Telephone: 01480 395256
www.i4automation.co.uk
sales@i4automation.co.uk

Application Note – Dual screens with Vision OPLC's

Dual Screens

There are many applications that call for two PLC/HMI screens to be showing the same information and for control to be available from both screens, typically one screen on either side of a machine saving the operator from having to cross the production line in the course of his job. A specific function is available to network Vision OPLC's however this involves the necessity of fitting an Ethernet card along with the associated routers and cable length limitations. For applications with longer cable runs the same effect can be achieved using the RS485 serial ports.

This application note describes some of the considerations that need to be made in order to achieve a usable system. These considerations fall into three distinct areas:

Baud rate versus cable length

Number of registers and bits to be read and written

Limitations of some HMI functions

Baud rate versus cable length

RS485 is generally good for about 1000m and can be extended further if appropriate boosters are used, however, two Vision units can talk directly to each other over RS485 using Modbus as the communication protocol. The higher the baud rate used the shorter will be the transmission time and hence the communication poll time will reduce, the user should use the highest baud rate available consistent with the length of the cable run, a figure which is hard to calculate theoretically as much depends on the level of noise encountered.

Number of register and bits to be read and written

The second factor affecting communication poll time, and therefore the users perceived response, is the number of registers and bits to be read and written, the higher the number the slower the response. Try to limit the comms to reading one block of registers, writing one block of registers and then reading & writing one block of bits, even better pack the bits into registers.

Good response results have been achieved with a program that used the following blocks.

Read 1000 bits
Write 400 bits
Read 300 registers (in 3 blocks of 100)
Write 200 registers (in 2 blocks of 100)

At the end of development the number of registers read was reduced to 150 with 100 registers written to because the extra registers had not been used.

Registers were read always but only written to once, if there was a change in value, whereas coils were read and written all of the time. A Modbus routine is available for Vision products if required.

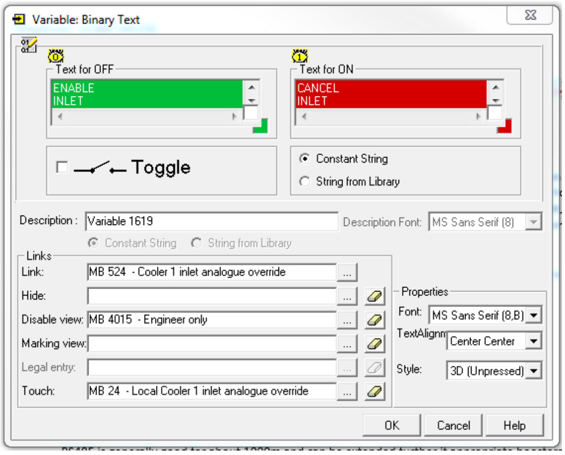
Limitations of some HMI functions

As only one of the PLC's will be attached to the real IO the control program appears only in the PLC with IO. Duplicate program exists in both PLC's to cover such things as alarms, navigation, passwords and house keeping functions. This minimises the number of registers and bits that need to be transmitted.

Both PLC's need nearly identical HMI screens with slight changing of addresses being the only difference.

The input of numeric values in either PLC can be made in the same manner as if only one PLC were used, once a value is changed that value is immediately transmitted and read instructions are suspended for a short period of time to allow the new value to register.

Bit values are generally treated the same as if only one PLC were present except where the 'toggle' function of an HMI object is used. The 'toggle' property of an HMI object such as a Binary Text Switch cannot be used, instead it is necessary to write a line of PLC code to parallel up the two switches and use the result to 'toggle' a bit. The 'link' and 'touch' property of the HMI object also need consideration, an example of which is shown below.



Variable: Binary Text

Text for OFF: ENABLE INLET

Text for ON: CANCEL INLET

□ Toggle

● Constant String ○ String from Library

Description : Variable 1619 — Description Font: MS Sans Serif (8)

