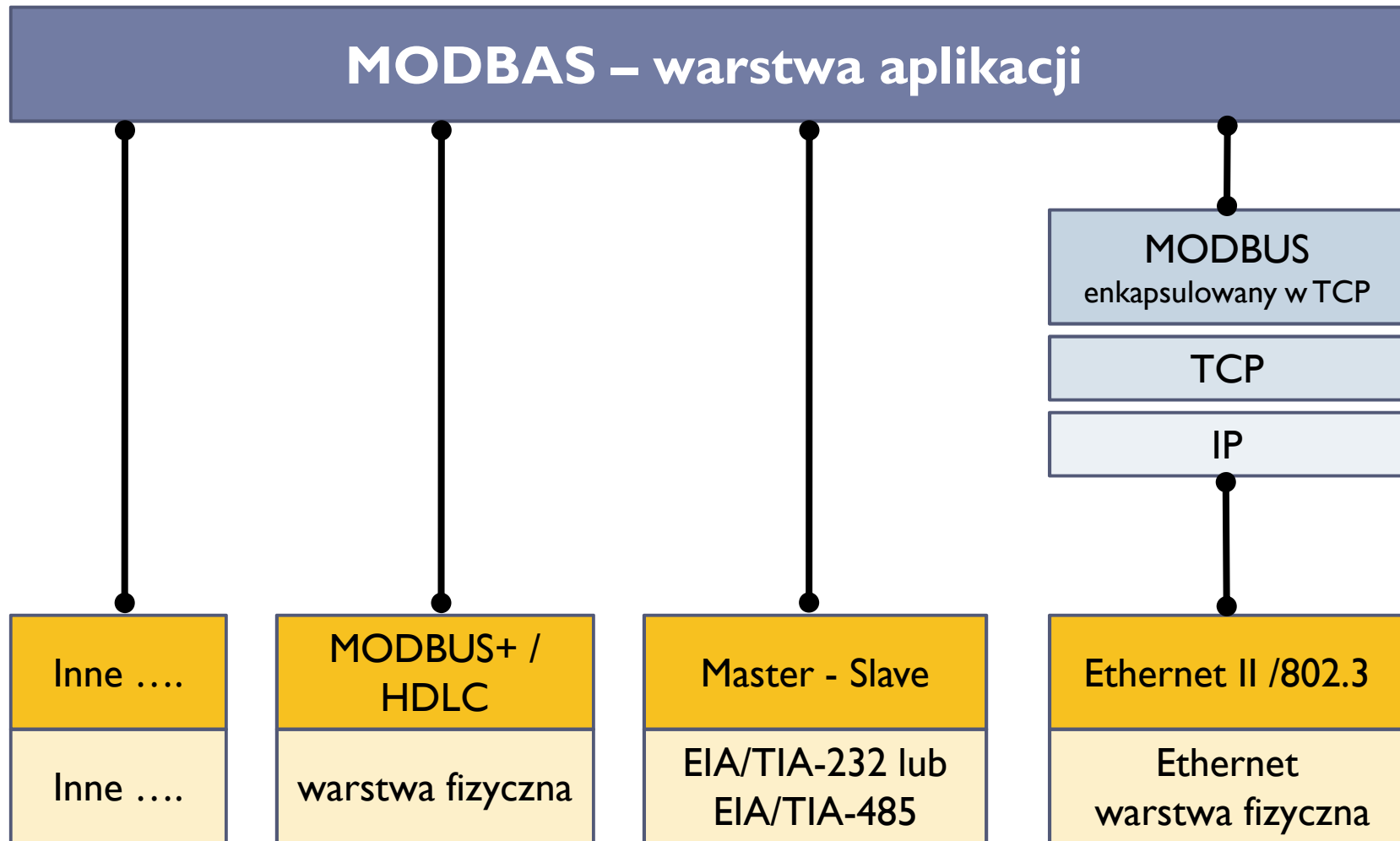
- ▶ Prosta reguła dostępu do łącza oparta na zasadzie master-slave
- ▶ Zabezpieczenie przesyłanych komunikatów przed błędami:
 - ▶ dla trybu ASCII (znakowego) - Longitudinal Redudancy Check, LRC
 - ▶ dla trybu RTU (binarnego) - Cyclic Redudancy Check, CRC
- ▶ Potwierdzenie wykonania rozkazów zdalnych i sygnalizacja błędów
- ▶ Skuteczne mechanizmy zabezpieczające przed zawieszeniem systemu
- ▶ Wykorzystanie asynchronicznej transmisji znakowej zgodnej z RS232C (ale czy tylko?)

MODBUS

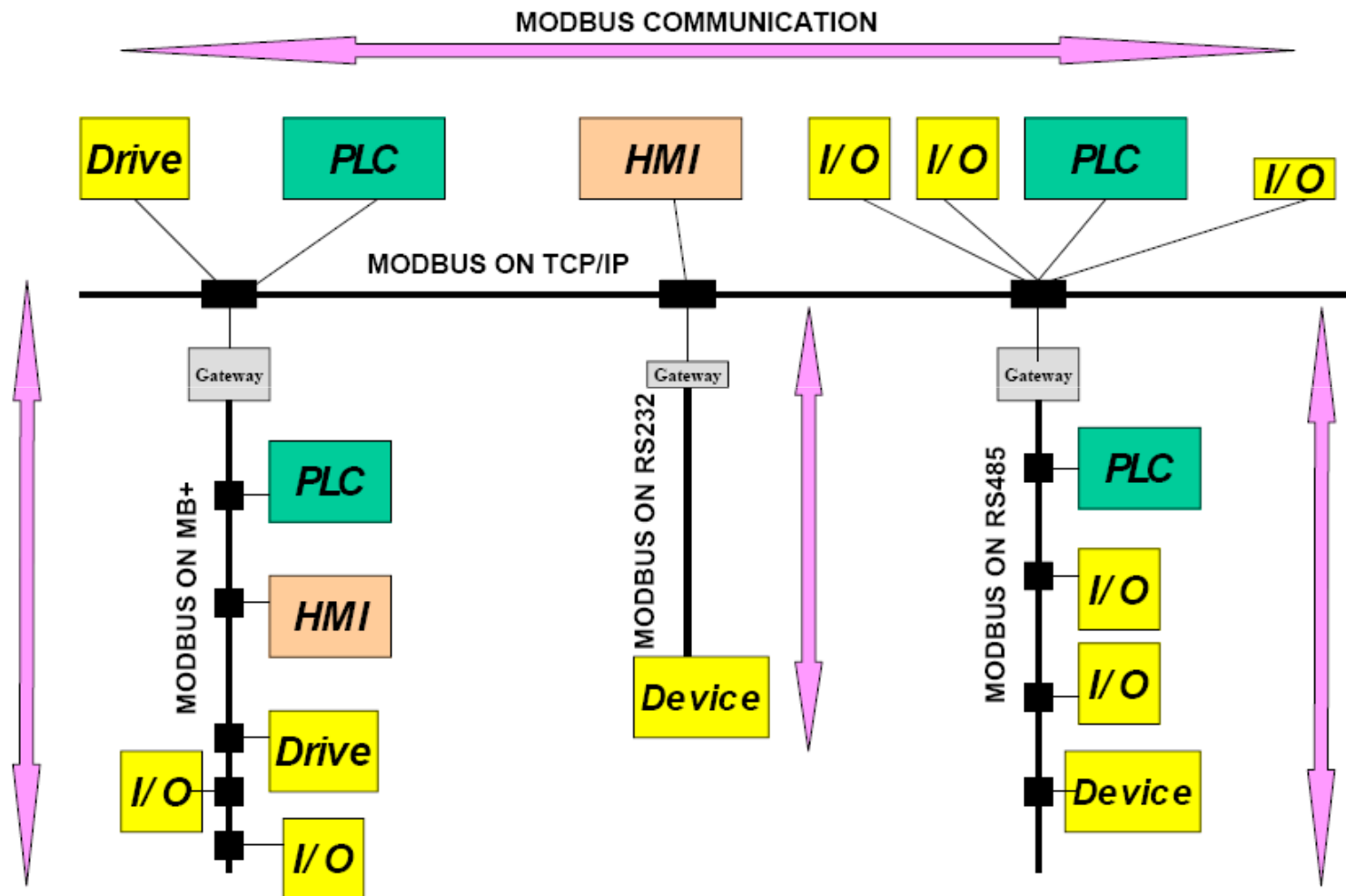
- aktualne możliwości realizacji

- ▶ **Asynchroniczna, znakowa transmisja szeregową:**
 - ▶ RS232
 - ▶ lub RS485
- ▶ **TCP/IP poprzez Ethernet (enkapsulowany MODBUS)**
 - ▶ na zarezerwowanym 502 porcie
- ▶ **MODBUS PLUS**
 - ▶ szybka sieć związana z przekazywaniem znacznika
 - ▶ z efektywnym protokołem zorientowanym bitowo HDLC (*ang. High-level Data Link Control*)

