Industrial Automation
(Automação de Processos Industriais)

PLC Programming languages
Structured Text - Networking

http://users.isr.ist.utl.pt/~jag/courses/api1920/api1920.html

Prof. José Gaspar, 2019/2020

Structured Text

Networking (in Unity Pro)

Keywords: MODBUS, READ_VAR, WRITE_VAR

Modbus is a serial communications protocol originally published by Modicon (now Schneider Electric) in 1979 for use with its programmable logic controllers (PLCs). Simple and robust, it has since become a de facto standard communication protocol, and it is now a commonly available means of connecting industrial electronic devices.

Examples of Field Bus (IEC 61158) standards: MODBUS (Schneider), PROFIBUS (Field Bus type, Siemens), CAN bus (Controller Area Network, 1983 Robert Bosch GmbH), ...

Structured Text *Networking (in Unity Pro)*

Modbus RTU — Binary representation of the data for protocol communication. Includes CRC. Modbus messages are framed (separated) by idle (silent) periods.

Modbus ASCII — Makes use of ASCII characters for protocol communication.

Modbus TCP/IP or Modbus TCP — Modbus variant for communications over TCP/IP networks, connecting over port 502.

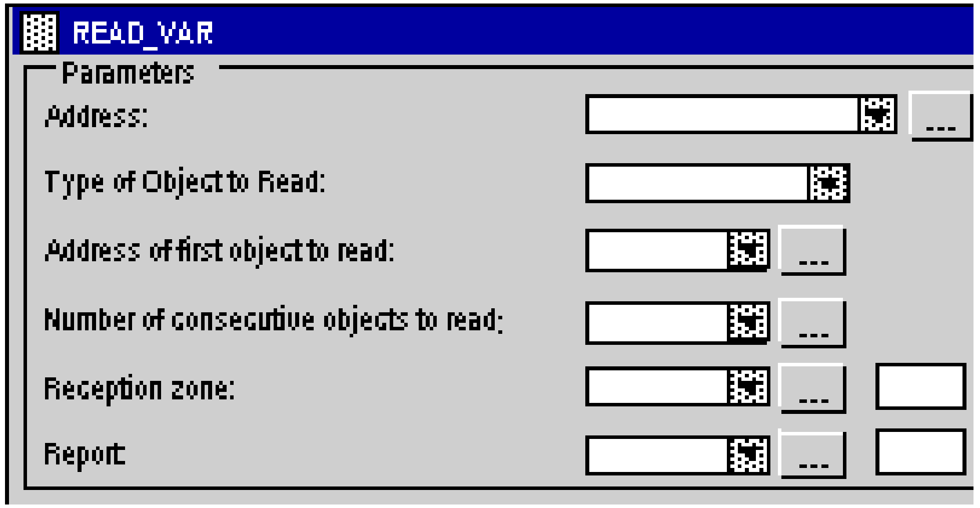
RTU = Remote Terminal Unit
MTU = Main Terminal Unit
CRC = Cyclic Redundancy Check
TCP = Transmission Control Protocol
ASCII = American Standard Code for Information Interchange

Structured Text

Networking (in Unity Pro)

Modbus	Function type	Function name / Function code	
Bit access	Physical Discrete Inputs	**Read Discrete Inputs**	2
	Internal Bits or Physical Coils	**Read Coils**	1
		Write Single Coil	5

Structured Text *Networking (in Unity Pro) – READ_VAR*



Address:
ADDR(STRING)
ARRAY [0..5] OF INT

Type of object to read:
'%M' for reading internal bits
'%MW' for reading internal words
'%S' for reading system bits
'%SW' for reading system words
'%I' for reading input bits
'%IW' for reading input words

