

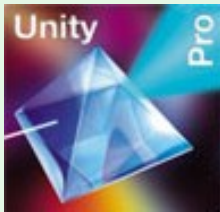


# TCPOPEN

## Modicon X80 PMEUCM0302 Application



PMEUCM0302



Collaborative Automation  
by  
**Schneider**  
Electric

**Niobrara Research & Development Corporation**  
PO Box 3418  
Joplin, MO 64803  
800-235-6723  
Tel: +1 417-624-8918  
Fax: +1 417-624-8920  
[www.niobrara.com](http://www.niobrara.com)

©2019  
**Niobrara Research & Development Corporation**

### Overview

Niobrara's PMEUCM is a programmable communications module for Schneider Electric's Modicon M580 PAC. The TCCPOPEN application for the PMEUCM allows the M580 CPU to control up to 16 Ethernet sockets as either Client or Server. Available with two RS-232 serial ports and two Ethernet ports, the PMEUCM may allow the M580 PLC to communicate with a variety of external devices. The PMEUCM uses the Ethernet backplane to communicate with the M580 CPU or eNOC DIO master. Configuration is done through a DTM in Unity Pro. A LCD display and joystick make front panel troubleshooting quick and easy.

### TCP and UDP Sockets

The TCCPOPEN application allows PLC code to control each socket which may be configured to be a server (listen) or a client (connect). Settings such as local and remote TCP (or UDP) port numbers, IP Addresses, connection status, are all controlled through the IODDT structure provided by the DTM. DFBs are provided to automatically control moving the socket data to/from arrays of bytes for each socket.

### Sample M580 ST Code

Sample Unity Pro code is provided to demonstrate a Modbus/TCP Server, Modbus/TCP Client, and Telnet Server that is written in structured text. The M580 Modbus/TCP server provides Holding Register data for 10 virtual slaves with FC3 reads and FC16 writes.

```

1) UCM1_Modbus_Server -SR- (MAST)
(* Note: This simple example Modbus/TCP server only allows a single Modbus/TCP message per Ethernet packet. *)
if (UCM1_Modbus_Control[UCM_socketnumber].Zero = 0) and (UCM1_Data_IN_Length[UCM_socketnumber] = UCM1_Modbus_Control[UCM_socketnumb
(* Increment a counter for the individual socket number... *)
Modbus_Slaves[UCM1_Modbus_Control[UCM_socketnumber].RXSlave] := Modbus_Slaves[UCM1_Modbus_Control[UCM_soc
for UCM_Y := 1 to 10 do (* Register 19 is the total message counter for all server sockets on all slaves *)
  Modbus_Slaves[UCM_Y][19] := Modbus_Slaves[UCM_Y][19] + 1;
end_for;

UCM1_Data_OUT[UCM_socketnumber][6] := UCM1_Data_IN[UCM_socketnumber][16]; (* Slave *)
UCM1_Data_OUT[UCM_socketnumber][7] := UCM1_Data_IN[UCM_socketnumber][7]; (* Opcode *)

if (UCM1_Modbus_Control[UCM_socketnumber].RXSlave = 1) and (UCM1_Modbus_Control[UCM_socketnumber].RXSlave <= 10) then
  case UCM1_Modbus_Control[UCM_socketnumber].Opcode of
  3: (* Function Code 3 Read Holding Registers *)
    UCM1_Modbus_Control[UCM_socketnumber].Read_Start_Register := word_to_int(byte_as_word(UCM1_Data_In[UCM_sock
    UCM1_Modbus_Control[UCM_socketnumber].Read_Count := word_to_int(byte_as_word(UCM1_Data_In[UCM_sock
    UCM1_Data_OUT[UCM_socketnumber][8] := int_to_byte(UCM1_Modbus_Control[UCM_socketnumber].Read_Count
    UCM1_Modbus_Control[UCM_socketnumber].Payload_Length_OUT := (UCM1_Modbus_Control[UCM_socketnumber].
    UCM1_Data_OUT_Length[UCM_socketnumber] := (UCM1_Modbus_Control[UCM_socketnumber].Read_Count * 2) +
  for UCM_x := 0 to UCM1_Modbus_Control[UCM_socketnumber].Read_Count - 1 do
    if (UCM1_Modbus_Control[UCM_socketnumber].Read_Start_Register + UCM_x) < 500 then
      UCM1_Modbus_Control[UCM_socketnumber].Data[UCM_x] := Modbus_Slaves[UCM1_Modbus_Control[UCM_socketnumber].
      word_as_byte(int_to_word(UCM1_Modbus_Control[UCM_socketnumber].Data[UCM_x]), UCM1_Data
    else
      (* Illegal register number, report exception code 16x02 *)
      UCM1_Modbus_Control[UCM_socketnumber].Error := 2;
    end_if;
  end_for;

16: (* Function Code 16 Write Multiple Holding Registers *)
    UCM1_Modbus_Control[UCM_socketnumber].Payload_Length_OUT := 6;

```

The Modbus/TCP client code shows how to connect to remote servers and generate messages to do Modbus/TCP queries. The telnet server shows a simple server interface that can be accessed from any telnet client.

