POLYTECH ANGERS

université angers

Industrial Networks

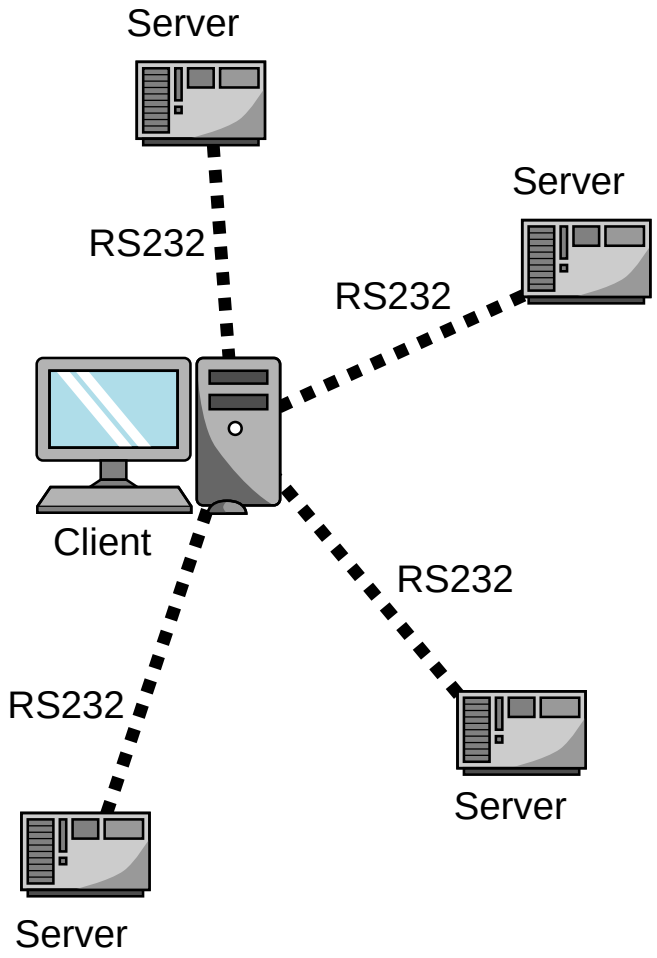
ModBus TCP/IP

Introduction

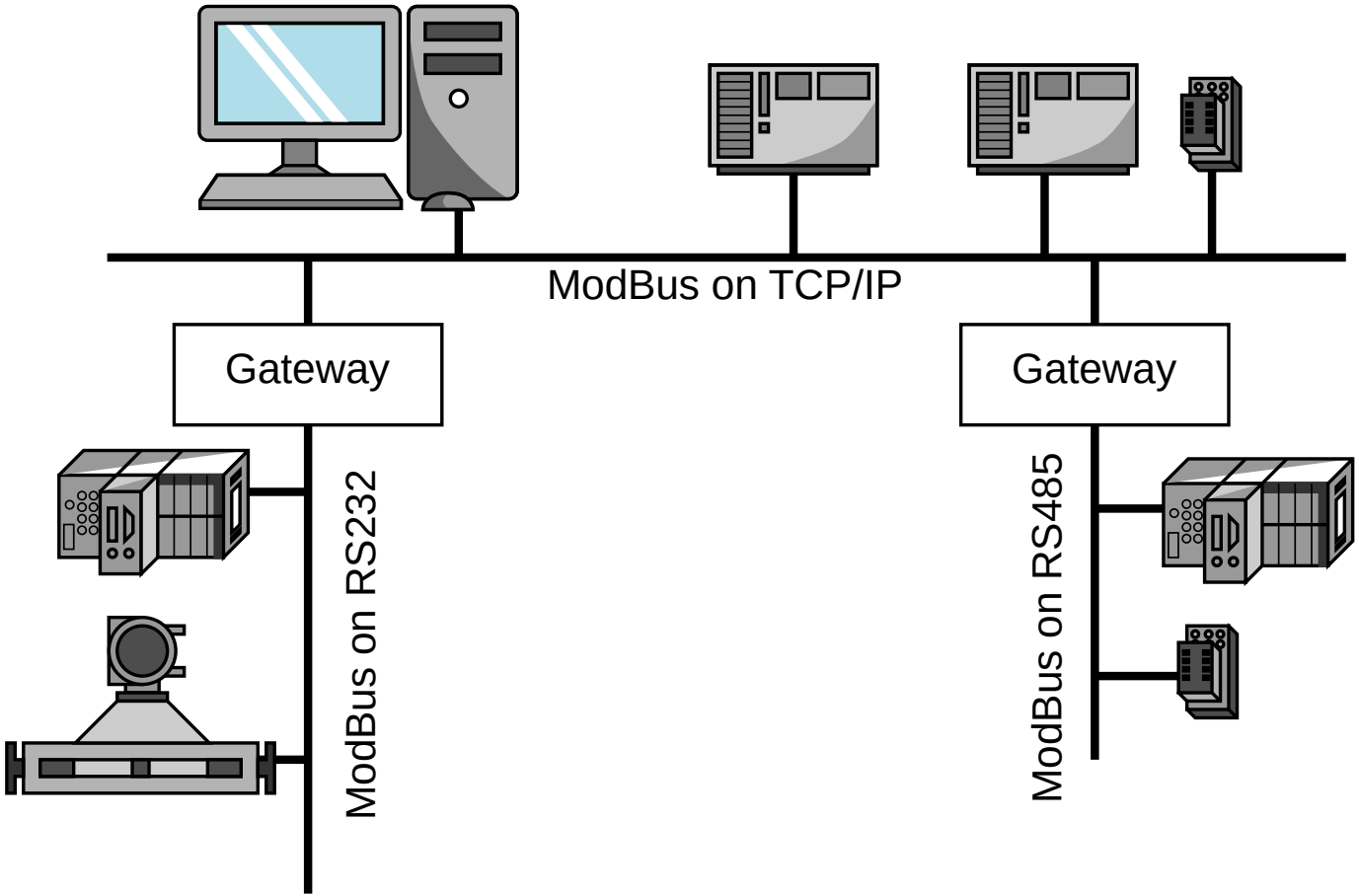
- What is ModBus?
  - Communication protocol
  - Open protocol without royalties
  - Not restricted to industrial automation (home automation)
  - Documentation available for free
    - Modbus Application Protocol Specification
    - Modbus over Serial Line Specification and Implementation Guide
    - The Modbus Messaging on TCP/IP Implementation Guide
- A little history
  - Developed by Modicom
  - 1979