MODBUS - stos komunikacyjny



MODBUS - stos komunikacyjny

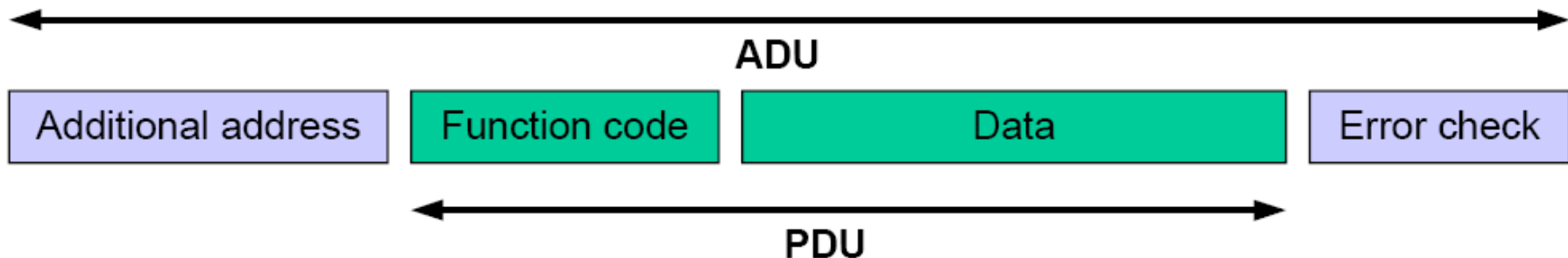


MODBUS – warstwa aplikacji

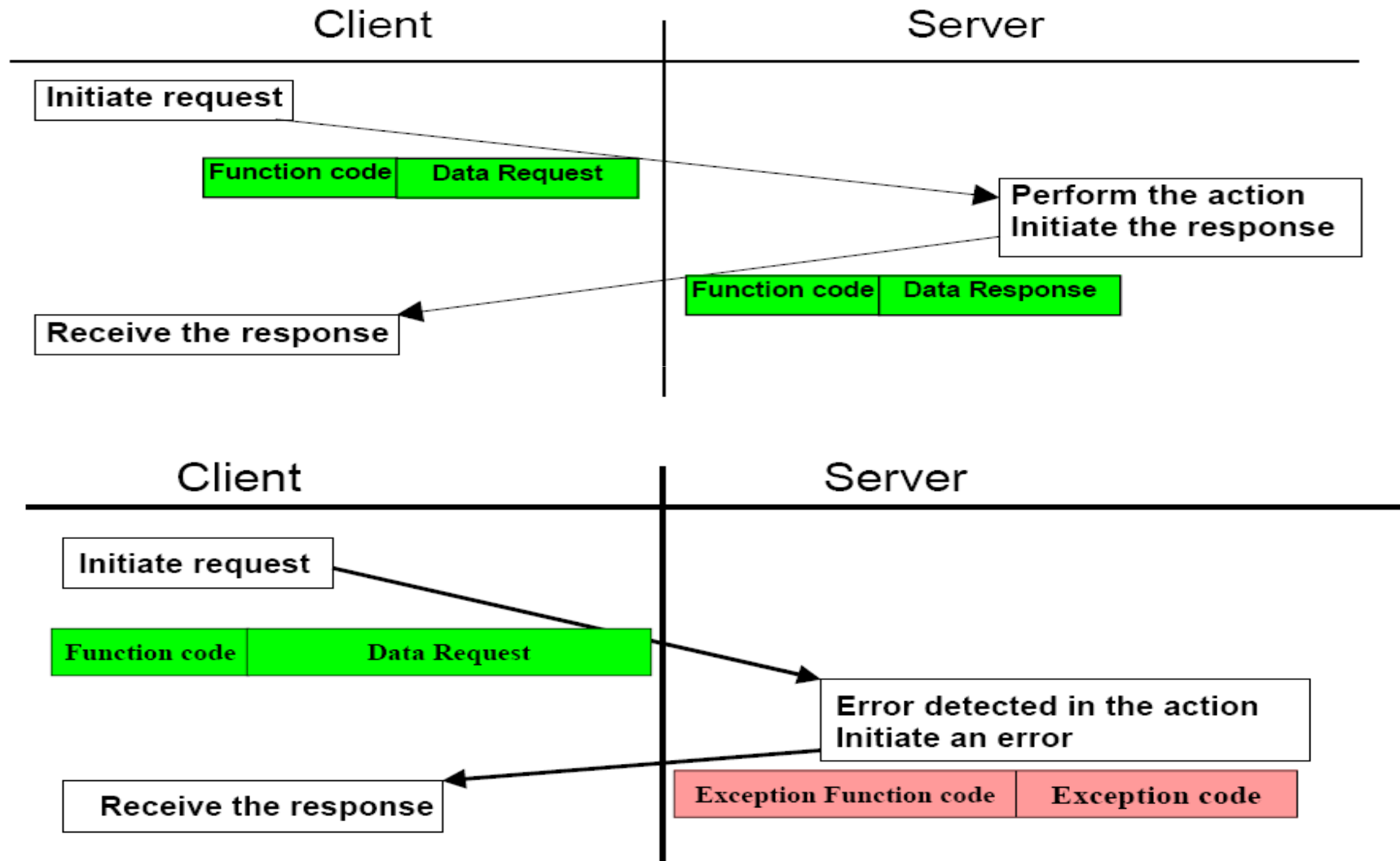
- ▶ Protokół przekazujący informacje w warstwie aplikacji jest protokołem typu client-server
- ▶ Jest protokołem typu query–response (zapytanie-odpowiedź)
- ▶ Definiuje trzy proste formaty zapytań i odpowiedzi PDU (*ang. Protocol Data Unit*):
 - ▶ MODBUS Request PDU, **mb_req_pdu**
 - ▶ MODBUS Response PDU, **mb_rsp_pdu**
 - ▶ MODBUS Exception Response PDU, **mb_excep_rsp_pdu**

MODBUS – warstwa aplikacji

- ▶ Formaty zapytań i odpowiedzi są niezależne od „niższych” warstw komunikacyjnych modelu
- ▶ W zależności od implementacji MODBUS, PDU się „rozrasta”, dodawane są kolejne pola ramki – ADU (*ang. Application Data Unit*)
- ▶ Długość komunikatu ADU:
 - ▶ dla transmisji RS232/RS485 : 256 bajtów
 - ▶ dla transmisji MODBUS TCP : 260 bajtów



MODBUS – warstwa aplikacji - transakcje



MODBUS – warstwa aplikacji

- transakcje – standardowe kody odpowiedzi wyjątkowej

Kod	Opis
01	Niedozwolona funkcja
02	Niedozwolony zakres (adres) danych
03	Niedozwolona wartość danej
04	Błąd urządzenia Slave
05	Potwierdzenie pozytywne
06	Brak gotowości urządzenia Slave
07	Potwierdzenie negatywne
08	Błąd parzystości pamięci

Realizacja MODBUS z wykorzystaniem transmisji szeregowej

Najważniejsze cechy protokołu MODBUS

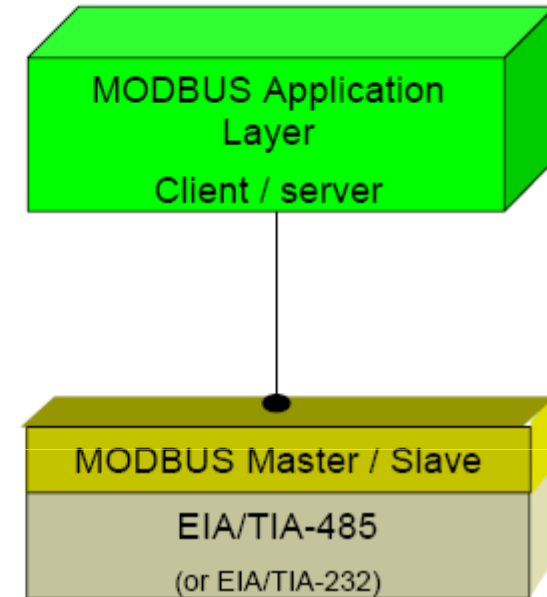
- ▶ Zasada dostępu do łącza „Query – Response” („Master-Slave”) gwarantuje bezkonfliktowe współdzielenie magistrali przez wiele węzłów
- ▶ Węzeł nadrzędny (Master) steruje pracą sieci
- ▶ Węzły podrzędne (Slaves) nie podejmują samodzielnie transmisji, odpowiadają na zdalne polecenia od węzła nadrzędnego
- ▶ Każdy z węzłów podrzędnych posiada przypisany unikalny adres z zakresu 1-247 (adres broadcast = 0)
- ▶ Węzeł nadrzędny nie posiada adresu

Najważniejsze cechy protokołu MODBUS, cd.