● Constant String ○ String from Library

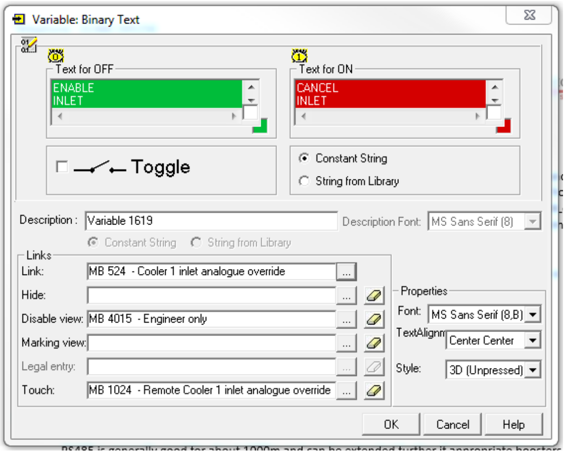
Links

Link: MB 524 - Cooler 1 inlet analogue override
Hide:
Disable view: MB 4015 - Engineer only
Marking view:
Legal entry:
Touch: MB 24 - Local Cooler 1 inlet analogue override

Properties

Font: MS Sans Serif (8,B)
TextAlignm: Center Center
Style: 3D (Unpressed)

PLC 1



Variable: Binary Text

Text for OFF: ENABLE INLET

Text for ON: CANCEL INLET

□ Toggle

● Constant String ○ String from Library

Description : Variable 1619 — Description Font: MS Sans Serif (8)

● Constant String ○ String from Library

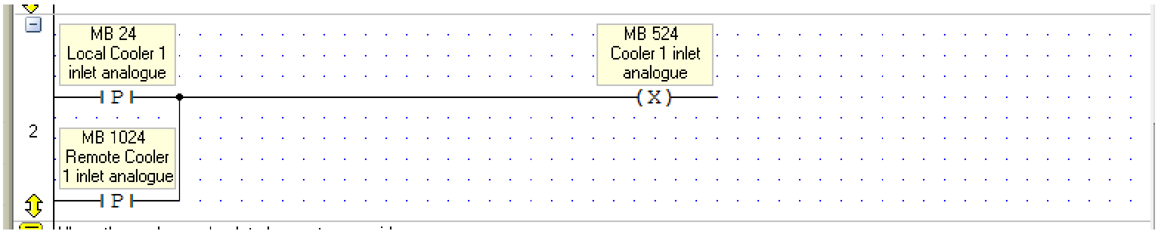
Links

Link: MB 524 - Cooler 1 inlet analogue override
Hide:
Disable view: MB 4015 - Engineer only
Marking view:
Legal entry:
Touch: MB 1024 - Remote Cooler 1 inlet analogue override

Properties

Font: MS Sans Serif (8,B)
TextAlignm: Center Center
Style: 3D (Unpressed)

PLC 2



Ladder code in PLC 1 to combine the 'toggle' function.

Note that both of the Binary Text boxes use the same 'link' address, which is the combined output, but have different 'touch' addresses. PLC 2 is the Modbus master and reads the link address while writing the touch address.

Memory map

PLC 1 – Slave		PLC 2 – Master	Modbus		
			Firing order	Type	Command
MB 0 – MB 999	→	MB 0 – MB 999	5	Polled	RC #1
MB 1000 – MB 1399	←	MB 1000 – MB 1399	1	Polled	FC #15
MI 0 – MI 99	→	MI 0 – MI 99	2	Polled	RHR #3
MI 1000 – MI 1099	→	MI 1000 – MI 1099	3	Polled	RHR #3
MI 1100 – MI 1199	→	MI 1100 – MI 1199	4	Polled	RHR #3
MI 1000 – MI 1099	←	MI 1000 – MI 1099	6	On demand	PHR #16
MI 1100 – MI 1199	←	MI 1100 – MI 1199	7	On Demand	PHR #16

