```
PROGRAM PLC_PRG
VAR
    SysFirstLoop: BOOL:=TRUE; (* First program loop *)
    Sp: SysSerialPort; (* Serial port *)
    PSts: SysGetIpInfos; (* Peer status *)
    Svr: SysTCPServer; (* TCP server *)
    WMsgPulse: BOOL; (* Welcome message pulse *)
    RxChars: ARRAY[ 0..1 ] OF INT; (* Auxiliary variable *)
    RxTxStr: STRING(256); (* Rx/Tx string *)
    IChars: ARRAY[ 0..1 ] OF INT; (* Input characters *)
END_VAR
```

```
(* ************************************************************************** *)
(* EMULAZIONE CONVERTITORE ETHERNET/SERIALE                                   *)
(* ************************************************************************** *)
(* Questo programma emula un convertitore Ethernet/Seriale.                   *)
(* -------------------------------------------------------------------------- *)

    (* -------------------------------------------------------------------- *)
    (* ESEGUO INIZIALIZZAZIONI                                              *)
    (* -------------------------------------------------------------------- *)
    (* Eseguo apertura porta seriale e socket TCP. *)

    IF (SysFirstLoop) THEN
        SysFirstLoop:=FALSE; (* First program loop *)

        (* Eseguo apertura e parametrizzazione porta seriale. *)

        Sp.Open:=TRUE; (* Open command *)
        Sp.File:=Sysfopen('COM2', 'rw'); (* File pointer *)
        Sp.Baudrate:=19200; (* 19200 bps *)
        Sp.DataBits:=8; (* 8 data bit *)
        Sp.Parity:='E'; (* Even parity *)
        Sp.StopBits:=1; (* 1 stop bit *)
        Sp.DTRManagement:=DTR_AUTO_WO_TIMES; (* DTR auto managed *)
        Sp.DTRComplement:=FALSE; (* DTR not complemented *)
        Sp.DTROnTime:=0; (* On time set to 0 *)
        Sp.DTROffTime:=0; (* Off time set to 0 *)

        (* Eseguo apertura e parametrizzazione socket TCP/IP. *)

        PSts.File:=Sysfopen('TCPSKT', 'rw'); (* File pointer *)
        Svr.Enable:=TRUE; (* Enable command *)
        Svr.Files:=ADR(PSts.File); (* File pointer *)
        Svr.PeerIPEn:='192.168.0.255'; (* Accept all IPs in the range 192.168.0.xxx *)
        Svr.PeerPortEn:=16#FFFF; (* Accept all ports *)
        Svr.Port:=1000; (* Listening port *)
        Svr.MaxConn:=1; (* Number of accepted connections *)
```

```
0037        Svr.LifeTm:=30; (* Life time (S) *)
0038    END_IF;
0039
0040    (* ---------------------------------------------------------------------- *)
0041    (* GESTIONE PORTA SERIALKE E SERVER TCP                                   *)
0042    (* ---------------------------------------------------------------------- *)
0043    (* Gestione porta seriale. *)
0044
0045    Sp(); (* Manage the serial port *)
0046    IF NOT(Sp.Opened) THEN RETURN; END_IF;
0047
0048    (* Gestione server TCP. *)
0049
0050    Svr(); (* Manage the TCP/IP server *)
0051    PSts(); (* Peer status *)
0052    IF NOT(SysIsFOpen(PSts.File)) THEN WMsgPulse:=FALSE; RETURN; END_IF;
0053
0054    (* ---------------------------------------------------------------------- *)
0055    (* GESTIONE CONNESSIONE TCP                                               *)
0056    (* ---------------------------------------------------------------------- *)
0057    (* Su connessione TCP il sistema invia un messaggio di Welcome. *)
0058
0059    IF NOT(WMsgPulse) THEN
0060        WMsgPulse:=TRUE; (* Welcome message pulse *)
0061        RxTxStr:='Welcome !$r$n'; (* Rx/Tx string *)
0062        Sysfwrite(ADR(RxTxStr), LEN(RxTxStr), 1, PSts.File);
0063    END_IF;
0064
0065    (* ---------------------------------------------------------------------- *)
0066    (* CONTROLLO RICEZIONE CARATTERI                                          *)
0067    (* ---------------------------------------------------------------------- *)
0068    (* Ricevo caratteri da porta seriale e dal socket TCP. Se non ricevo più  *)
0069    (* caratteri per almeno un loop di programma invio i caratteri ricevuti   *)
0070    (* uno stream verso l'altro stream. Se caratteri ricevuti superano i 3/4  *)
0071    (* della dimensione del buffer di ricezione viene gestito l'invio.        *)
0072    (* ---------------------------------------------------------------------- *)
0073    (* Controllo se ricezione caratteri da porta seriale. *)
0074
0075    IF (((SysGetIChars(Sp.File) = IChars[0]) AND (IChars[0] > 0)) OR (IChars[0] > 192)) THEN
0076        RxChars[0]:=Sysfread(ADR(RxTxStr), 1, IChars[0], Sp.File); (* Received characters *)
0077        RxChars[0]:=Sysfwrite(ADR(RxTxStr), 1, RxChars[0], PSts.File); (* Transmitted characters *)
0078    END_IF;
0079
0080    (* Controllo se ricezione caratteri da socket TCP. *)
0081
0082    IF (((SysGetIChars(PSts.File) = IChars[1]) AND (IChars[1] > 0)) OR (IChars[1] > 192)) THEN
0083        RxChars[1]:=Sysfread(ADR(RxTxStr), 1, IChars[1], PSts.File); (* Received characters *)
```

```
0084         RxChars[1]:=Sysfwrite(ADR(RxTxStr), 1, RxChars[1], Sp.File); (* Transmitted characters *)
0085     END_IF;
0086
0087     (* Salvo numero caratteri in ricezione dai due streams. *)
0088
0089     IChars[0]:=SysGetIChars(Sp.File); (* Input characters *)
0090     IChars[1]:=SysGetIChars(PSts.File); (* Input characters *)
0091
0092 (* [End of file] *)
```