- ▶ Dwa różne tryby transmisji ASCII (znakowy) lub RTU (binarny)
- ▶ Komunikaty zawierające polecenia i odpowiedzi mają identyczną strukturę
- ▶ Maksymalna długość komunikatów wynosi 256 bajtów
- ▶ Znaki są przesyłane szeregowo od najmłodszego do najstarszego bitu

MODBUS - implementacja szeregową - a model warstwowy ISO/OSI

Layer	ISO/OSI Model	
7	Application	MODBUS Application Protocol
6	Presentation	Empty
5	Session	Empty
4	Transport	Empty
3	Network	Empty
2	Data Link	MODBUS Serial Line Protocol
1	Physical	EIA/TIA-485 (or EIA/TIA-232)



- ▶ Warstwa fizyczna oparta jest o specyfikacje RS232 lub RS485
- ▶ Warstwa łącza danych wykorzystuje:
 - ▶ protokół typu master-slave, typ transmisji ASCII lub RTU
- ▶ Warstwa aplikacji wykorzystuje:
 - ▶ protokół komunikacyjny typu client-server:
(client to master, server to slave)

MODBUS – warstwa fizyczna

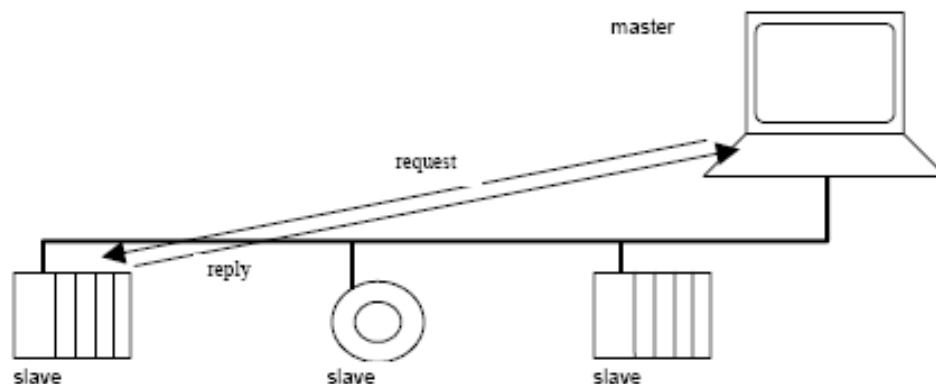
- ▶ Warstwa fizyczna oparta o specyfikacje:
 - ▶ RS232
 - ▶ lub RS485

!!! *Informacje z poprzednich wykładów* !!!

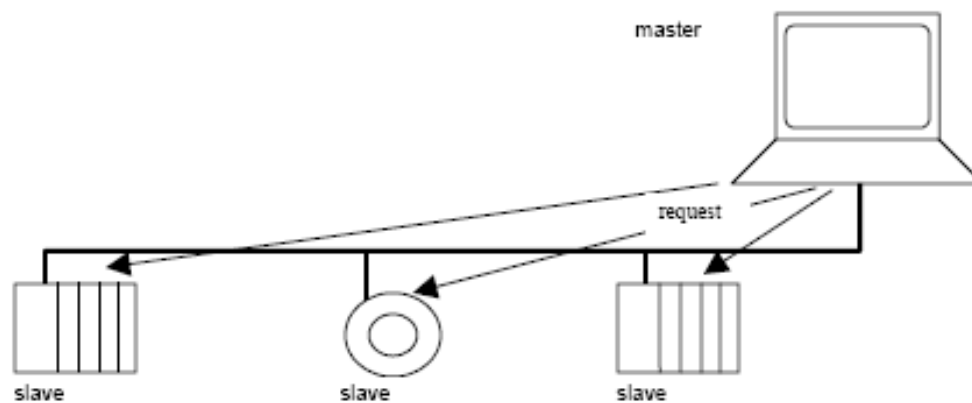
MODBUS – warstwa łączy danych - transakcje

- ▶ Tylko Master inicjalizuje transakcje
- ▶ Pozostałe jednostki Slave odpowiadają na zdalne zapytania Mastera
- ▶ Transakcja składa się z:
 - ▶ Polecenia (Query) wysłanego z Mastera do Slavea
 - ▶ Odpowiedzi (Response) przesyłanej z Slavea do Mastera
- ▶ Odpowiedzi (Response) zawiera dane żądane przez Mastera lub potwierdzenie realizacji jego polecenia
- ▶ Jeżeli Slave wykryje błąd przy obiorze wiadomości, lub jeżeli nie jest w stanie wykonać polecenia, wysyła odpowiedni komunikat do Mastera (odpowieź szczególna/wyjatkowa, ang. *Exception Response*)

MODBUS – warstwa łącza danych - transakcje



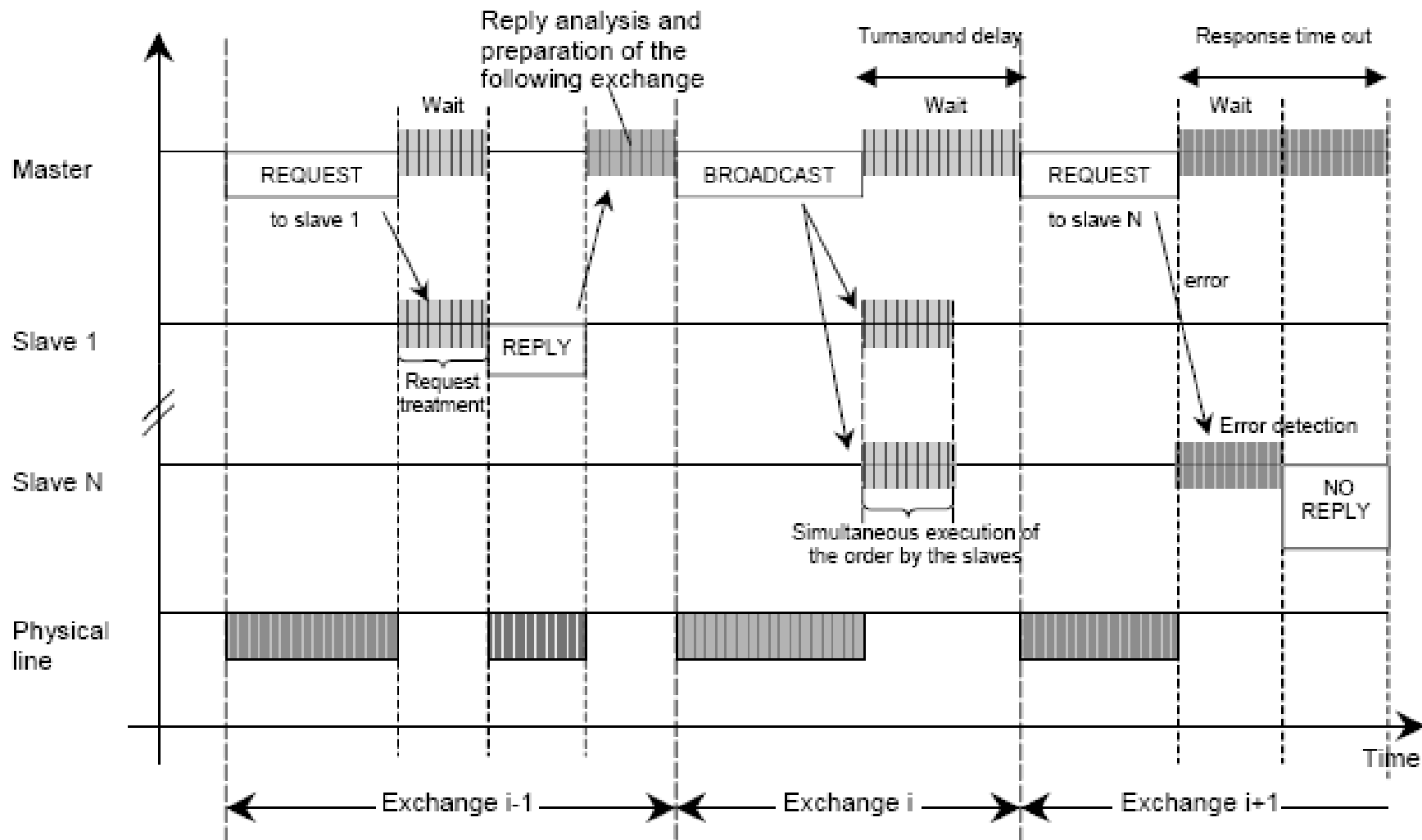
Unicast mode



Broadcast mode
(rozszewcze)