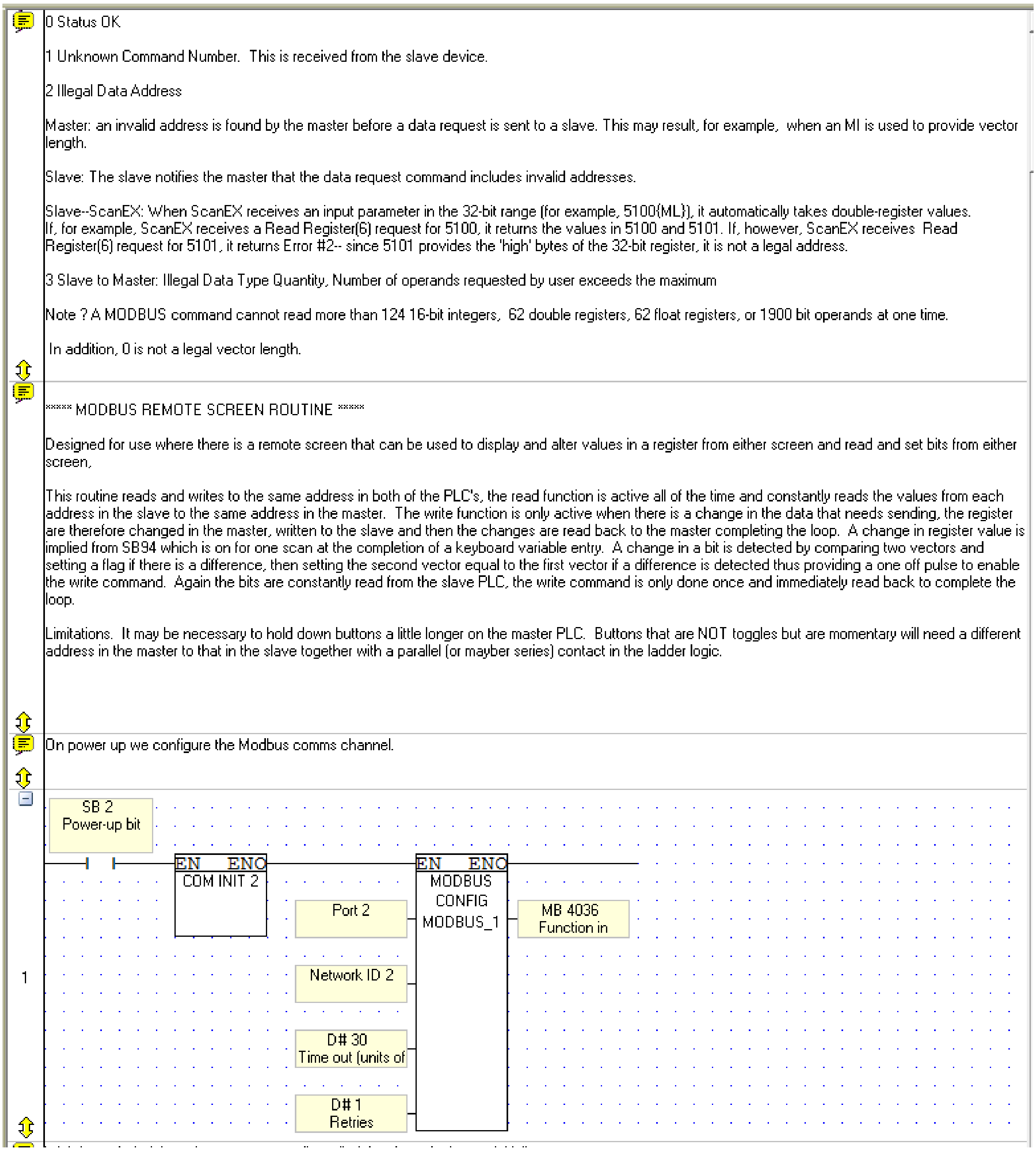
The table above shows the size and direction of the five polled Modbus commands and the two on demand commands. The command is the Modbus command number eg RHR #3 is Read Holding Registers (Command No 3.)

Master & Slave programs

The following pages contain the program for the master and slave PLC's that transfer the above registers and bits.

Tony Spearing
Technical Manager
February 2014

Master Program – Which can be used as a subroutine which is called all of the time.



0 Status OK

1 Unknown Command Number. This is received from the slave device.

2 Illegal Data Address

Master: an invalid address is found by the master before a data request is sent to a slave. This may result, for example, when an MI is used to provide vector length.

Slave: The slave notifies the master that the data request command includes invalid addresses.

Slave--ScanEX: When ScanEX receives an input parameter in the 32-bit range (for example, 5100(ML)), it automatically takes double-register values. If, for example, ScanEX receives a Read Register(6) request for 5100, it returns the values in 5100 and 5101. If, however, ScanEX receives Read Register(6) request for 5101, it returns Error #2-- since 5101 provides the 'high' bytes of the 32-bit register, it is not a legal address.

3 Slave to Master: Illegal Data Type Quantity, Number of operands requested by user exceeds the maximum

Note ? A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, 62 float registers, or 1900 bit operands at one time.

In addition, 0 is not a legal vector length.

