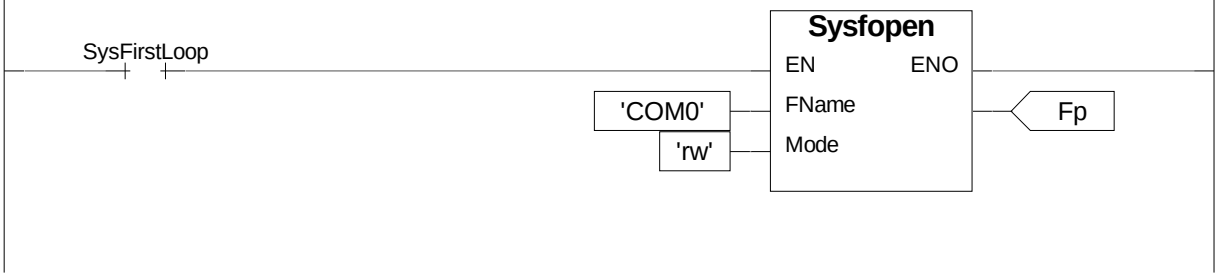
```
VAR_GLOBAL
Do00M00 AT %QX0.0 : BOOL; (* Output 00 module 00 *)
Do01M0 AT %QX0.1 : BOOL; (* Output 01 module 00 *)
END_VAR
```

PROGRAM Communication

```
VAR
Fp : FILEP; (* File pointer *)
Dummy : BOOL; (* Dummy variable *)
SMode : SetSMode; (* Set serial mode *)
DiSnd : DISend; (* DISend FB instance *)
DORcv : DOReceive; (* DOReceive FB instance *)
END_VAR
```