MODBUS – warstwa łącza danych

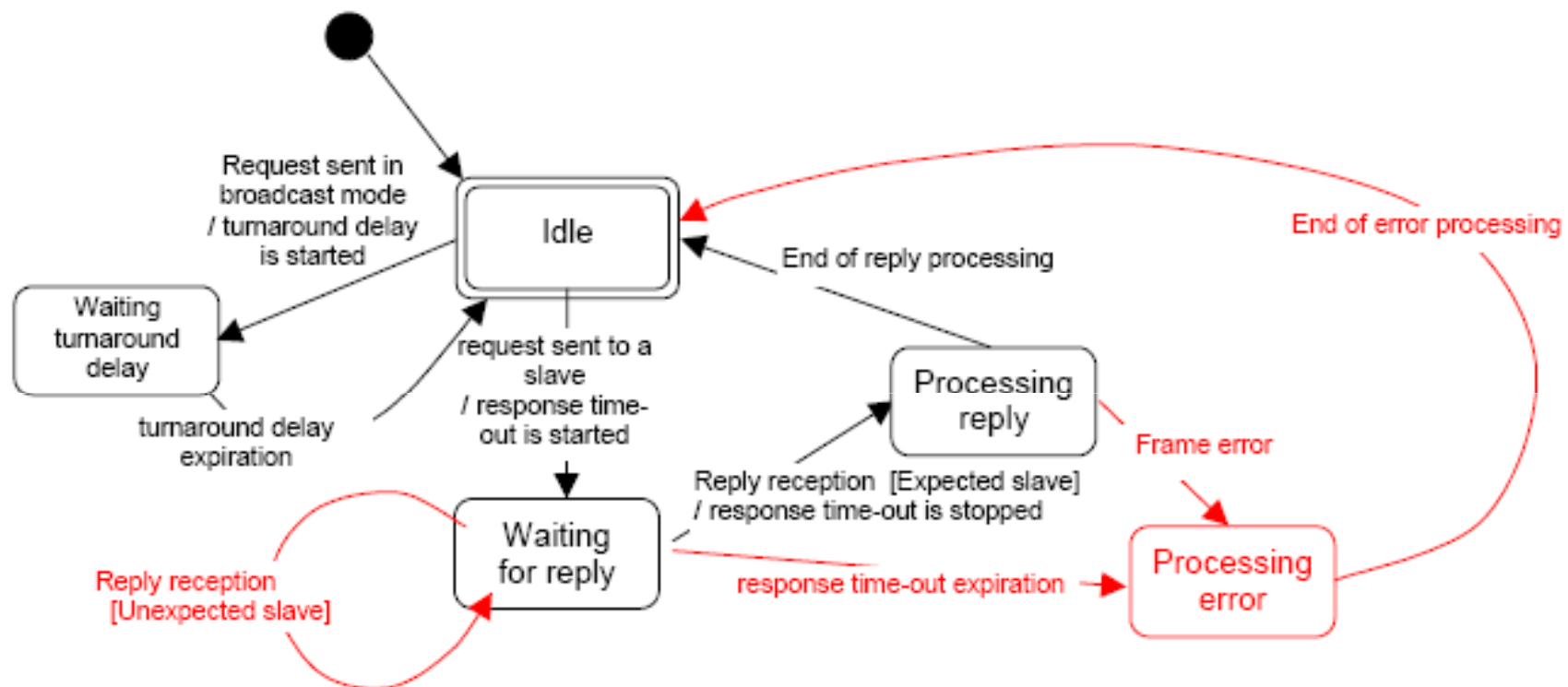
- transakcje



MODBUS – warstwa łącza danych - transakcje

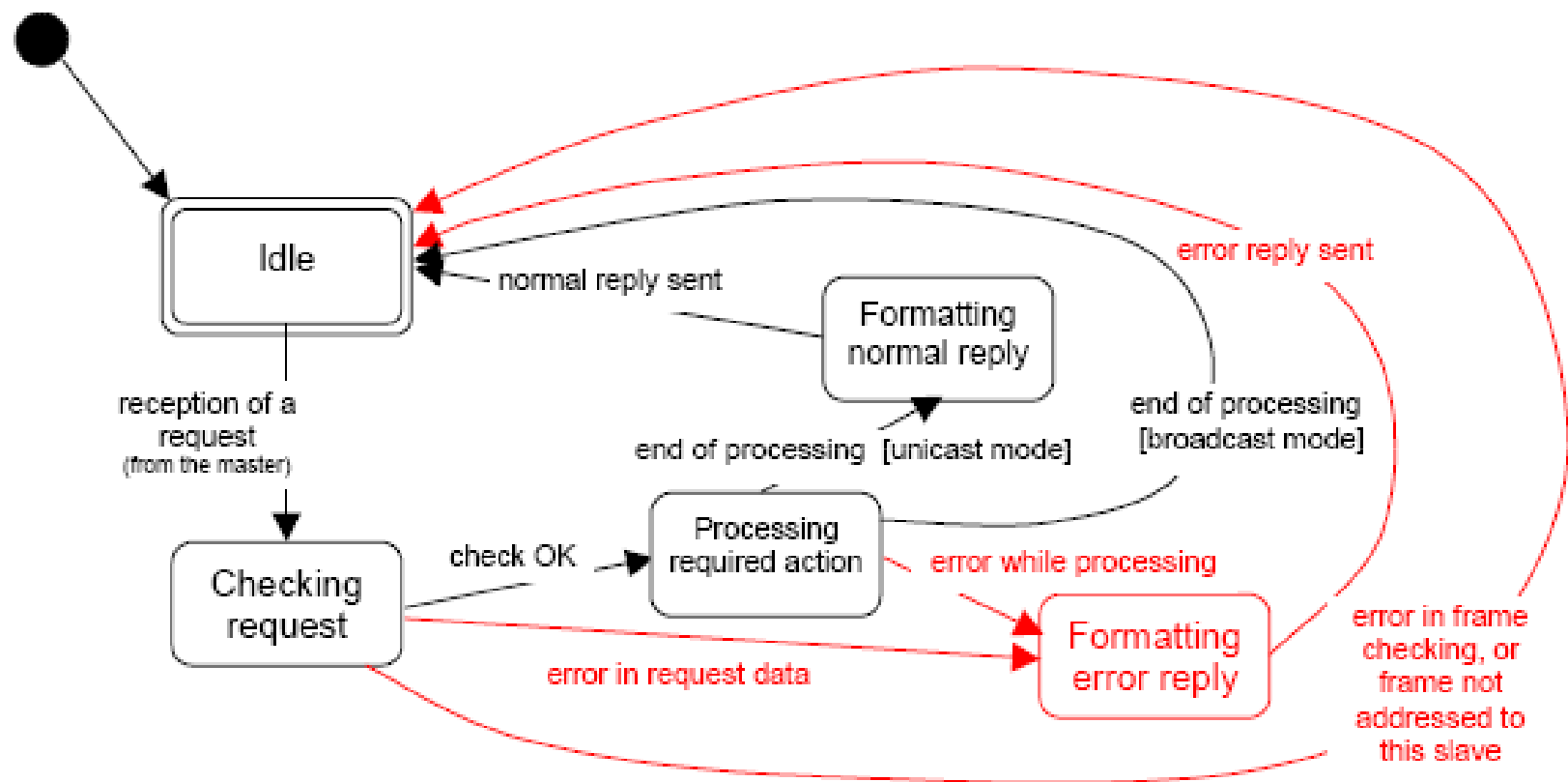
- ▶ W jednostce master użytkownik ustawia parametr „maksymalnego czasu odpowiedzi na ramkę zapytania”
- ▶ „Maksymalny czas odpowiedzi” dobiera się tak by najwolniejszy slave pracujący w systemie zdążył odpowiedzieć na ramkę zapytania
- ▶ Przekroczenie „maksymalnego czasu odpowiedzi” jest traktowane przez mastera jako błąd
- ▶ Program mastera jest odpowiedzialny za obsługę błędów (np. powtórzenie ramki polecenia czy powiadomienie operatora)
- ▶ Jeżeli slave wykryje błąd w ramce nie odpowiada na nią, powoduje to przekroczenie „maksymalnego czasu odpowiedzi” i przerwanie transakcji