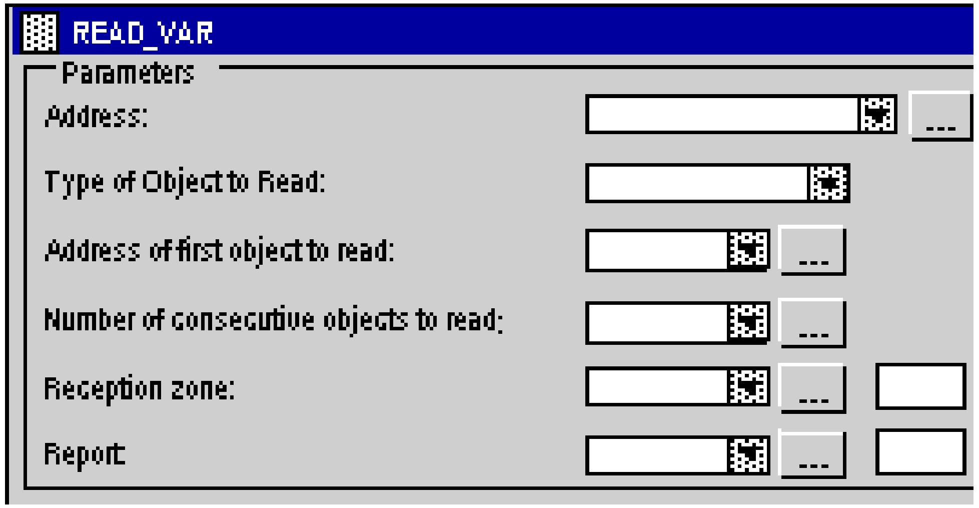
Address of first object to read:
The possible objects are of the DINT type (variables, constants, immediate value)

Number of consecutive objects to read:
The possible objects are of the INT type (variables, constants, immediate value)

Reception zone:
The reception zone is an integer array. The size of this array depends on the number of objects to read. This integer array can be located or not.