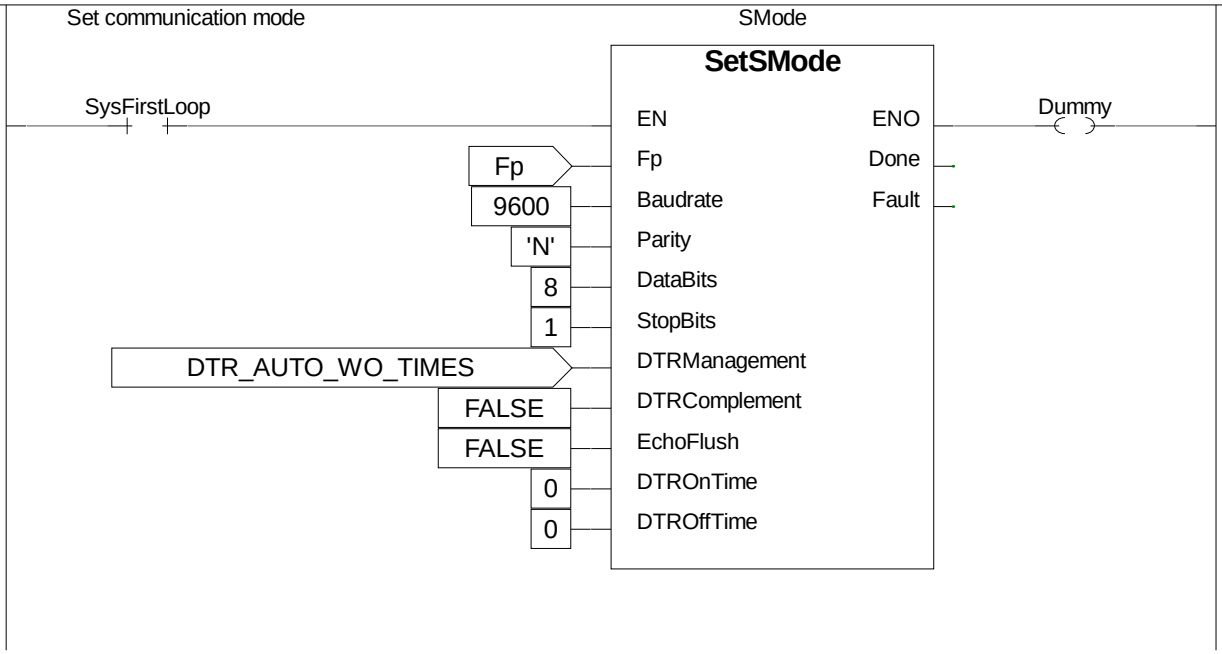
1

Open the serial COM

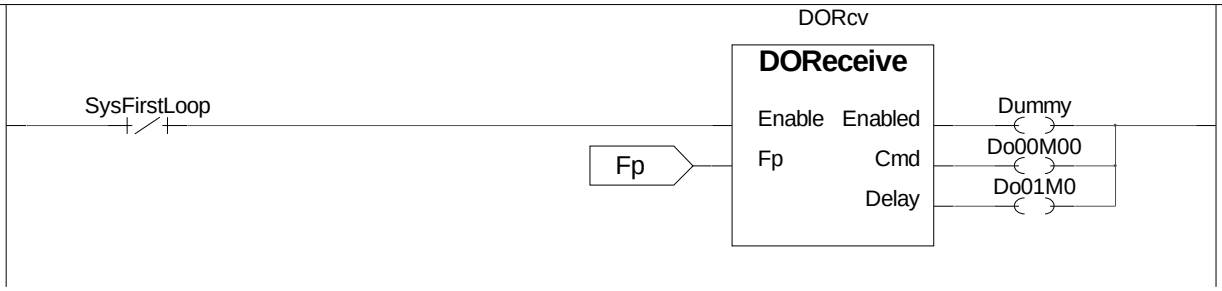


2

Set communication mode

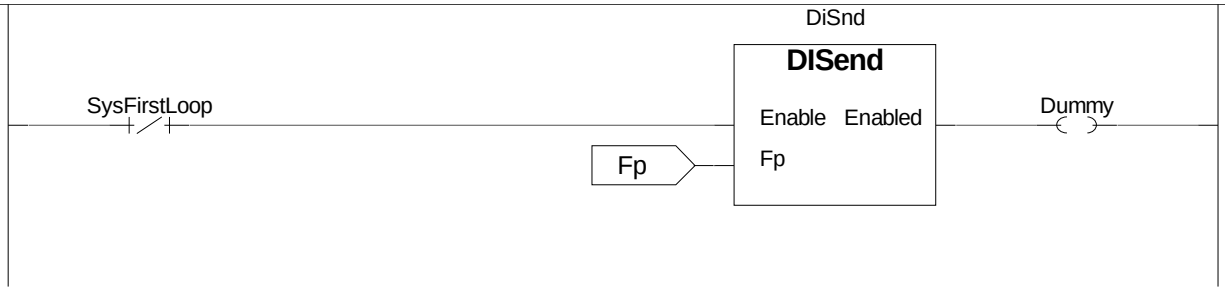


3



	Project : DIOManage	
	PROGRAM : Communication	
	Release : SendDIStat	Ver :1.00
	Author :	Date:29/09/2012
	Note :	Page:1 of 2

4



	Project : DIOManage	
	PROGRAM : Communication	
	Release : SendDIStat	Ver :1.00
	Author :	Date:29/09/2012
	Note :	Page:2 of 2