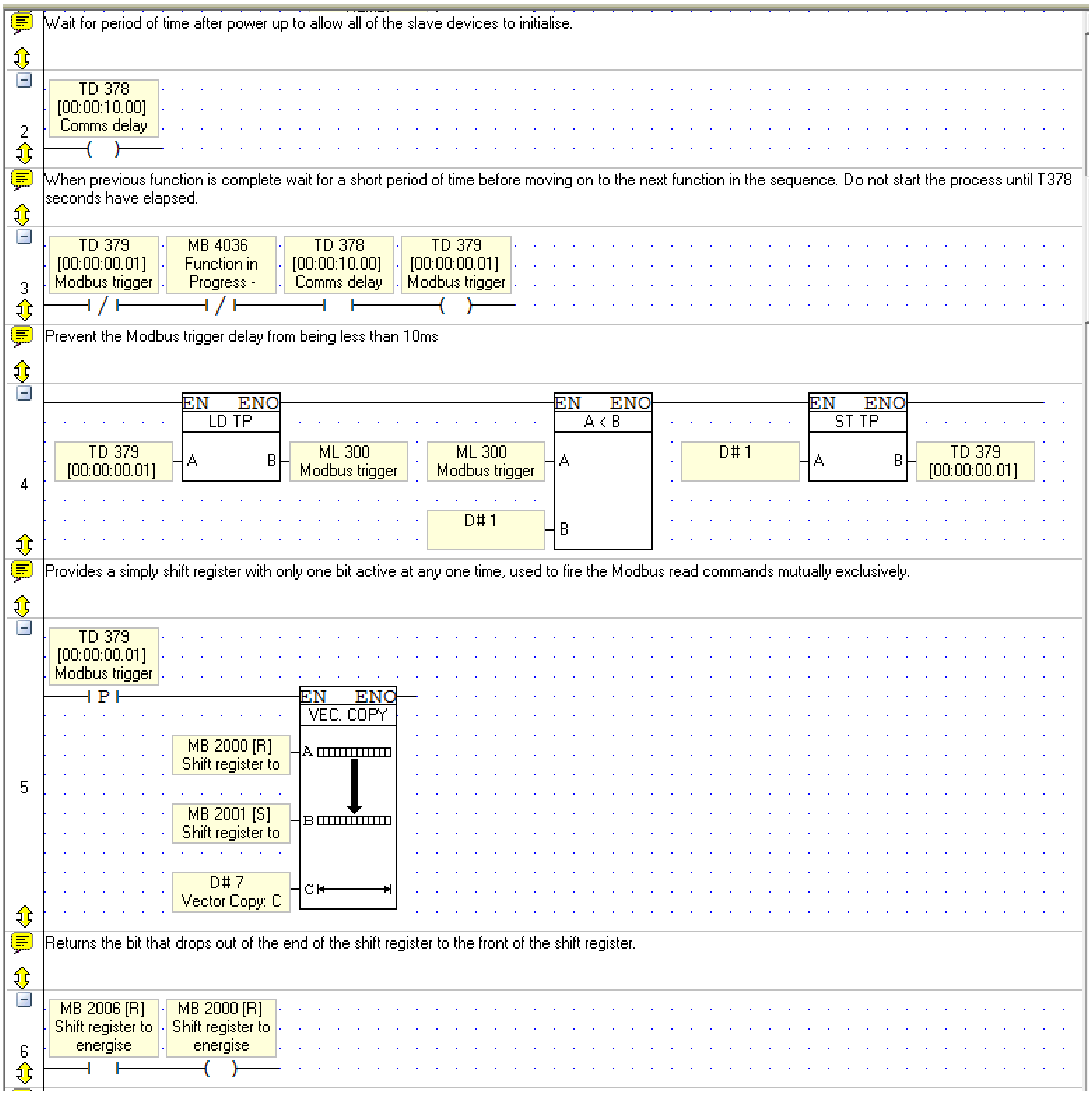
***** MODBUS REMOTE SCREEN ROUTINE *****

Designed for use where there is a remote screen that can be used to display and alter values in a register from either screen and read and set bits from either screen,

This routine reads and writes to the same address in both of the PLC's, the read function is active all of the time and constantly reads the values from each address in the slave to the same address in the master. The write function is only active when there is a change in the data that needs sending, the register are therefore changed in the master, written to the slave and then the changes are read back to the master completing the loop. A change in register value is implied from SB94 which is on for one scan at the completion of a keyboard variable entry. A change in a bit is detected by comparing two vectors and setting a flag if there is a difference, then setting the second vector equal to the first vector if a difference is detected thus providing a one off pulse to enable the write command. Again the bits are constantly read from the slave PLC, the write command is only done once and immediately read back to complete the loop.