MODBUS – warstwa łącza danych - transakcje



Graf działania Mastera

MODBUS – warstwa łącza danych - transakcje



Graf działania Slavea

MODBUS – warstwa łącza danych - ramka ASCII – „historyczna”

Znacznik początku	Adres	Funkcja	Dane	Kontrola LRC	Znacznik końca
1 ZNAK :	2 ZNAKI	2 ZNAKI	n ZNAKÓW od 0 do 2x252	2 ZNAKI	2 ZNAKI CR CL

- ▶ System kodowania, heksadecymalny, znaki ASCII 0-9,A-F
- ▶ Każdy znak heksadecymalny odpowiada 4 bitom
- ▶ Znaki przesyłane asynchronicznie, z lub bez kontroli parzystości (jednostka informacyjna 10 bitów z bitami startu i stopu)

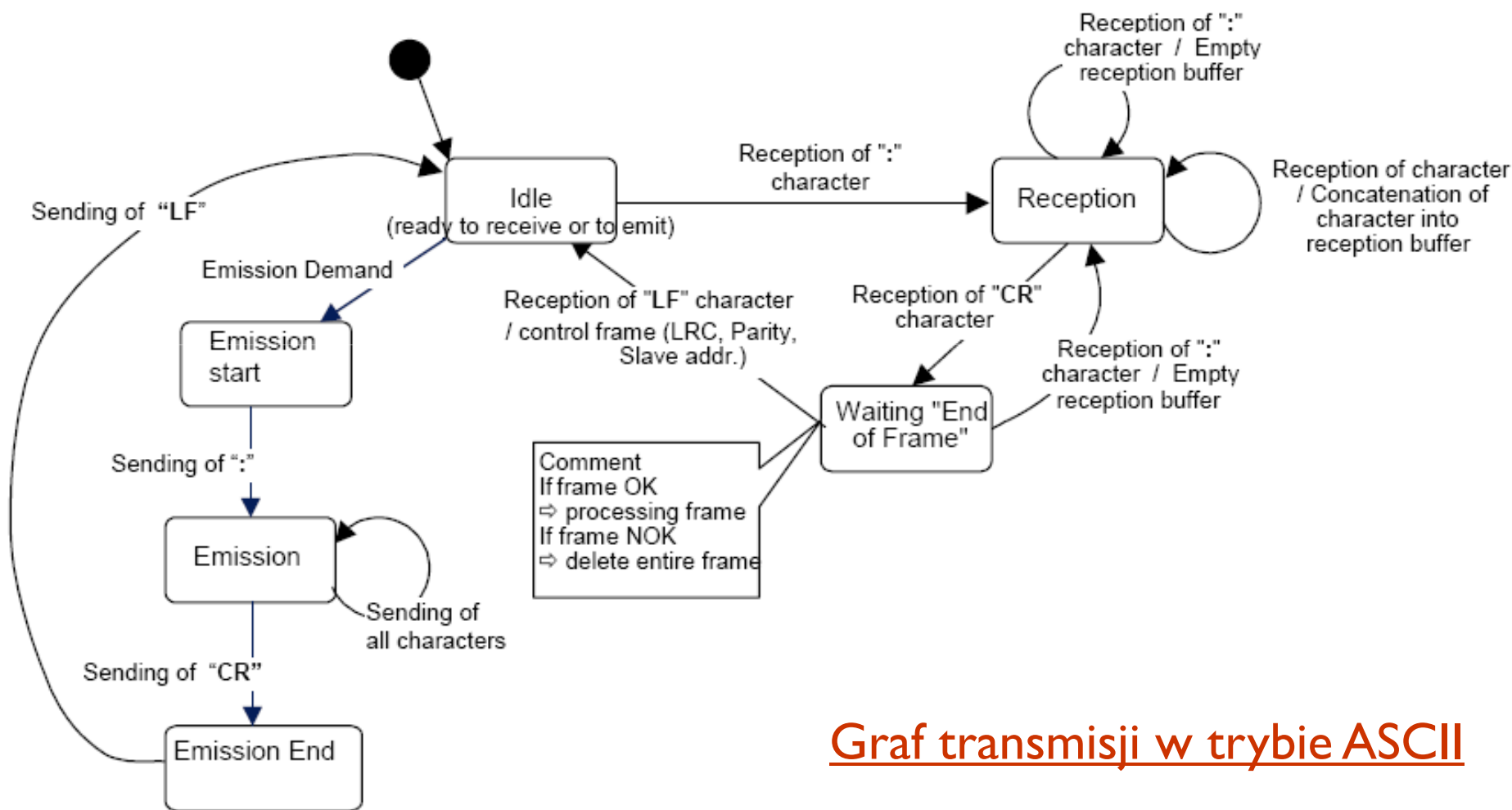
START	1 BIT	2 BIT	3 BIT	4 BIT	5 BIT	6 BIT	7 BIT	PAR	STOP
START	1 BIT	2 BIT	3 BIT	4 BIT	5 BIT	6 BIT	7 BIT	STOP	STOP

MODBUS – warstwa łącza danych - ramka ASCII – „historyczna”

- ▶ Dopuszczalne odstępy pomiędzy znakami: do 1 sekundy
- ▶ Długość ramki: do 513 znaków (pole danych 2x252)
- ▶ Ramki „niedokończone” muszą być wykrywane i odesłane

- ▶ Ramka zawiera pole kodu LRC, zabezpieczające część informacyjną ramki (bez znaku początku i końca)
- ▶ Znaki LRC dołączane są na końcu ramki, przed znacznikiem końca ramki (CR, LF)
- ▶ Mechanizm obliczania LRC:
 - ▶ należy obliczyć 8-bitową sumę (bez przeniesień) bajtów ramki
 - ▶ wynik LRC to dopełnienie do 2 obliczonej sumy

MODBUS – warstwa łącza danych - ramka ASCII – „historyczna”



Graf transmisji w trybie ASCII

MODBUS – warstwa łącza danych - ramka RTU

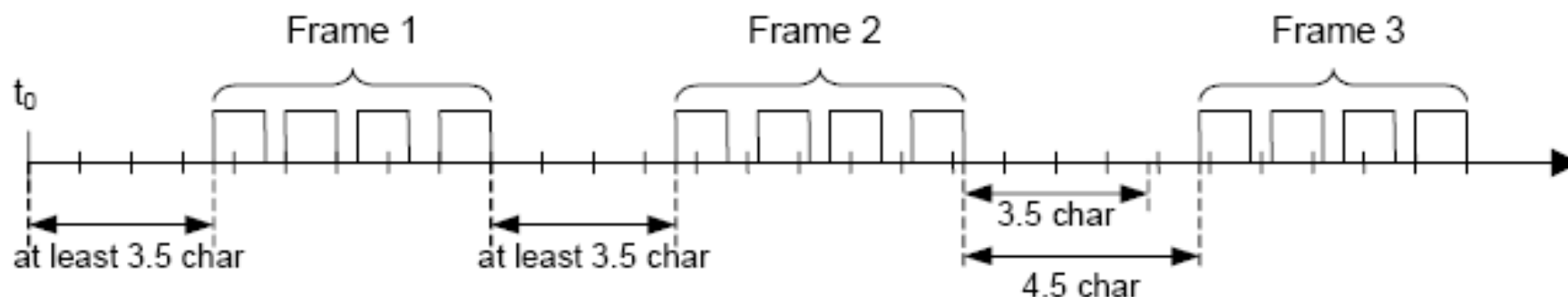
Początek ramki	Adres	Funkcja	Dane	Kontrola CRC	Koniec ramki
Cisza na łączu ≥ 3,5 znaku	8 bitów	8 bitów	n x 8 BITÓW od 0 do 252 bajtów	CRC Lo CRC HI 16 bitów	Cisza na łączu ≥ 3,5 znaku

- ▶ 8-bitowe, binarne kodowanie danych
- ▶ Znaki przesyłane asynchronicznie, z lub bez kontroli parzystości (jednostka informacyjna 11 bitów z bitami startu i stopu)