FUNCTION_BLOCK DISend

```
VAR_INPUT
Enable : BOOL; (* FB Enable *)
Fp : FILEP; (* File pointer *)
END_VAR

VAR_OUTPUT
Enabled : BOOL := FALSE; (* FB Enabled *)
END_VAR

VAR_EXTERNAL
SysModulesReady : WORD; (* SlimLine extension module ready *)
END_VAR

VAR
IStatus : ARRAY[ 0..2 ] OF UINT; (* Digital inputs status *)
i : USINT; (* Auxiliary variable *)
j : INT; (* Auxiliary variable *)
DIAcq : ARRAY[ 0..2 ] OF SysGetPhrDI; (* FB SysGetPhrDI instance *)
END_VAR
```

```
(* ********************************************************************** *)
(* FUNCTION BLOCK "DISend"                                                *)
(* ********************************************************************** *)
(* This FB sends a string on serial line on a change state of logic inputs.   *)
(* ---------------------------------------------------------------------- *)

    (* ---------------------------------------------------------------- *)
    (* FB ENABLE                                                        *)
    (* ---------------------------------------------------------------- *)
    (* Check if the FB is enabled. *)

    IF NOT(Enable) THEN
        Enabled:=FALSE; (* FB Enabled *)
        RETURN;
    END_IF;

    Enabled:=TRUE; (* FB Enabled *)

    (* Check if the output serial buffer is empty. *)

    IF (SysGetOSpace(Fp) <> TO_INT(SysGetTxBSize(Fp))) THEN RETURN; END_IF;

    (* ---------------------------------------------------------------- *)
    (* MANAGE DIGIT                                                     *)
    (* ---------------------------------------------------------------- *)
    (* Acquires the logic inputs from modules 0, 1, 2. To avoid errors the *)
    (* acquisition is made only if the module is connected to the CPU. *)

    IF TO_BOOL(SysModulesReady AND 16#0001) THEN DIAcq[0](Address:=0, Mode:=DI_16_L); END_IF;
    IF TO_BOOL(SysModulesReady AND 16#0002) THEN DIAcq[1](Address:=1, Mode:=DI_16_L); END_IF;
    IF TO_BOOL(SysModulesReady AND 16#0004) THEN DIAcq[2](Address:=2, Mode:=DI_16_L); END_IF;

    (* Compare the input status. *)

    IF ((IStatus[0] <> DIAcq[0].Value) OR
        (IStatus[1] <> DIAcq[1].Value) OR
        (IStatus[2] <> DIAcq[2].Value)) THEN
```

```
38
39          (* The inputs are changed copy their status. *)
40
41          IStatus[0]:=TO_UINT(DIAcq[0].Value); (* Digital inputs status *)
42          IStatus[1]:=TO_UINT(DIAcq[1].Value); (* Digital inputs status *)
43          IStatus[2]:=TO_UINT(DIAcq[2].Value); (* Digital inputs status *)
44
45          (* Send the string. *)
46
47          j:=Sysfputc(TO_INT(16#2A), Fp); (* Send the "*" character *)
48          FOR i:=0 TO 15 DO j:=Sysfputc(TO_INT(16#30+TO_USINT(VBitTest(IStatus[0], i))), Fp); END_FOR;
49          FOR i:=0 TO 15 DO j:=Sysfputc(TO_INT(16#30+TO_USINT(VBitTest(IStatus[1], i))), Fp); END_FOR;
50          FOR i:=0 TO 15 DO j:=Sysfputc(TO_INT(16#30+TO_USINT(VBitTest(IStatus[2], i))), Fp); END_FOR;
51          j:=Sysfputc(TO_INT(16#23), Fp); (* Send the "#" character *)
52      END_IF;
53
54 (* [End of file] *)
55
56
```

FUNCTION_BLOCK SetSMode

(SFR054B000) Set the serial communication mode
ENCRYPTED CODE

```
VAR_INPUT
Fp : FILEP;
Baudrate : UDINT; (* Baudrate *)
Parity : STRING[ 1 ]; (* Parity type *)
DataBits : USINT; (* Nr of data bits *)
StopBits : USINT; (* Nr of stop bits *)
DTRManagement : USINT; (* DTR management type *)
DTRComplement : BOOL; (* Complement the DTR signal *)
EchoFlush : BOOL; (* Flush the echo *)
DTROnTime : UINT; (* DTR On wait time *)
DTROffTime : UINT; (* DTR Off wait time *)
END_VAR

VAR_OUTPUT
Done : BOOL; (* Execution done *)
Fault : BOOL; (* Execution fault *)
END_VAR
```

1

	Project : DIOManage	
	FUNCTION BLOCK : SetSMode	
	Release : SendDIStat	Ver :1.00
	Author :	Date:29/09/2012
	Note :	Page:1 of 1