Limitations. It may be necessary to hold down buttons a little longer on the master PLC. Buttons that are NOT toggles but are momentary will need a different address in the master to that in the slave together with a parallel (or mayber series) contact in the ladder logic.

On power up we configure the Modbus comms channel.



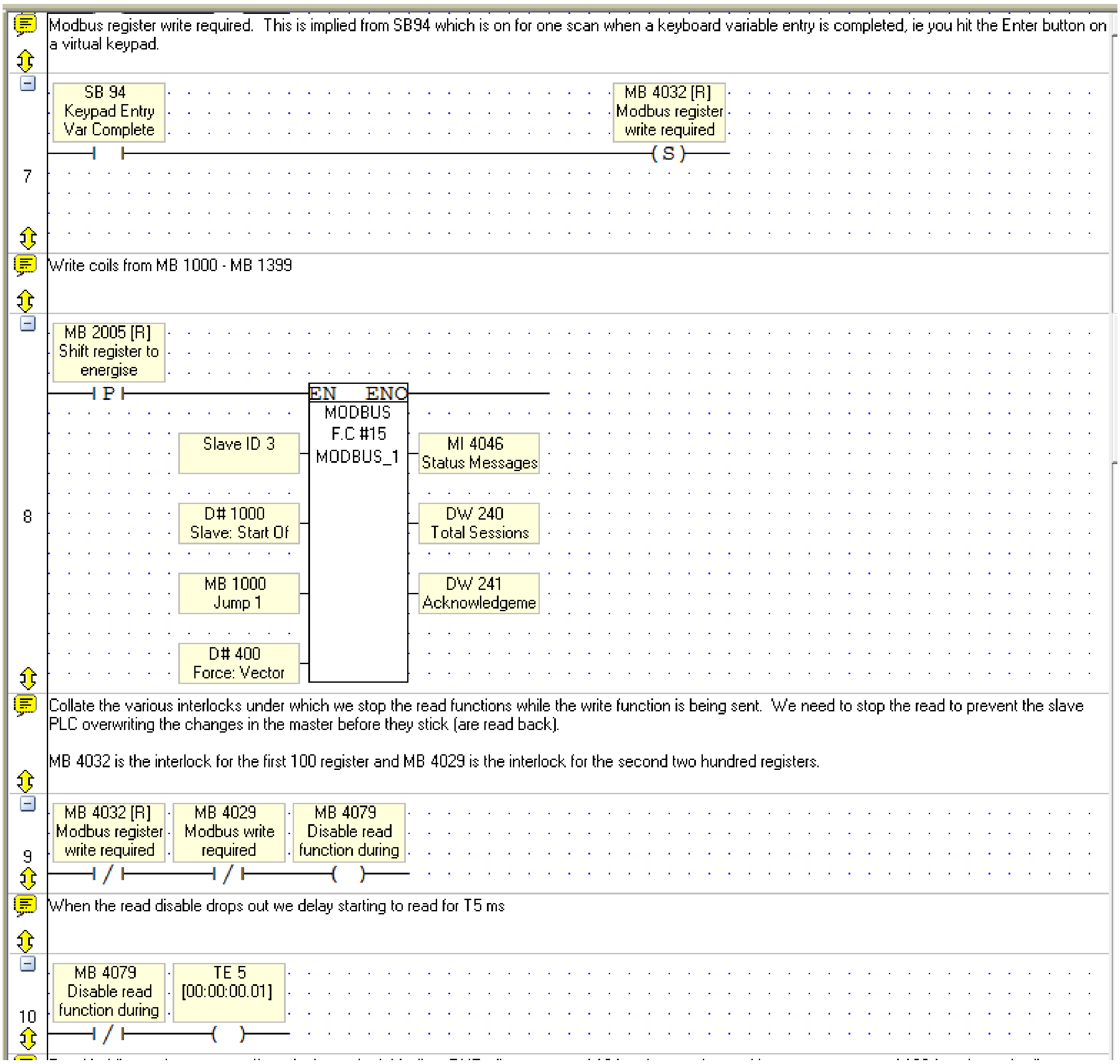
Wait for period of time after power up to allow all of the slave devices to initialise.