Initial Topology

- Client/server approach
  - 1 client (Computer, programming device)
  - 247 servers (PLCs Modicom)
- At the beginning: One serial RS232 port/server
  - Star topology



Current topology

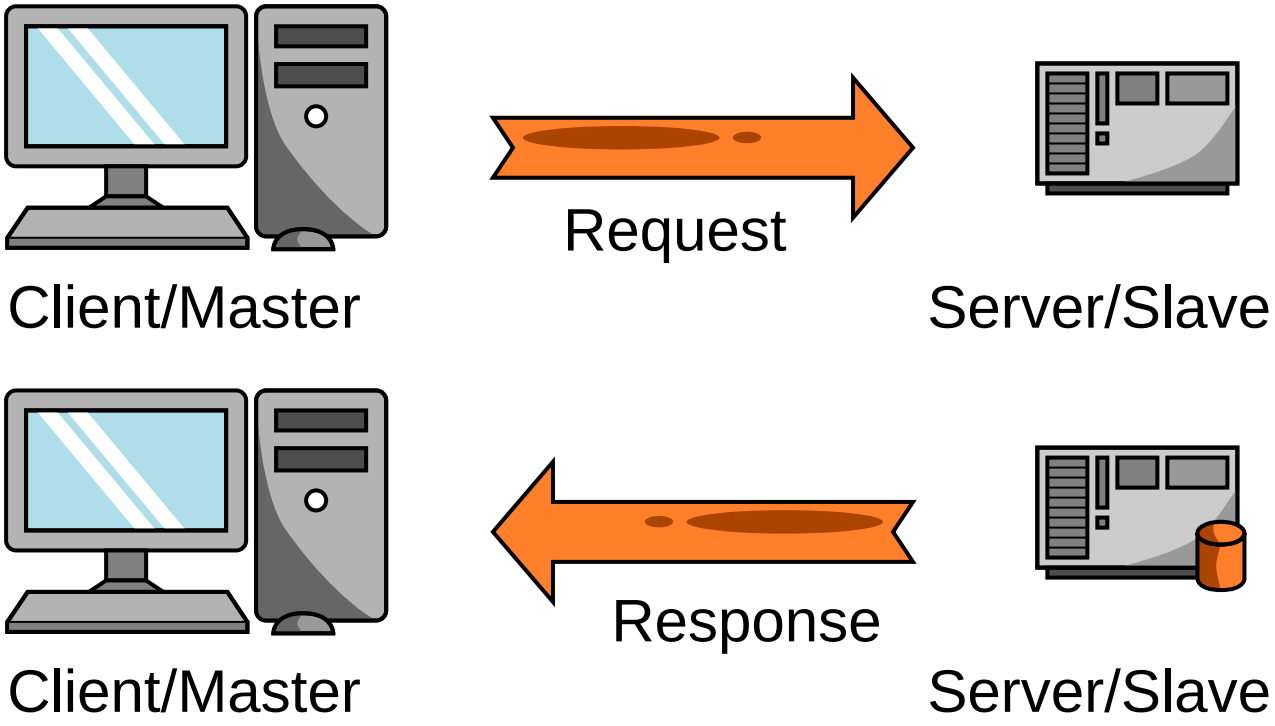


Presentation

- Only the client can initiate a message
- Addressing
  - The client does not have an address
  - The servers are numbered from 1 to 247
  - Address 0 is reserved for broadcasting