START	1 BIT	2 BIT	3 BIT	4 BIT	5 BIT	6 BIT	7 BIT	8 BIT	PAR	STOP
START	1 BIT	2 BIT	3 BIT	4 BIT	5 BIT	6 BIT	7 BIT	8 BIT	STOP	STOP

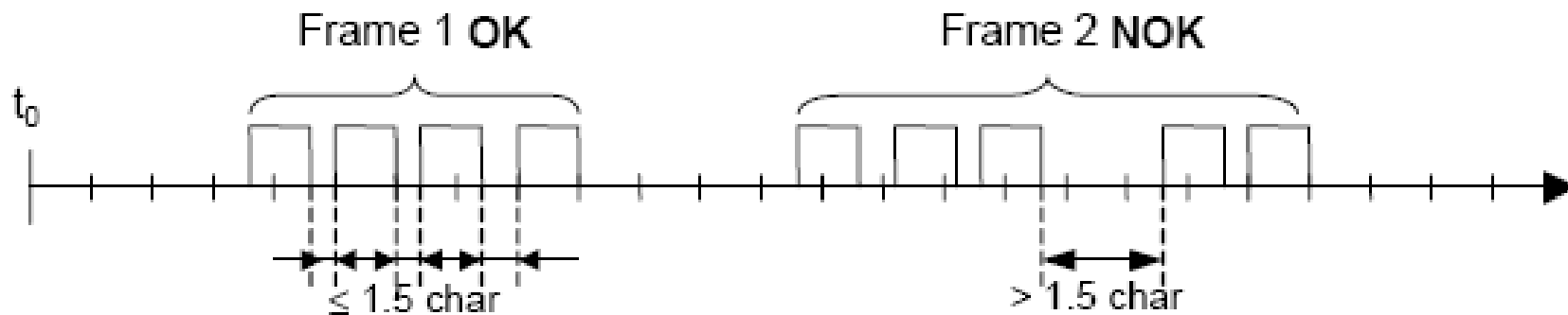
MODBUS – warstwa łącza danych - ramka RTU

- ▶ Wiadomość rozpoczyna się odstępem czasowym trwającym $3,5 \times$ czas trwania pojedynczego znaku (cisza na łączu)
- ▶ Wiadomość kończy się odstępem czasowym trwającym $3,5 \times$ czas trwania pojedynczego znaku, po tym odstępie można wysłać następną wiadomość



MODBUS – warstwa łącza danych - ramka RTU

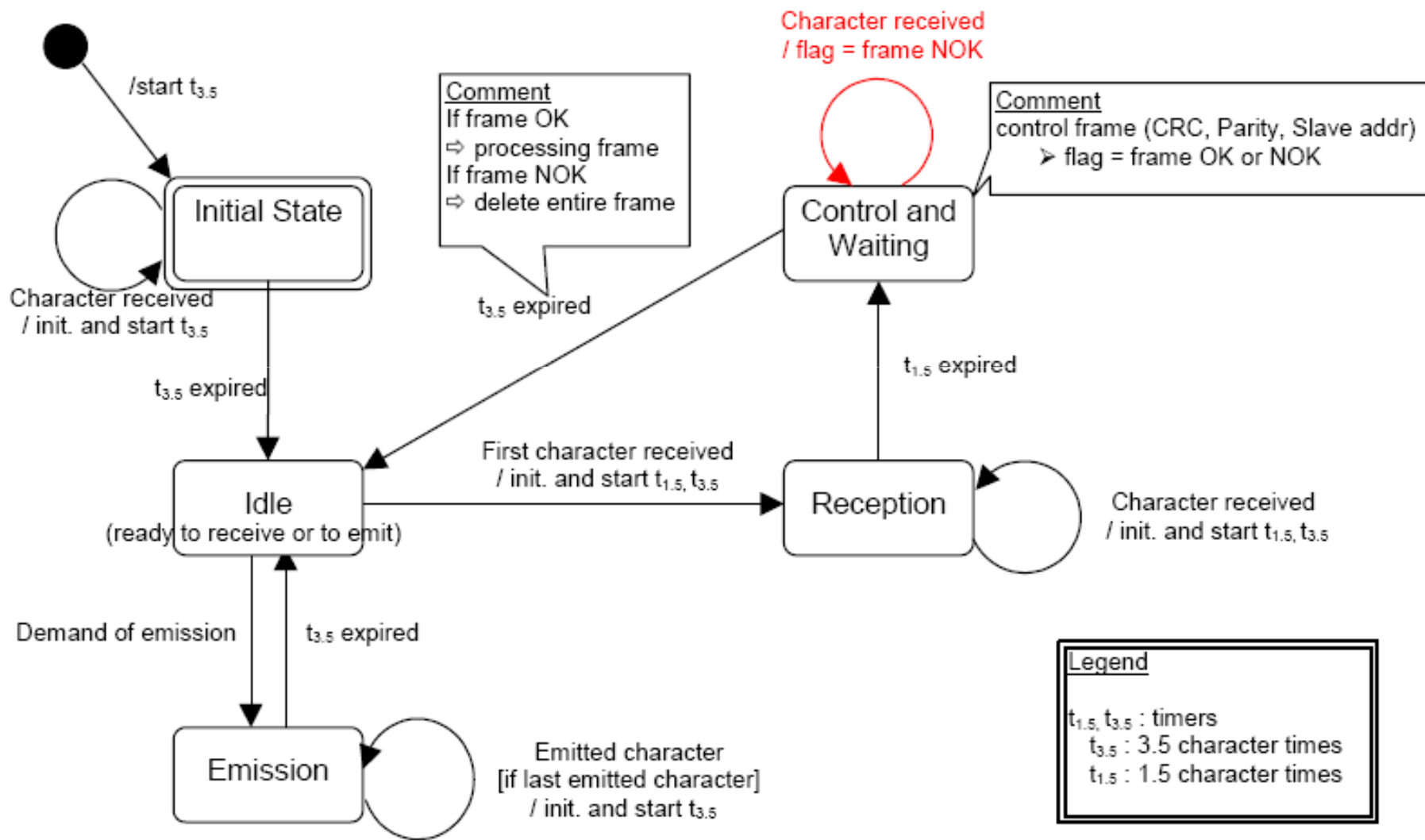
- ▶ Ramka musi być transmitowana w sposób ciągły – odstęp pomiędzy poszczególnymi znakami tworzącymi ramkę musi być mniejszy od $1,5 \times \text{długość znaku}$



MODBUS – warstwa łącza danych

- ramka RTU

Graf transmisji w trybie RTU



MODBUS – warstwa łącza danych - ramka RTU

- ▶ Sprawdzanie spójności ramki poprzez stosowania słowa kontrolnego CRC16
- ▶ Obliczane jest CRC zawartości ramki
- ▶ Mechanizm obliczania CRC16:
 - ▶ bezpośrednio
 - ▶ za pomocą tablic
- ▶ Pole kontrolne zajmuje dwa bajty dołączane na końcu ramki
- ▶ Pierwszy bajt jest mniej znaczącym bajtem CRC Lo
- ▶ Drugi bajt jest starszym bajtem CRC Hi