When previous function is complete wait for a short period of time before moving on to the next function in the sequence. Do not start the process until T378 seconds have elapsed.

Prevent the Modbus trigger delay from being less than 10ms

Provides a simply shift register with only one bit active at any one time, used to fire the Modbus read commands mutually exclusively.

Returns the bit that drops out of the end of the shift register to the front of the shift register.



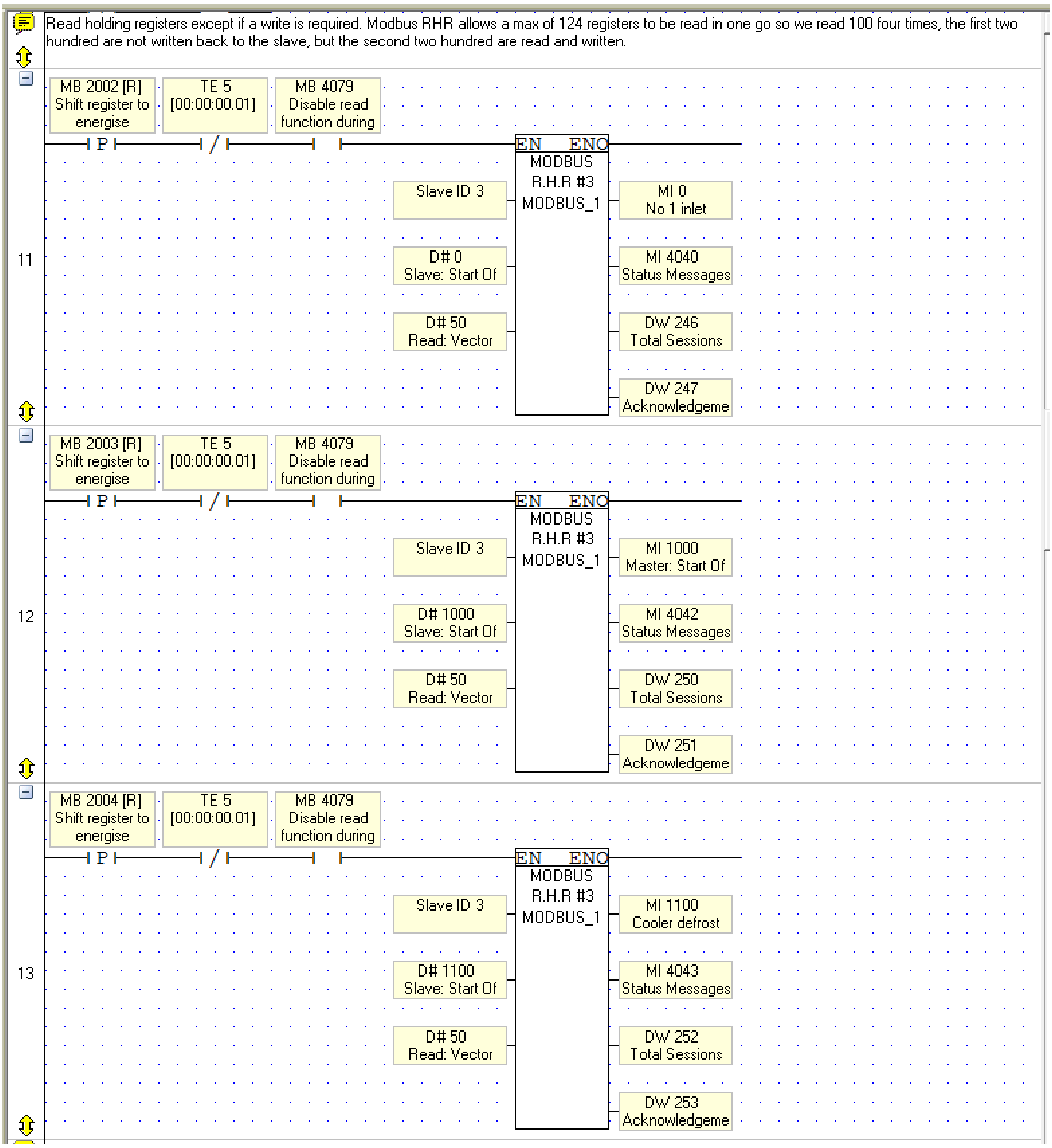
Modbus register write required. This is implied from SB94 which is on for one scan when a keyboard variable entry is completed, ie you hit the Enter button on a virtual keypad.