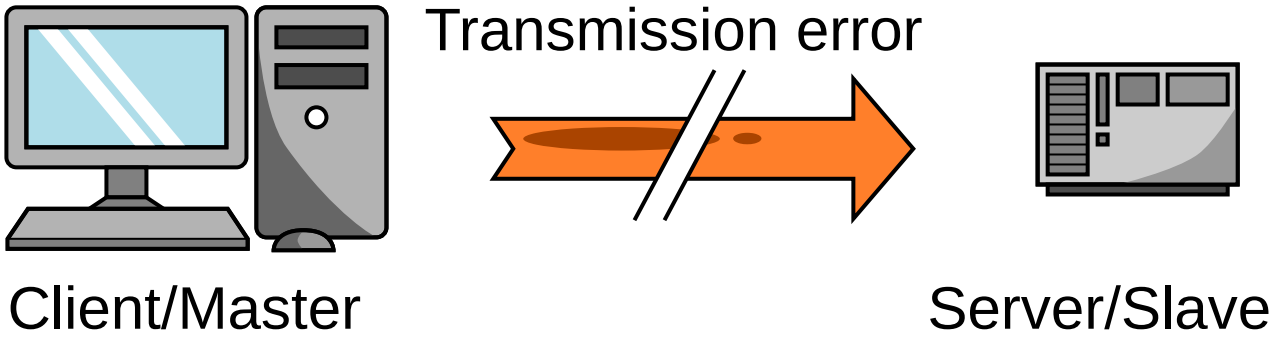
Presentation

IMPORTANT



- When broadcasting servers do not respond

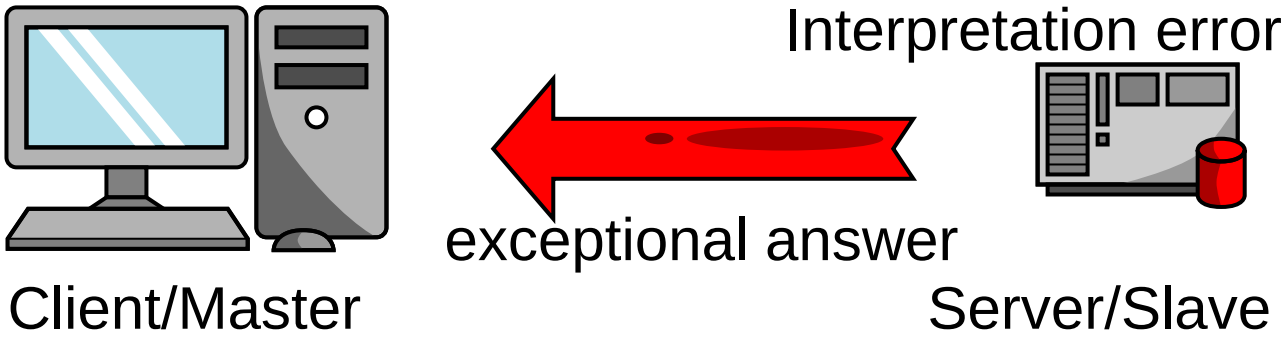
Presentation



- The server remains silent
- Reiteration after a waiting time
- The server is declared out of service after three tries

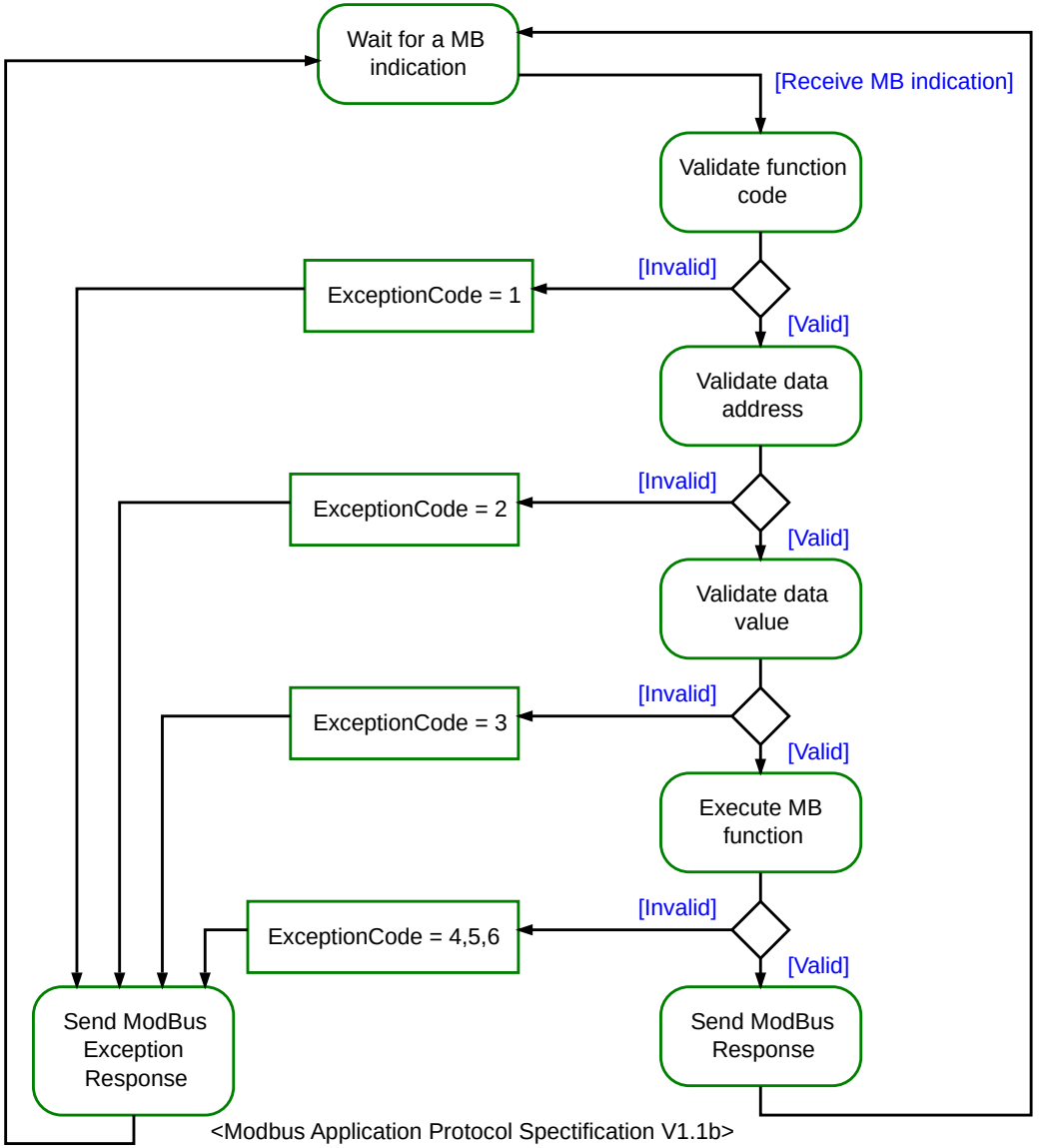
Presentation

IMPORTANT



- Request interpretation problem
- Exception code
  - 1: Unknown function, 2: Incorrect address…