A second example is more complicated and involves connecting through the Internet to the Weather Underground text server. The client connects to the server, parses the prompt messages and sends replies to gather the local weather conditions based on a USA airport code.

```

UCMI_Weather_Client <SR> : [MAST]

(* Here is the Ethernet frame data as a new string *)
UCMI_Telnet_Inbound_String := ASCII_to_string(UCMI_Telnet_Int);

(* The incoming Telnet stream may be split into multiple Ethernet frames so we rebuild the RX stream into a single
UCMI_Telnet_RX_String := Concat_Str(UCMI_Telnet_RX_String, UCM1_Telnet_Inbound_String);

(* Zero the input length to allow more data to come... *)
UCMI_Data_IN_Length[UCM_socketnumber] := 0;

CASE UCM1_Sock_Control[UCM_socketnumber].Client_State OF

1:   (* Waiting on 'Press Return to continue' *)
    UCM1_Telnet_Search := 'Press Return to continue';
    UCM1_Telnet_TempLocation := FIND_INT(UCMI_Telnet_RX_String, UCM1_Telnet_Search);
    if UCM1_Telnet_TempLocation > 0 then
        (* Shorten the RX stream to remove unwanted text from the beginning *)
        UCM1_Telnet_RX_String := right_int(UCMI_Telnet_RX_String, len_int(UCMI_Telnet_RX_String) - (
        (* send a <CR><LF> to continue *)
        UCM1_Telnet_Outbound_String := '$n';
        UCM1_Sock_Control[UCM_socketnumber].OK_to_SEND := True;
        UCM1_Sock_Control[UCM_socketnumber].Client_State := 2;
    end_if;

2:   (* Waiting on 'enter 3 letter forecast city code' *)
    UCM1_Telnet_Search := 'enter 3 letter forecast city code';
    UCM1_Telnet_TempLocation := FIND_INT(UCMI_Telnet_RX_String, UCM1_Telnet_Search);
    if UCM1_Telnet_TempLocation > 0 then
        (* Shorten the RX stream to remove unwanted text from the beginning *)
        UCM1_Telnet_RX_String := right_int(UCMI_Telnet_RX_String, len_int(UCMI_Telnet_RX_String) - (
        (* send a JLN<CR><LF> for Joplin, MO weather *)
        UCM1_Telnet_Outbound_String := 'JLNSn';
        UCM1_Sock_Control[UCM_socketnumber].OK_to_SEND := True;
        UCM1_Sock_Control[UCM_socketnumber].Client_State := 3;
    end_if;

3:   (* Waiting on 'Weather Conditions at' *)
    UCM1_Telnet_Search := 'Weather Conditions at';
    UCM1_Telnet_TempLocation := FIND_INT(UCMI_Telnet_RX_String, UCM1_Telnet_Search);
    if UCM1_Telnet_TempLocation > 0 then (* Pull out the time *)
        (* Shorten the RX stream to remove unwanted text from the beginning *)
        UCM1_Telnet_RX_String := right_int(UCMI_Telnet_RX_String, len_int(UCMI_Telnet_RX_String) - (

```

The weather example deals with stream data from the remote server spanning multiple Ethernet frames and gives the user some idea of the power available to the M580 source code.

## Other Use Examples

Some ideas implemented by customers using TCPOPEN in the PMEUCM:

- Connecting to a Printer Server using TCP sockets to send barcode print strings
- Reading data from a Positioning System Radar using UDP sockets
- Sending HTTP Post messages to a web server in an IP Speaker to play mp3 files
- Communicating with a remote serial device connected to a terminal server

**Niobrara Research & Development Corporation**  
 PO Box 3418  
 Joplin, MO 64803  
 800-235-6723  
 Tel: +1 417-624-8918  
 Fax: +1 417-624-8920  
[www.niobrara.com](http://www.niobrara.com)



©2019  
 Niobrara Research & Development Corporation