FUNCTION_BLOCK DOReceive

```
VAR_INPUT
Enable : BOOL; (* FB Enable *)
Fp : FILEP; (* File pointer *)
END_VAR

VAR_OUTPUT
Enabled : BOOL := FALSE; (* FB Enabled *)
Cmd : BOOL; (* Output command *)
Delay : BOOL; (* Output delayed *)
END_VAR

VAR
TimeBf : ARRAY[ 0..1 ] OF UDINT; (* Time buffer (uS) *)
RxIDx : USINT; (* Receive data index *)
RxData : ARRAY[ 0..15 ] OF USINT; (* Rx data buffer *)
END_VAR
```

```
(* ********************************************************************** *)
(* FUNCTION BLOCK "DOReceive"                                             *)
(* ********************************************************************** *)
(* This FB receives string from serial line to change state of logic outputs. *)
(*                                                                        *)
(* By receiving 1111 (four consecutive ‘1’) from RS232 as a string, digital   *)
(* output 00 is activated and at the same time digital out 01 is activated    *)
(* for 5 sec.                                                             *)
(* By receiving 0000 to RS232 as a string, digital output 00 is immediately   *)
(* deactivated                                                            *)
(* ---------------------------------------------------------------------- *)

    (* ---------------------------------------------------------------------- *)
    (* FB ENABLE                                                              *)
    (* ---------------------------------------------------------------------- *)
    (* Check if the FB is enabled. *)

    IF NOT(Enable) THEN
        RxIDx:=0; (* Receive data index *)
        Cmd:=FALSE; (* Output command *)
        Delay:=FALSE; (* Output delayed *)
        Enabled:=FALSE; (* FB Enabled *)
        RETURN;
    END_IF;

    Enabled:=TRUE; (* FB Enabled *)

    (* Manage the delayed output time. *)

    IF NOT(Delay) THEN
        TimeBf[0]:=SysGetSysTime(TRUE); (* Time buffer (uS) *)
    ELSIF ((SysGetSysTime(TRUE)-TimeBf[0]) >= 5000000) THEN
        Delay:=FALSE; (* Output delayed *)
    END_IF;

    (* ---------------------------------------------------------------------- *)
    (* RECEIVE THE SERIAL COMMAND                                             *)
    (* ---------------------------------------------------------------------- *)
    (* Check if there are any character in the receive buffer. *)

```

```
41      IF NOT(TO_BOOL(SysGetIChars(Fp))) THEN
42
43          (* If no characters are received for a 100 mS reception restarts. *)
44
45          IF ((SysGetSysTime(TRUE)-TimeBf[1]) >= 100000) THEN
46              RxIDx:=0; (* Receive data index *)
47          END_IF;
48
49          RETURN;
50      END_IF;
51
52      (* One or more characters are received store received character. *)
53
54      TimeBf[1]:=SysGetSysTime(TRUE); (* Time buffer (uS) *)
55      RxData[RxIDx]:=TO_USINT(Sysfgetc(Fp));
56
57      (* Check if received all the 4 characters. *)
58
59      RxIDx:=RxIDx+1; (* Receive data index *)
60      IF (RxIDx < 4) THEN RETURN; END_IF;
61      RxIDx:=0; (* Receive data index *)
62
63      (* Test the "1111" command. *)
64
65      IF ((RxData[0] = 16#31) AND (RxData[1] = 16#31) AND (RxData[2] = 16#31) AND (RxData[3] = 16#31)) TH
    EN
66          Cmd:=TRUE; (* Output command *)
67          Delay:=TRUE; (* Output delayed *)
68          RETURN;
69      END_IF;
70
71      (* Test the "0000" command. *)
72
73      IF ((RxData[0] = 16#30) AND (RxData[1] = 16#30) AND (RxData[2] = 16#30) AND (RxData[3] = 16#30)) TH
    EN
74          Cmd:=FALSE; (* Output command *)
75          RETURN;
76      END_IF;
77
78  (* [End of file] *)
79
80
```

FUNCTION VBitTest

(SFR058A100) Tests a bit of a variable
ENCRYPTED CODE

```
VAR_INPUT
Variable : UDINT; (* Variable to test *)
Bit : USINT; (* Bit to test *)
END_VAR
```

1

	Project : DIOManage	
	FUNCTION : VBitTest	
	Release : SendDIStat	Ver :1.00
	Author :	Date:29/09/2012
	Note :	Page:1 of 1