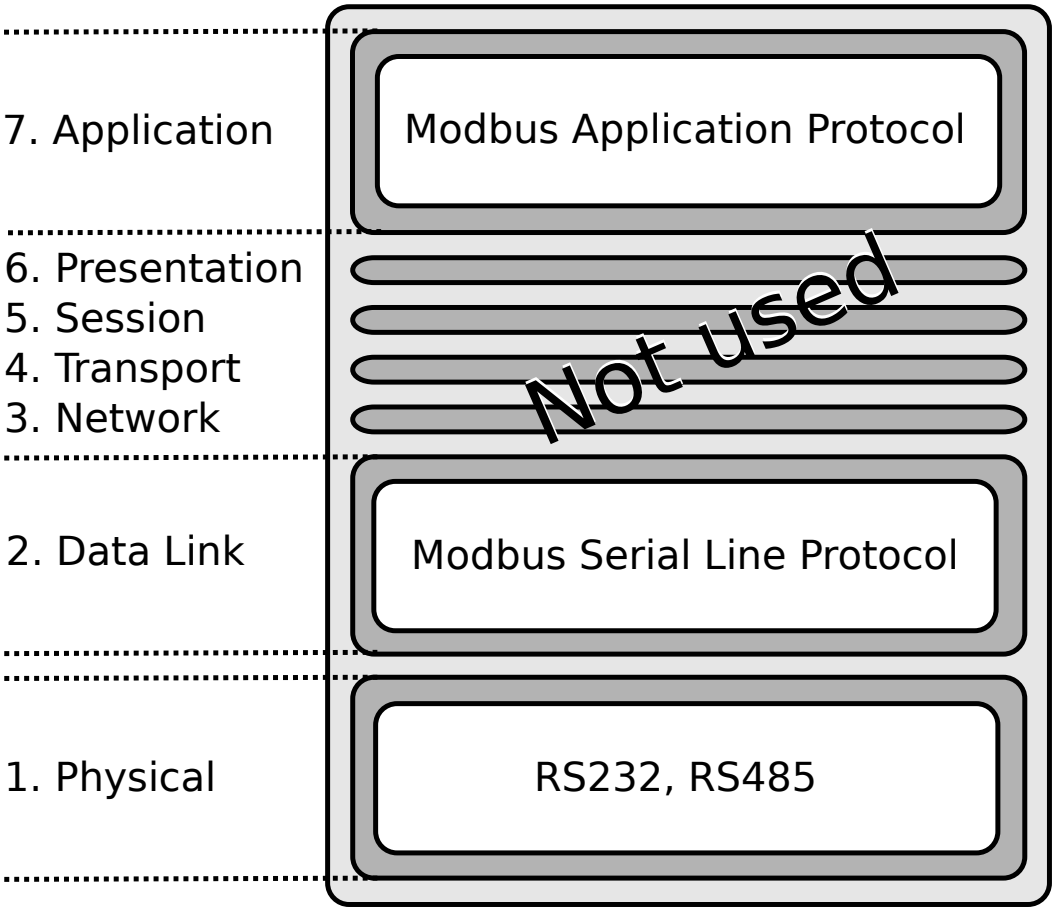
ModBus Transaction state diagram



<Modbus Application Protocol Specification V1.1b>

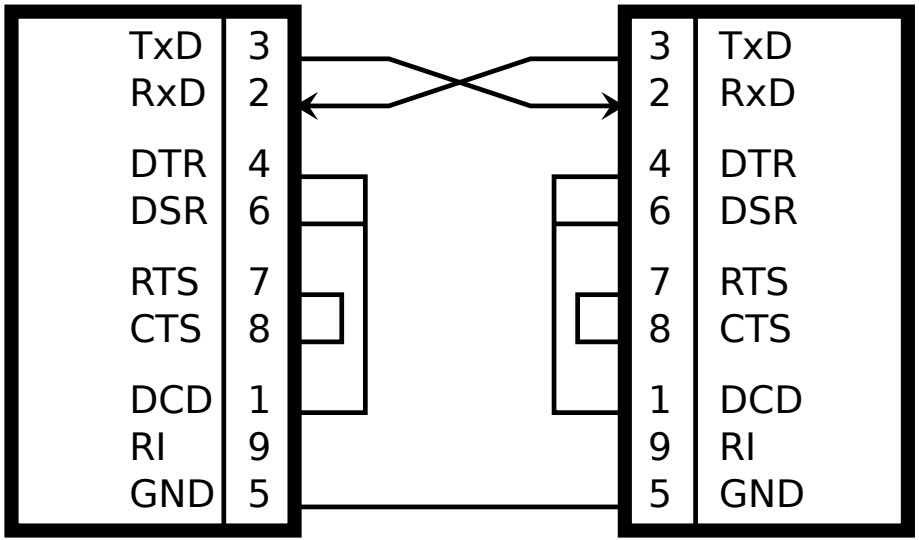
Serial ModBus

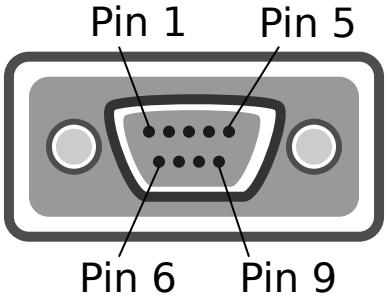
- ModBus and the ISO model



Serial ModBus – Physical layer

- RS232
  - Simple wiring





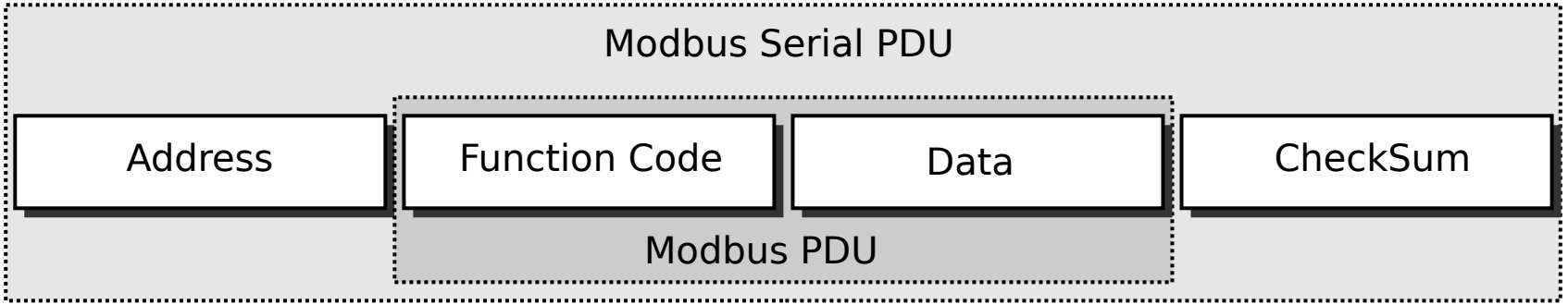
SUB-D 9 male plug

1	DCD	Data Carrier Detect
2	RxD	Received Data
3	TxD	Transmit Data
4	DTR	Data Terminal Ready
5	GND	Ground
6	DSR	Data Set Ready
7	RTS	Request to Send
8	CTS	Clear to Send
9	RI	Ring Indicator