Report: The report is an array of 4 integers

Structured Text　　*Networking (in Unity Pro) – READ_VAR*



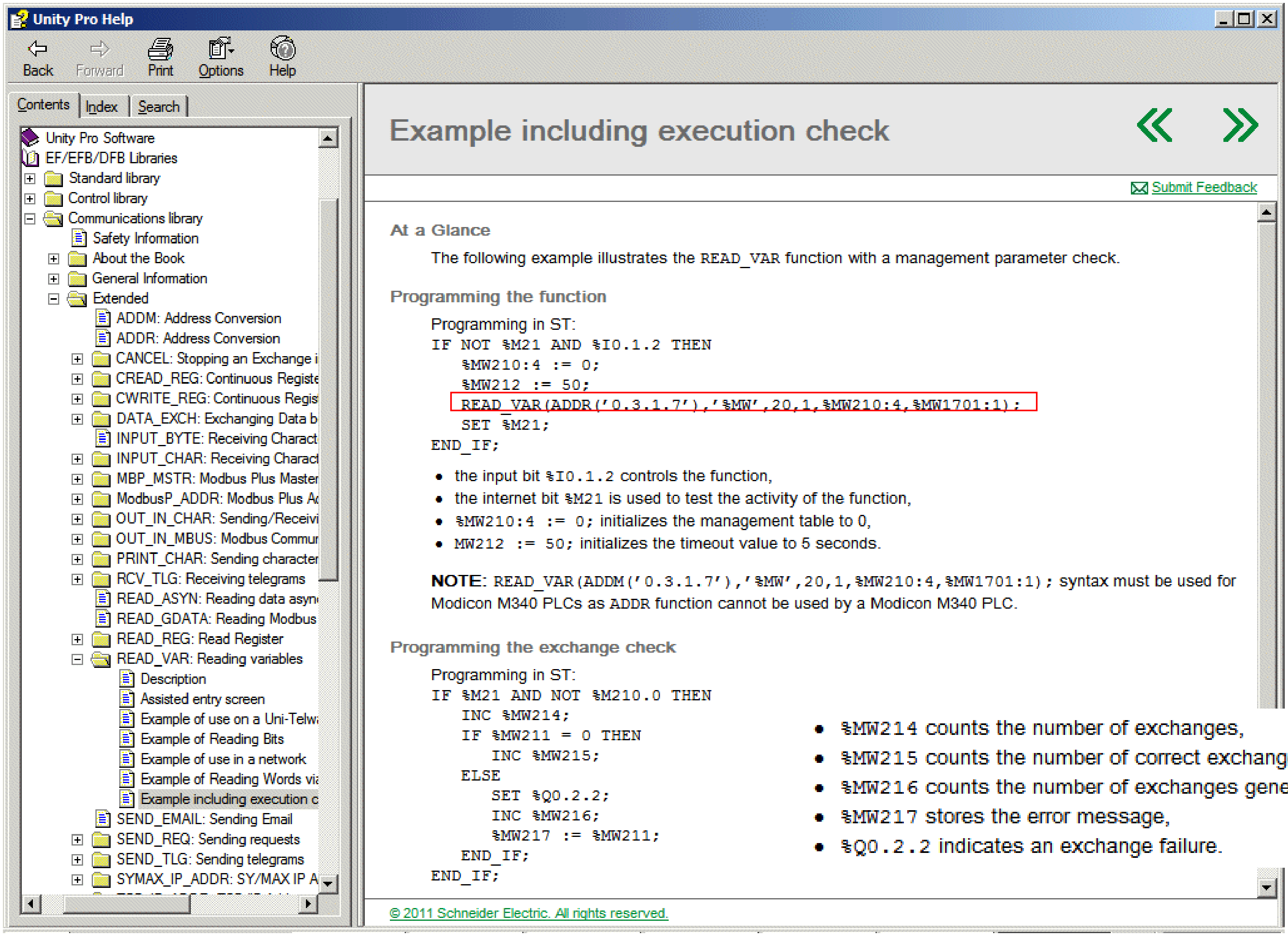
READ_VAR

Parameters

Address:	
Type of Object to Read:	
Address of first object to read:	
Number of consecutive objects to read:	
Reception zone:	
Report:	

Challenge: how to make READ_VAR non-blocking in an operating system without using processes nor threads?

Structured Text *Networking (in Unity Pro)*


Unity Pro Help
Back
Forward
Print
Options
Help
Contents
Index
Search
Unity Pro Software
EF/EFB/DFB Libraries
Standard library
Control library
Communications library
Safety Information
About the Book
General Information
Extended
ADDM: Address Conversion
ADDR: Address Conversion
CANCEL: Stopping an Exchange i
CREAD_REG: Continuous Registe
CWRITE_REG: Continuous Regis
DATA_EXCH: Exchanging Data b
INPUT_BYTE: Receiving Charact
INPUT_CHAR: Receiving Charact
MBP_MSTR: Modbus Plus Master
ModbusP_ADDR: Modbus Plus A
OUT_IN_CHAR: Sending/Receivi
OUT_IN_MBUS: Modbus Commur
PRINT_CHAR: Sending character
RCV_TLG: Receiving telegrams
READ_ASYN: Reading data asyn
READ_GDATA: Reading Modbus
READ_REG: Read Register
READ_VAR: Reading variables
Description
Assisted entry screen
Example of use on a Uni-Telwa
Example of Reading Bits
Example of use in a network
Example of Reading Words via
Example including execution c
SEND_EMAIL: Sending Email
SEND_REQ: Sending requests
SEND_TLG: Sending telegrams
SYMAX_IP_ADDR: SY/MAX IP A
Submit Feedback

Example including execution check

At a Glance

The following example illustrates the `READ_VAR` function with a management parameter check.

Programming the function

Programming in ST:

```
IF NOT %M21 AND %I0.1.2 THEN
   %MW210:4 := 0;
   %MW212 := 50;
   READ_VAR(ADDR('0.3.1.7'),'%MW',20,1,%MW210:4,%MW1701:1);
   SET %M21;
END_IF;
```

- the input bit `%I0.1.2` controls the function,
- the internet bit `%M21` is used to test the activity of the function,
- `%MW210:4 := 0;` initializes the management table to 0,
- `MW212 := 50;` initializes the timeout value to 5 seconds.

NOTE: `READ_VAR(ADDM('0.3.1.7'),'%MW',20,1,%MW210:4,%MW1701:1);` syntax must be used for Modicon M340 PLCs as `ADDR` function cannot be used by a Modicon M340 PLC.

Programming the exchange check

Programming in ST:

```
IF %M21 AND NOT %M210.0 THEN
   INC %MW214;
   IF %MW211 = 0 THEN
      INC %MW215;
   ELSE
      SET %Q0.2.2;
      INC %MW216;
      %MW217 := %MW211;
   END_IF;
END_IF;
```

© 2011 Schneider Electric. All rights reserved.

- `%MW214` counts the number of exchanges,
- `%MW215` counts the number of correct exchanges,
- `%MW216` counts the number of exchanges generating errors,
- `%MW217` stores the error message,
- `%Q0.2.2` indicates an exchange failure.