MODBUS – warstwa łącza danych

- ramka RTU – bezpośrednie generowanie CRC16

Postać wielomianu generującego $x^{16}+x^{15}+x^2+1$

- w postaci hex to: A001h
- w postaci binarnej: 1010 0000 0000 0001

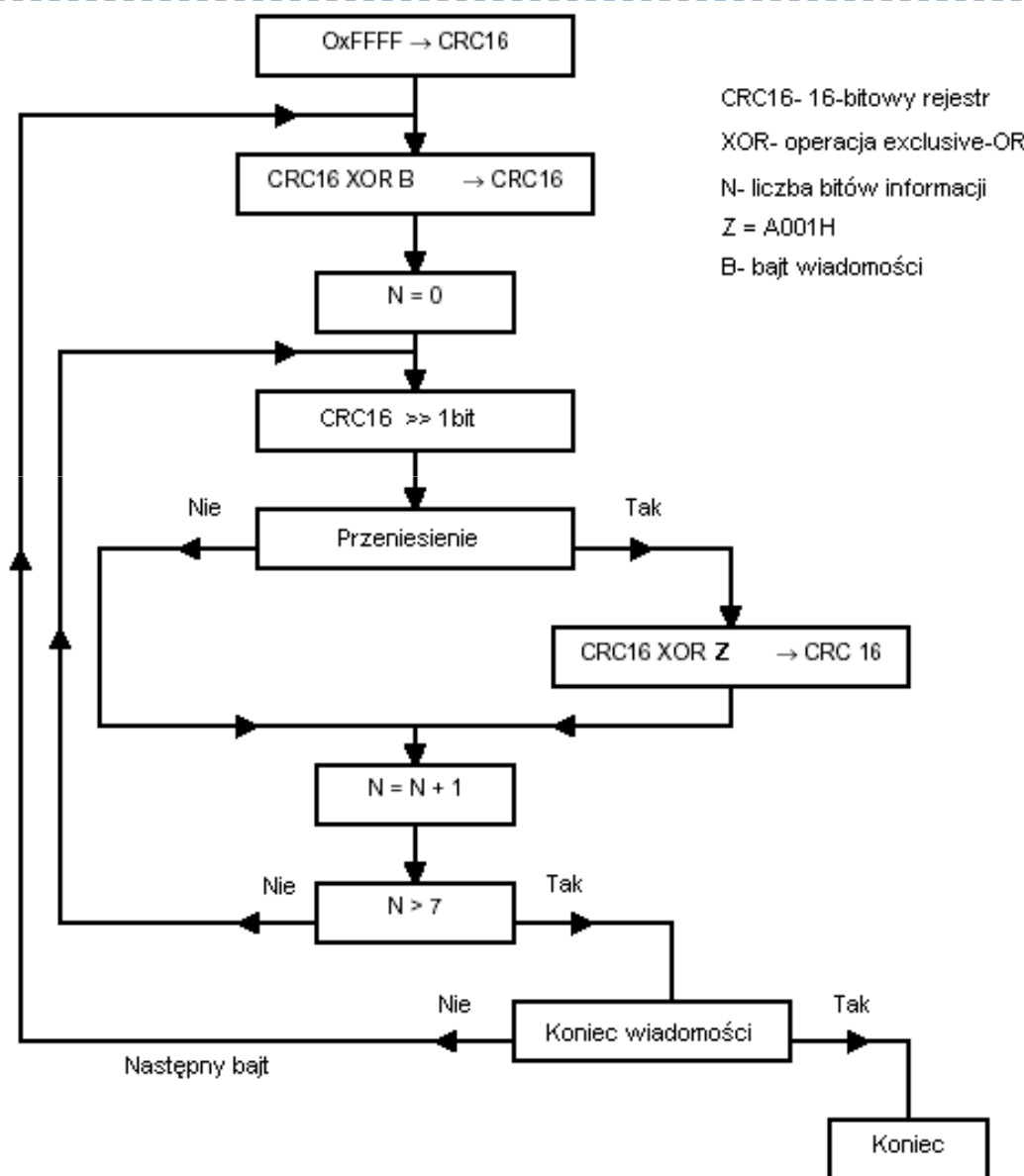
MODBUS – warstwa łącza danych

- ramka RTU – bezpośrednie generowanie CRC16

- 1) Załadowanie wartości 0xFFFF do 16-bitowego rejestru CRC.
 - 2) Pobranie jednego bajtu z bloku danych zabezpieczonej wiadomości i wykonanie operacji logicznej XOR z młodszym bajtem rejestru. Umieszczenie rezultatu w rejestrze CRC.
 - 3) Przesunięcie zawartości rejestru w prawo o jeden bit połączone z wpisaniem 0 na najbardziej znaczący bit.
 - 4) Sprawdzenie stanu najmłodszego bitu w rejestrze CRC. W przypadku, gdy jego wartość równa się 0, to następuje powrót do kroku 3, jeżeli 1, to wykonywana jest operacja XOR rejestru CRC ze stałą A001h.
 - 5) Powtórzenie kroków 3 i 4 osiem razy, co odpowiada przetworzeniu całego bajtu.
 - 6) Powtórzenie sekwencji 2, 3, 4, 5 dla kolejnego bajtu wiadomości.
 - 7) Zawartość CRC po wykonaniu wymienionych operacji jest poszukiwaną wartością CRC.
-

MODBUS – warstwa łącza danych

- ramka RTU – bezpośrednio generowanie CRC16



MODBUS – warstwa łączy danych - pole adresu

- ▶ Pole adresowe w ramce zawiera:
 - ▶ dwa znaki w trybie ASCII
 - ▶ osiem bitów w trybie RTU
- ▶ Zakres adresów jednostek slave wynosi od 0 do 247
- ▶ Adres 0 jest adresem rozgłoszeniowym, rozpoznawanym przez wszystkie jednostki slave pracujące w systemie
- ▶ Master adresuje slava umieszczając jego adres w polu adresowym ramki
- ▶ Gdy slave odsyła odpowiedź, umieszcza swój adres w polu adresowym ramki, co umożliwia sprawdzenie masterowi, z którym slavem realizowana jest transakcja

MODBUS – warstwa łącza danych - pole funkcji

- ▶ Pole funkcji w ramce zawiera:
 - ▶ dwa znaki w trybie ASCII
 - ▶ osiem bitów w trybie RTU
- ▶ Zakres kodów operacji przyjmuje wartości od 1 do 255
- ▶ Przy transmisji Master->Slave, pole funkcji zawiera kod rozkazu określający jakie działanie ma podjąć Slave na żądanie Mastera
- ▶ W przypadku odpowiedzi Slave->Master, pole funkcji wykorzystane jest do potwierdzenia wykonania polecenia lub sygnalizacji błędu
- ▶ W przypadku błędu Slave w polu funkcji umieszcza szczególną odpowiedź (ang. exception response) która stanowi kod funkcji z ustawionym najstarszym bitem na 1
- ▶ Dodatkowo Slave w polu danych ramki umieszcza kod błędu, co umożliwia Masterowi określić rodzaj lub powód błędu

MODBUS – warstwa łącza danych

- pole kodu funkcji - przykłady

<u>kod</u>	<u>kod (hex)</u>	<u>opis</u>
1	01h	odczyt wyjść bitowych
2	02h	odczyt wejść bitowych
3	03h	odczyt n rejestrów
4	04h	odczyt n rejestrów wejściowych
5	05h	zapis 1 bitu
6	06h	zapis 1 rejestru
7	07h	odczyt statusu urządzenia slave
8	08h	test diagnostyczny
15	0Fh	zapis n bitów
16	10h	zapis n rejestrów
17	11h	identyfikacja urządzenia slave
128-255	80h-FFh	zarezerwowane dla odpowiedzi błędnych

MODBUS – warstwa łączy danych - pole danych

- ▶ Pole danych ramki tworzy zestaw dwucyfrowych liczb heksadecymalnych, o zakresie od 00 do FF
 - ▶ w trybie ASCII reprezentowane dwoma znakami
 - ▶ w trybie RTU reprezentowane jednym znakiem
 - ▶ Pole danych ramki zawiera dodatkowe informacje (adresy rejestrów, liczba bajtów w polu danych, dane ...) potrzebne jednostce slave do wykonania rozkazu określonego kodem funkcji
 - ▶ np.: kiedy master żąda odczytu grupy rejestrów (kod funkcji 03h), to pole danych zawiera: adres rejestru początkowego oraz ilość rejestrów do odczytu
 - ▶ np.: kiedy master żąda zapisu grupy rejestrów (kod funkcji 10h), to pole danych zawiera: adres rejestru początkowego, ilość rejestrów, ilość pozostałych bajtów w polu danych oraz dane do zapisu
 - ▶ Niekiedy pole danych może mieć długość równą zero (operacja określona odpowiednim kodem funkcji nie wymaga żadnych parametrów)
-

MODBUS – „przechowywanie” danych

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

- ▶ Standard MODBUS definiuje cztery powyższe tablice
- ▶ W komunikatach MODBUS wykorzystuje się odpowiednie adresy a nie nazwy (etykiety np. w PLC)
- ▶ Należy offset, np. pierwszy rejestr związany z wyjściem analogowym będzie miał numer 40001 i adres 0000

MODBUS – przykład 1 – odczyt zmiennej dyskretnej - ZAPYTANIE

Request

This command is requesting the ON/OFF status of discrete coils # 20 to 56 from the slave device with address 17.

11 01 0013 0025 0E84

11: The Slave Address (17 = 11 hex)

01: The Function Code (read Coil Status)

0013: The Data Address of the first coil to read. (Coil 20 - 1 = 19 = 13 hex)

0025: The total number of coils requested. (coils 20 to 56 = 37 = 25 hex)

0E84: The CRC (cyclic redundancy check) for error checking.