Serial ModBus – Physical layer

- For 8 useful bits : 10 to 12 bits sent
  - 1 start bit
  - 1, 1.5, 2 stop bits
  - 1 parity bit
- Transmission from 75 to 115.2K baud
- ‘0’ => +[5 ; 25]V (12V over com1/com2 of a PC)
- ‘1’ => -[25 ; 5]V

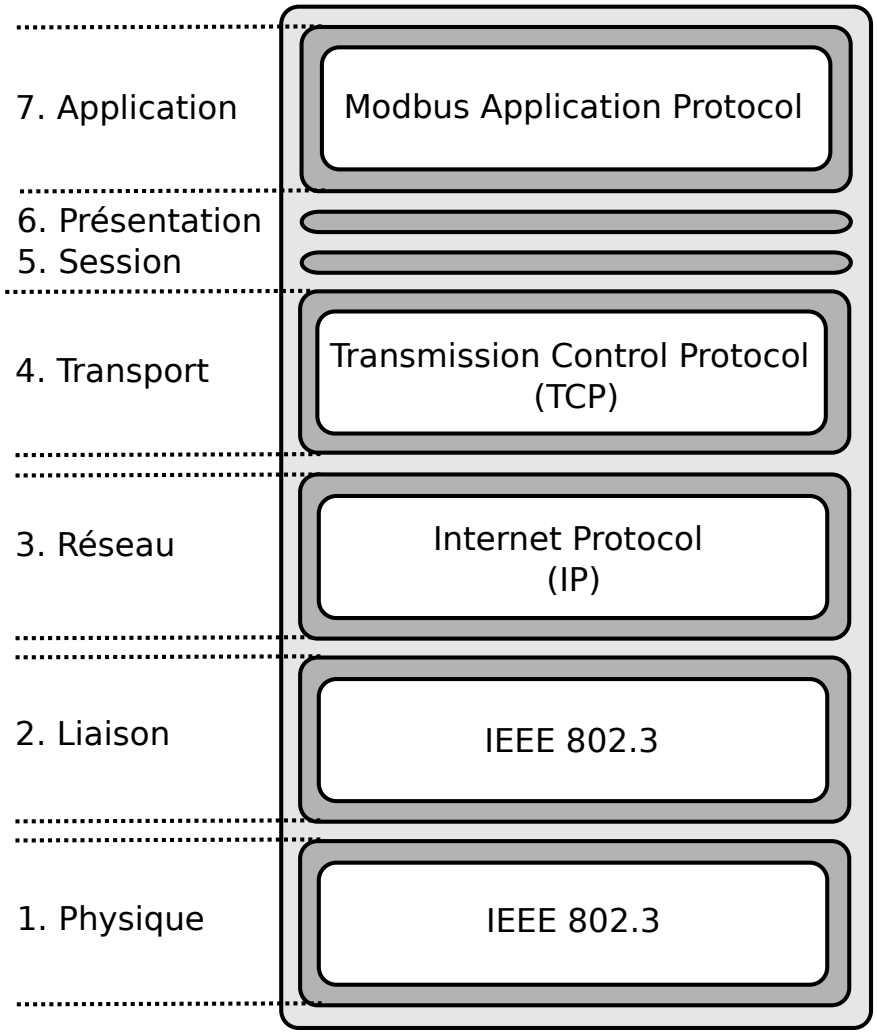
Serial ModBus – Data Link and Application Layers



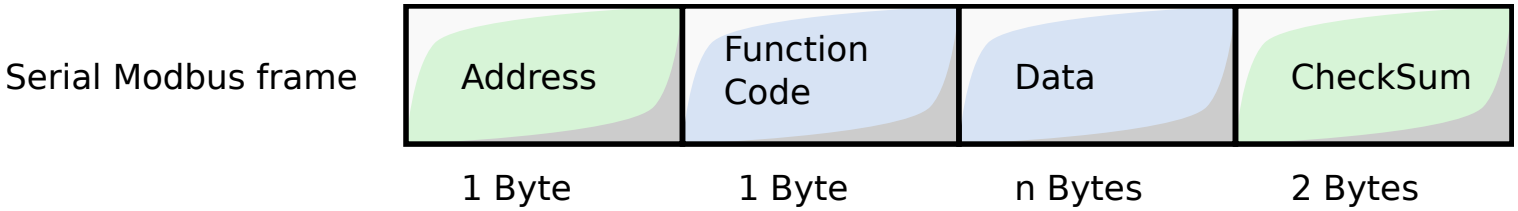
PDU: Protocol Data Unit

ModBus TCP/IP

- ModBus TCP/IP and the OSI model



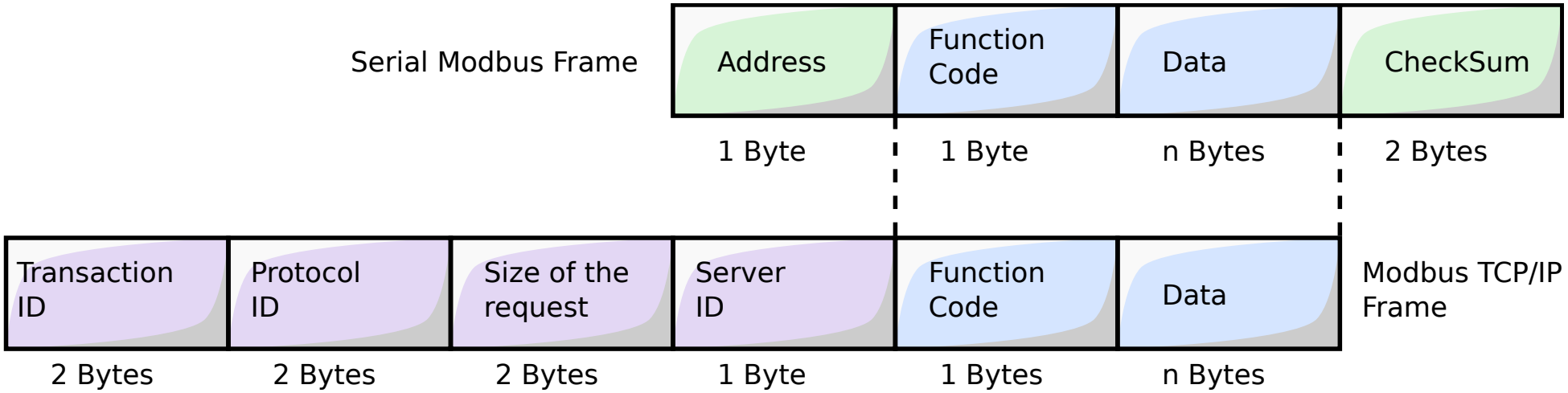
Serial ModBus



- Address : Server address
- Function code : The ModBus fonction
- Data : This field depends on the function code
- Checksum : Check whether or not there is a problem in the frame transmission

