```
PROGRAM Modbus1
VAR
    MB_Inputs : ARRAY [ 0 .. 99 ] OF WORD ;
    MB_Outputs : ARRAY [ 0 .. 99 ] OF WORD ;
    Huzzah_Read : BOOL ;
    Huzzah_Write : BOOL ;
    Modbus_Scan_Delay : Ton ;
    MB_Delay_Enable : BOOL ;
    MB_Delay_time : TIME := t#200ms ;
    MD_Delay_State : INT ;
    Huzzah_LED : BOOL ;
    Poll_Rep_timer : Ton ;
    MB_Poll_Rep : BOOL ;
    MB_Poll_rep_time : TIME := t#20ms ;
END_VAR
```

```
//Modbus Inputs
MB_Inputs [ 0 ] := IoConfig_Globals_Mapping . MBSlave_IN [ 0 ] ;
MB_Inputs [ 1 ] := IoConfig_Globals_Mapping . MBSlave_IN [ 1 ] ;
MB_Inputs [ 2 ] := IoConfig_Globals_Mapping . MBSlave_IN [ 2 ] ;
MB_Inputs [ 3 ] := IoConfig_Globals_Mapping . MBSlave_IN [ 3 ] ;

//Modbus Outputs
IoConfig_Globals_Mapping . MBSlave_OUT [ 0 ] := MB_Outputs [ 0 ] ;
IoConfig_Globals_Mapping . MBSlave_OUT [ 1 ] := MB_Outputs [ 1 ] ;
IoConfig_Globals_Mapping . MBSlave_OUT [ 2 ] := MB_Outputs [ 2 ] ;
IoConfig_Globals_Mapping . MBSlave_OUT [ 3 ] := MB_Outputs [ 3 ] ;

//Get GPS Data from inputs
GPS_Lat_LSW := MB_Inputs [ 1 ] ;
GPS_Lat_MSW := MB_Inputs [ 0 ] ;
GPS_Lon_LSW := MB_Inputs [ 3 ] ;
GPS_Lon_MSW := MB_Inputs [ 2 ] ;

//Modbus slave Inputs
MB_Inputs [ 10 ] := IoConfig_Globals_Mapping . Huzzah_Inputs [ 0 ] ;
MB_Inputs [ 11 ] := IoConfig_Globals_Mapping . Huzzah_Inputs [ 1 ] ;
MB_Inputs [ 12 ] := IoConfig_Globals_Mapping . Huzzah_Inputs [ 2 ] ;
MB_Inputs [ 13 ] := IoConfig_Globals_Mapping . Huzzah_Inputs [ 3 ] ;
MB_Inputs [ 14 ] := IoConfig_Globals_Mapping . Huzzah_Inputs [ 4 ] ;
MB_Inputs [ 15 ] := IoConfig_Globals_Mapping . Huzzah_Inputs [ 5 ] ;
MB_Inputs [ 16 ] := IoConfig_Globals_Mapping . Huzzah_Inputs [ 6 ] ;

//Modbus Outputs
IoConfig_Globals_Mapping . Huzzah_Outputs [ 0 ] := MB_Outputs [ 10 ] ;
IoConfig_Globals_Mapping . Huzzah_Outputs [ 1 ] := MB_Outputs [ 11 ] ;
IoConfig_Globals_Mapping . Huzzah_Outputs [ 2 ] := MB_Outputs [ 12 ] ;
IoConfig_Globals_Mapping . Huzzah_Outputs [ 3 ] := MB_Outputs [ 13 ] ;

IoConfig_Globals_Mapping . Huzzah_Read_Trigger := Huzzah_Read ;
```

```
35      IoConfig_Globals_Mapping . Huzzah_Write_Trigger := Huzzah_Write ;
36
37      //send joystick value back to huzzah for use as pwm speed
38      MB_Outputs [ 10 ] := MB_Inputs [ 11 ] ;
39      MB_Outputs [ 11 ] . 0 := NOT  Huzzah_LED ;
40
41      //Read first, then write. Arduino library on supports one type at a time in a
        single frame
42      Modbus_Scan_Delay ( IN := MB_Delay_Enable ,  PT := MB_Delay_time ,  Q => ,  ET => ) ;
43      Poll_Rep_timer ( IN := MB_Poll_Rep ,  PT := MB_Poll_rep_time ,  Q => ,  ET => ) ;
44
45      CASE  MD_Delay_State  OF
46
47          0 :
48              //Send the read request
49              Huzzah_Read := 1 ;
50              MB_Delay_Enable := 1 ;
51
52              //Keep triggering the poll while scan in on
53              MB_Poll_Rep := 1 ;
54              IF  Poll_Rep_Timer . Q  THEN
55                  Huzzah_Read := 0 ;
56                  MB_Poll_Rep := 0 ;
57              END_IF
58
59              //After Scan delay
60              IF  Modbus_Scan_Delay . q  THEN
61                  Huzzah_Read := 0 ;
62                  MB_Delay_Enable := 0 ;
63                  MD_Delay_State := 10 ;
64              END_IF
65
66          10 :
67
68              //Send the write request
69              Huzzah_Write := 1 ;
70              MB_Delay_Enable := 1 ;
71
72              //Keep triggering the poll while scan in on
73              MB_Poll_Rep := 1 ;
74              IF  Poll_Rep_Timer . Q  THEN
75                  Huzzah_Write := 0 ;
76                  MB_Poll_Rep := 0 ;
77              END_IF
78
79              //After delay
80              IF  Modbus_Scan_Delay . q  THEN
81                  Huzzah_Write := 0 ;
82                  MB_Delay_Enable := 0 ;
83                  MD_Delay_State := 0 ;
84              END_IF
```

```
85
86      END_CASE
87
```