MODBUS – przykład 1 – odczyt zmiennej dyskretnej - ODPOWIEDŹ

Response

11 01 05 CD6BB20E1B 45E6

11: The Slave Address (17 = 11 hex)

01: The Function Code (read Coil Status)

05: The number of data bytes to follow (37 Coils / 8 bits per byte = 5 bytes)

CD: Coils 27 - 20 (1100 1101)

6B: Coils 35 - 28 (0110 1011)

B2: Coils 43 - 36 (1011 0010)

0E: Coils 51 - 44 (0000 1110)

1B: 3 space holders & Coils 56 - 52 (0001 1011)

45E6: The CRC (cyclic redundancy check).

MODBUS – przykład 2 – zapis n rejestrów - ZAPYTANIE

Request

This command is writing the contents of two analog output holding registers # 40002 & 40003 to the slave device with address 17.

```
11 10 0001 0002 04 000A 0102 C6F0
```

11: The Slave Address (17 = 11 hex)

10: The Function Code (Preset Multiple Registers 16 = 10 hex)

0001: The Data Address of the first register. (# 40002 - 40001 = 1)

0002: The number of registers to write

04: The number of data bytes to follow (2 registers x 2 bytes each = 4 bytes)

000A: The value to write to register 40002

0102: The value to write to register 40003

C6F0: The CRC (cyclic redundancy check) for error checking.

MODBUS – przykład 2 – zapis n rejestrów - ODPOWIEDŹ

Response

11 10 0001 0002 1298

11: The Slave Address (17 = 11 hex)

10: The Function Code (Preset Multiple Registers 16 = 10 hex)

0001: The Data Address of the first register. (# 40002 - 40001 = 1)

0002: The number of registers written.

1298: The CRC (cyclic redundancy check) for error checking.

Realizacja MODBUS z wykorzystaniem Ethernetu (*MODBUS RTU z interfejsem TCP czyli MODBUS TCP/IP*)

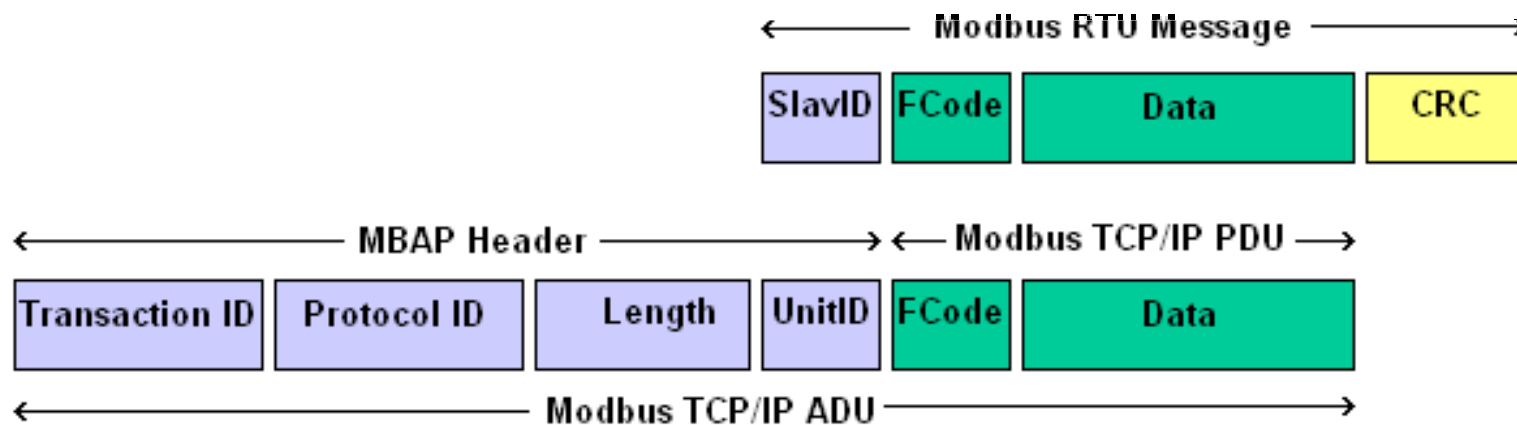
MODBUS TCP/IP

- ▶ MODBUS TCP/IP to protokół MODBUS RTU z interfejsem TCP/IP
- ▶ MODBUS TCP/IP wykorzystuje TCP/IP oraz sieć Ethernet do transportu danych o strukturze MODBUS
- ▶ Dla aplikacji MODBUS TCP wykorzystuje 502 port systemowy

Enkapsulacja ramki MODBUS RTU w ramce TCP (model TCP/IP)

MODBUS TCP/IP - ramka

- Ze standardowej ramki MODBUS RTU usuwa się pole adresu (identyfikator slava) oraz pole sumy kontrolnej CRC, wykorzystuje się dalej pole kodu funkcji i danych
- Ramka MODBUS transmitowana przez TCP/IP rozrasta się o 7 bajtowy nagłówek MBAP (*ang. Modbus Application Header*)



PDU - *Protocol Data Unit*
ADU - *Application Data Unit*

MODBUS TCP/IP – ramka – nagłówek MBAP

- ▶ Identyfikator transakcji (Transaktion ID – 2 bajty)
ustawiane przez klienta w celu identyfikacji kolejnego zapytania w ramach jednego połączenia TCP
- ▶ Identyfikator protokołu (Protocil ID – 2 bajty)
obecnie niewykorzystywane i ustawiane na zero: 00 00
- ▶ Długość pola (Lenght – 2 bajty)
wskazuje liczbę bajtów wiadomości
- ▶ Identyfikator jednostki (Unit ID – 1 bajt)
ustawiane przez klienta i powtarzane przez serwer identyfikuje jednoznacznie „podłączonego” klienta

MODBUS TCP/IP – ramka – przykład - krok 1

Here is an example of a Modbus RTU request for the content of analog output holding registers # 40108 to 40110 from the slave device with address 17.

11 03 006B 0003 7687

11: The SlaveID Address (17 = 11 hex)

03: The Function Code (read Analog Output Holding Registers)

006B: The Data Address of the first register requested. (40108-40001 = 107 = 6B hex)

0003: The total number of registers requested. (read 3 registers 40108 to 40110)

7687: The CRC (cyclic redundancy check) for error checking.

Removing the SlaveID and CRC gives the PDU:

03 006B 0003

MODBUS TCP/IP – ramka – przykład - krok 2

The equivalent request to this Modbus RTU example

```
11 03 006B 0003 7687
```

in Modbus TCP is:

```
0001 0000 0006 11 03 006B 0003
```

0001: Transaction Identifier

0000: Protocol Identifier

0006: Message Length (6 bytes to follow)

11: The Unit Identifier (17 = 11 hex)

03: The Function Code (read Analog Output Holding Registers)

006B: The Data Address of the first register requested. (40108-40001 = 107 = 6B hex)

0003: The total number of registers requested. (read 3 registers 40108 to 40110)

MODBUS – podsumowanie

Protokół MODBUS - podsumowanie końcowe

MODBUS – podsumowanie

- ▶ Zaimplementowane „proste” rozwiązania
- ▶ Jawna specyfikacja protokołu
- ▶ Zabezpieczenia przesyłanych komunikatów przed błędami
- ▶ Potwierdzenie wykonania rozkazów zdalnych oraz sygnalizacja błędów
- ▶ Skuteczne mechanizmy zabezpieczające przed zawieszeniem systemu

MODBUS – podsumowanie

- ▶ Reguła dostępu do łącza oparta na zasadzie master-slave
- ▶ Zaimplementowane 1, 2 i 7 warstwa modelu ISO/OSI
- ▶ Dwa różne tryby transmisji: ASCII, RTU
- ▶ Możliwość realizacji poprzez:
 - ▶ RS-232 lub RS-485
 - ▶ TCP/IP
 - ▶ Modbus plus
- ▶ Transakcja składa się z polecenia (query) wysyłanego z jednostki master do jednostki slave oraz z odpowiedzi (response) przesyłanej z jednostki slave do jednostki master

BIBLIOGRAFIA

- ▶ Modbus Application Protocol Specification
<http://www.modbus.org>
- ▶ Modbus Messaging Implementation Guide
<http://www.modbus.org>
- ▶ Modbus over serial line
<http://www.modbus.org>
- ▶ W. Mielczarek. Szeregowe interfejsy cyfrowe. Helion, Gliwice 1993.

Dziękuję za uwagę !!!