Serial ModBus

IMPORTANT



- Transaction ID: Used when multiple ModBus transactions are sent
- Protocol ID : 0 for ModBus service
- Query Size : The number of bytes of the rest of the query
- Server Identification : Used to locate a server that is not on the TCP/IP network

ModBus Functions

				Fct Codes (hex)
Data Access	Bit Access	Physical Discrete Inputs	Read Discrete Inputs	02
		Internal Bits or Physical Coils	Read Coils	01
			Write Single Coil	05
			Write Multiple Coils	0F
	16 Bits Access	Physical Input Register	Read Input Registers	04
		Internal Register Or Physical Output Registers	Read Holding Registers	03
			Write Single Register	06
			Write Multiple Registers	10
			Read/Write Multiple Registers	17
			Mask Write Registers	16
			Read FIFO queue	18
	File record access		Read File record	14
			Write File record	15
Diagnostics			Read Exception status	07
			Diagnostic	08
			Get Com event counter	0B
			Get Com event Log	0C
			Report Slave ID	11
			Read device Identification	2B
Other			Encapsulated Interface Transport	2B

ModBus Functions

- Write single register
  - Request

Function code	1 Byte	0x06
Register Address	2 Bytes	0x0000 to 0xFFFF
Register Value	2 Bytes	0x0000 to 0xFFFF

  - Response

Function code	1 Byte	0x06
Register Address	2 Bytes	0x0000 to 0xFFFF
Register Value	2 Bytes	0x0000 to 0xFFFF

ModBus Functions

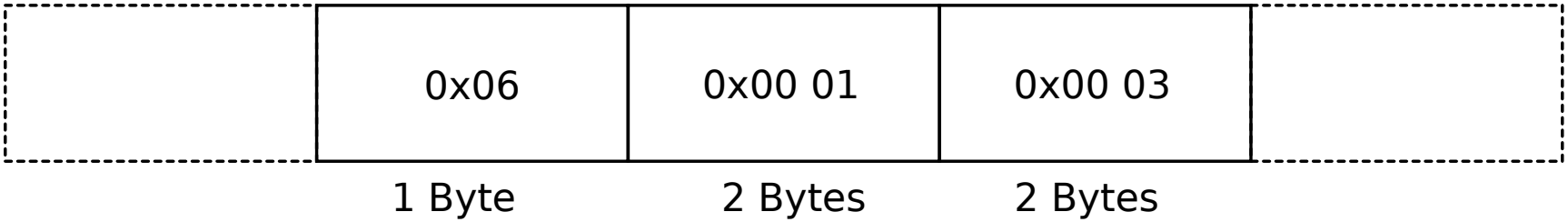
- Write single register
  - Write the value 3 in the register 1

Function code	1 Byte	0x06
Register Address	2 Bytes	0x0000 to 0xFFFF
Register Value	2 Bytes	0x0000 to 0xFFFF

ModBus Functions

- Write single register
  - Write the value 3 in the register 1

Request and response



ModBus Functions

- Write multiple registers
  - Request

Function Code	1 Byte	0x10
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Registers	2 Bytes	0x0001 to 0x007B
Byte Count	1 Byte	2 * N
Registers Value	N * 2 Bytes	value

  - Response

Function Code	1 Byte	0x10
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Registers	2 Bytes	0x0001 to 0x007B

ModBus Functions

- Write multiple registers
  - Writing two registers from register 5, values $(10)_{10}$ and $(258)_{10}$

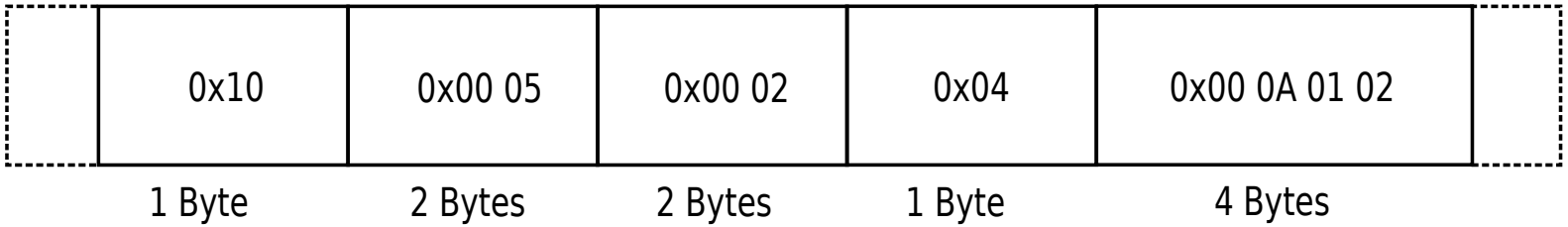
Function Code	1 Byte	0x10
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Registers	2 Bytes	0x0001 to 0x007B
Byte Count	1 Byte	2 * N
Registers Value	N * 2 Bytes	value

Function Code	1 Byte	0x10
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Registers	2 Bytes	0x0001 to 0x007B

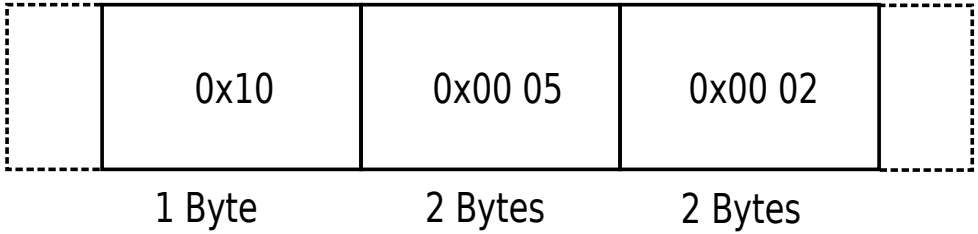
ModBus Functions

- Write multiple registers
  - Writing two registers from register 5, values $(10)_{10}$ and $(258)_{10}$

Request



Response



ModBus Functions

- Read multiple registers
  - Request

Function Code	1 Byte	0x03
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Registers	2 Bytes	1 to 125 (0x7D)

  - Response

Function Code	1 Byte	0x03
Byte count	1 Byte	2 * N
Register value	N * 2 Bytes	

ModBus Functions

- Read multiple registers
  - We want to read from register 260 to register 262
  - What is the expected answer knowing that the register values are 103, 532 and 702?

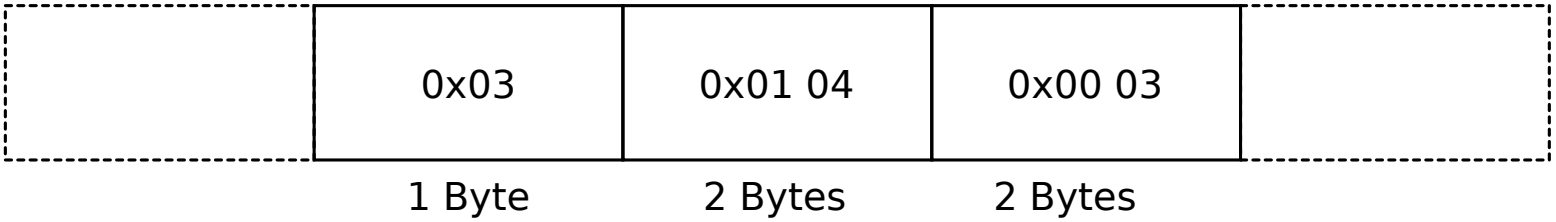
Function Code	1 Byte	0x03
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Registers	2 Bytes	1 to 125 (0x7D)

Function Code	1 Byte	0x03
Byte count	1 Byte	2 * N
Register value	N * 2 Bytes	

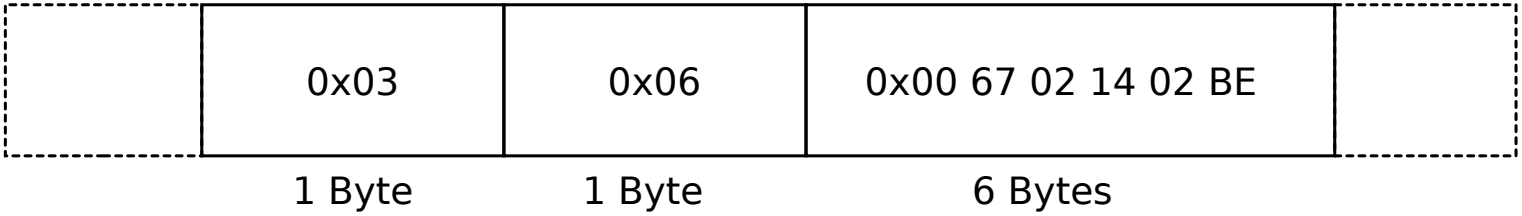
ModBus Functions

- Read multiple registers
  - We want to read from register 260 to register 262
  - What is the expected answer knowing that the register values are 103, 532 and 702?

Request



Response



ModBus Functions

- Write single coil
  - Request

Function Code	1 Byte	0x05
Output Address	2 Bytes	0x0000 to 0xFFFF
Output Value	2 Bytes	0x0000 or 0xFF00

  - Response

Function Code	1 Byte	0x05
Output Address	2 Bytes	0x0000 to 0xFFFF
Output Value	2 Bytes	0x0000 or 0xFF00

ModBus Functions

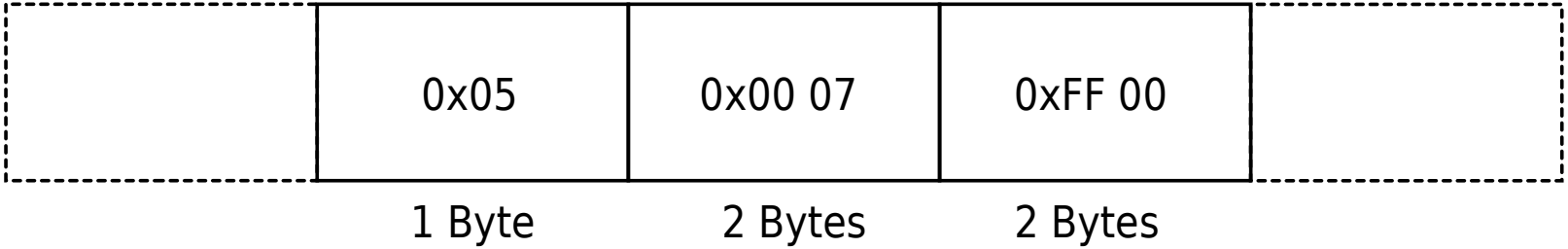
- Write single coil
  - Forcing the coil 7 to 1

Function Code	1 Byte	0x05
Output Address	2 Bytes	0x0000 to 0xFFFF
Output Value	2 Bytes	0x0000 or 0xFF00

ModBus Functions

- Write single coil
  - Forcing the coil 7 to 1

Request and response



ModBus Functions

- Read N coils
  - Request

Function Code	1 Byte	0x01
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Coils	2 Bytes	1 to 2000 (0x7D0)

  - Response

Function Code	1 Byte	0x01
Byte Count	1 Byte	N
Coil Status	n Byte	n = N or N+1

ModBus Functions

- Read N coils
  - Read from the coil 20 to 38
  - What is the response knowing that the coils 37, 35, 32, 30, 25, 24 and 21 are set to 0, and the other are set to 1.

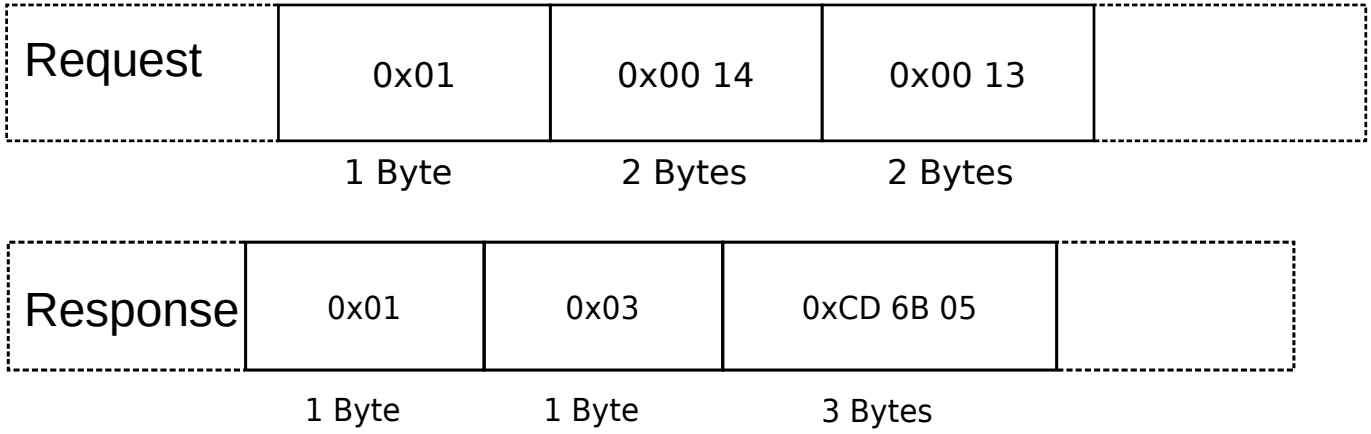
Function Code	1 Byte	0x01
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Coils	2 Bytes	1 to 2000 (0x7D0)

Function Code	1 Byte	0x01
Byte Count	1 Byte	N
Coil Status	n Byte	n = N or N+1

The coil status are defined byte by byte, from the LSB to the MSB (0 if not defined)

ModBus Functions

- Read N coils
  - Read from the coil 20 to 38
  - What is the response knowing that the coils 37, 35, 32, 30, 25, 24 and 21 are set to 0, and the other are set to 1.



ModBus Functions

- Write multiple coils
  - Request

Function Code	1 Byte	0x0F
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Outputs	2 Bytes	0x0001 to 0x07B0
Byte Count	1 Byte	N
Outputs Value	N * 1 Byte	

  - Response

Function Code	1 Byte	0x0F
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Outputs	2 Bytes	0x0001 to 0x07B0

ModBus Functions

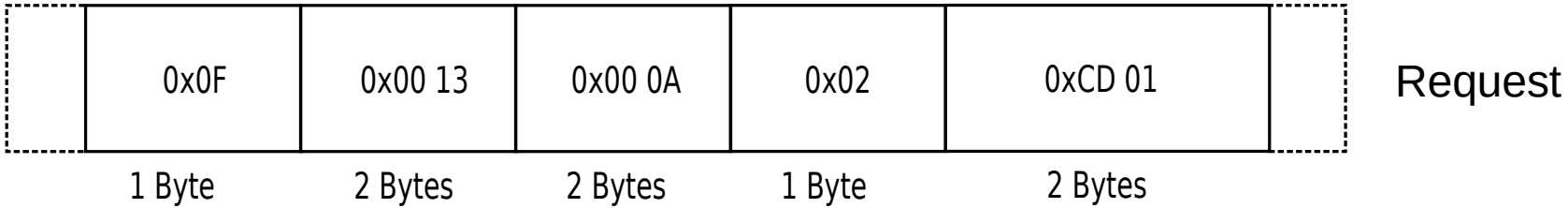
- Write multiple coils
  - Write 10 bits from coil 19
  - Coils 28, 24, 23 and 20 should be equal to 0, and the others to 1

Function Code	1 Byte	0x0F
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Outputs	2 Bytes	0x0001 to 0x07B0
Byte Count	1 Byte	N
Outputs Value	N * 1 Byte	

Function Code	1 Byte	0x0F
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Outputs	2 Bytes	0x0001 to 0x07B0

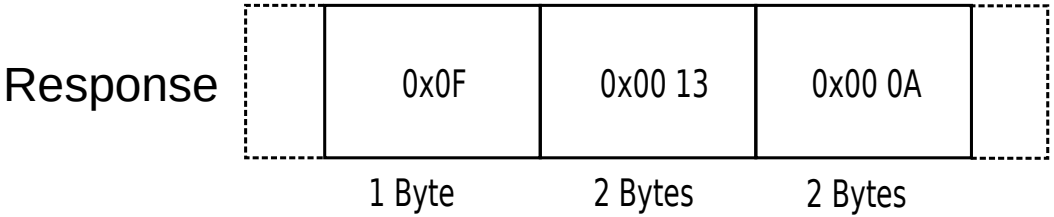
ModBus Functions

- Write multiple coils
  - Write 10 bits from coil 19
  - Coils 28, 24, 23 and 20 should be equal to 0, and the others to 1



Bit distributions in bytes

26	25	24	23	22	21	20	19	X	X	X	X	X	28	27
1	1	0	0	1	1	0	1	0	0	0	0	0	0	1



Exception responses

IMPORTANT

- If the message received by the slave does not match the message sent by the master then the slave does not respond
- If the received message is correct but the slave can not process it (unknown function code, address not correct…) then the latter returns an exception response
- Some exception codes
  - 1 : Unknown function
  - 2 : Incorrect address
  - 3 : Incorrect data
  - 4 : PLC not ready…