Write coils from MB 1000 - MB 1399

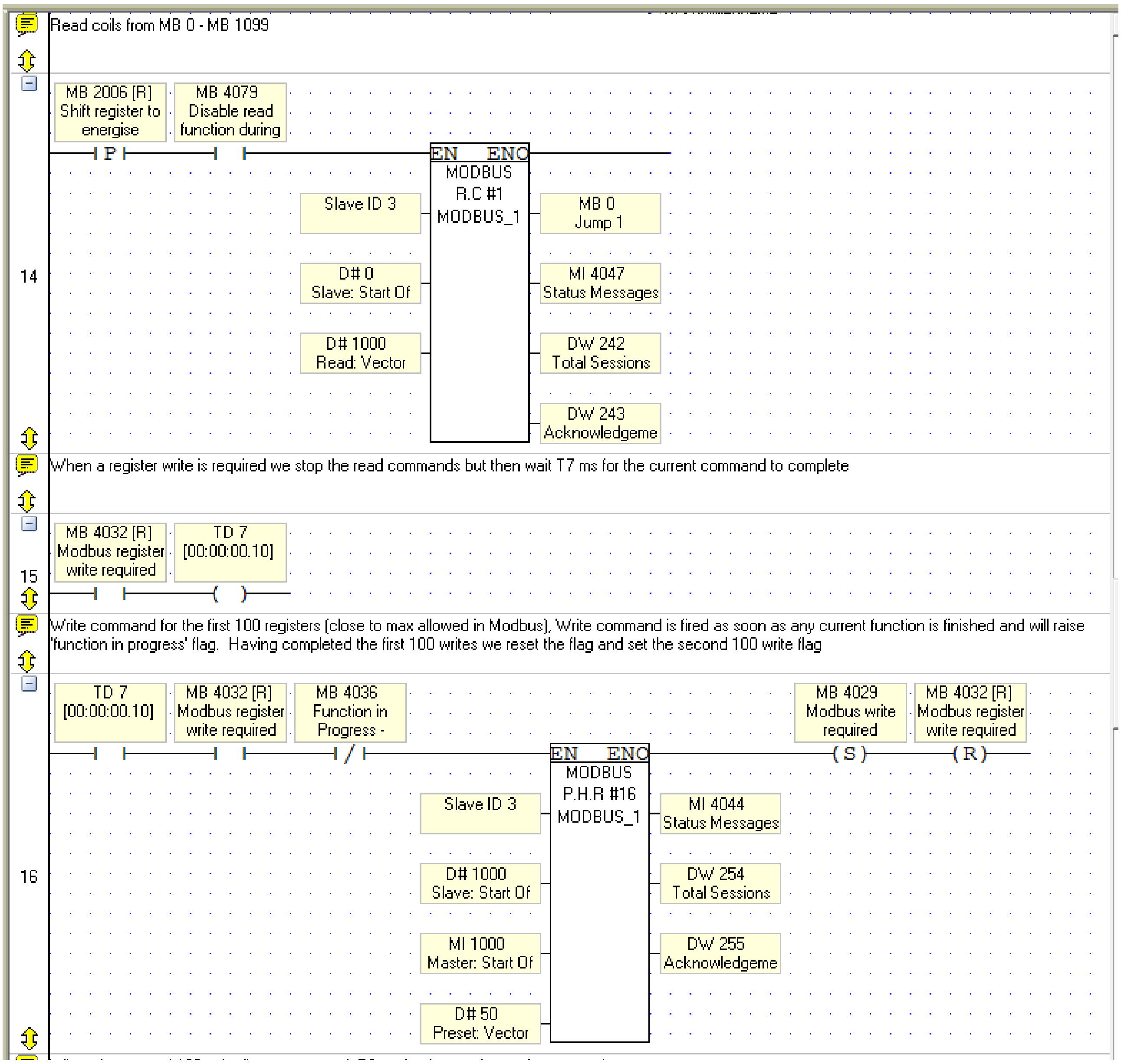
Collate the various interlocks under which we stop the read functions while the write function is being sent. We need to stop the read to prevent the slave PLC overwriting the changes in the master before they stick (are read back).

MB 4032 is the interlock for the first 100 register and MB 4029 is the interlock for the second two hundred registers.

When the read disable drops out we delay starting to read for T5 ms



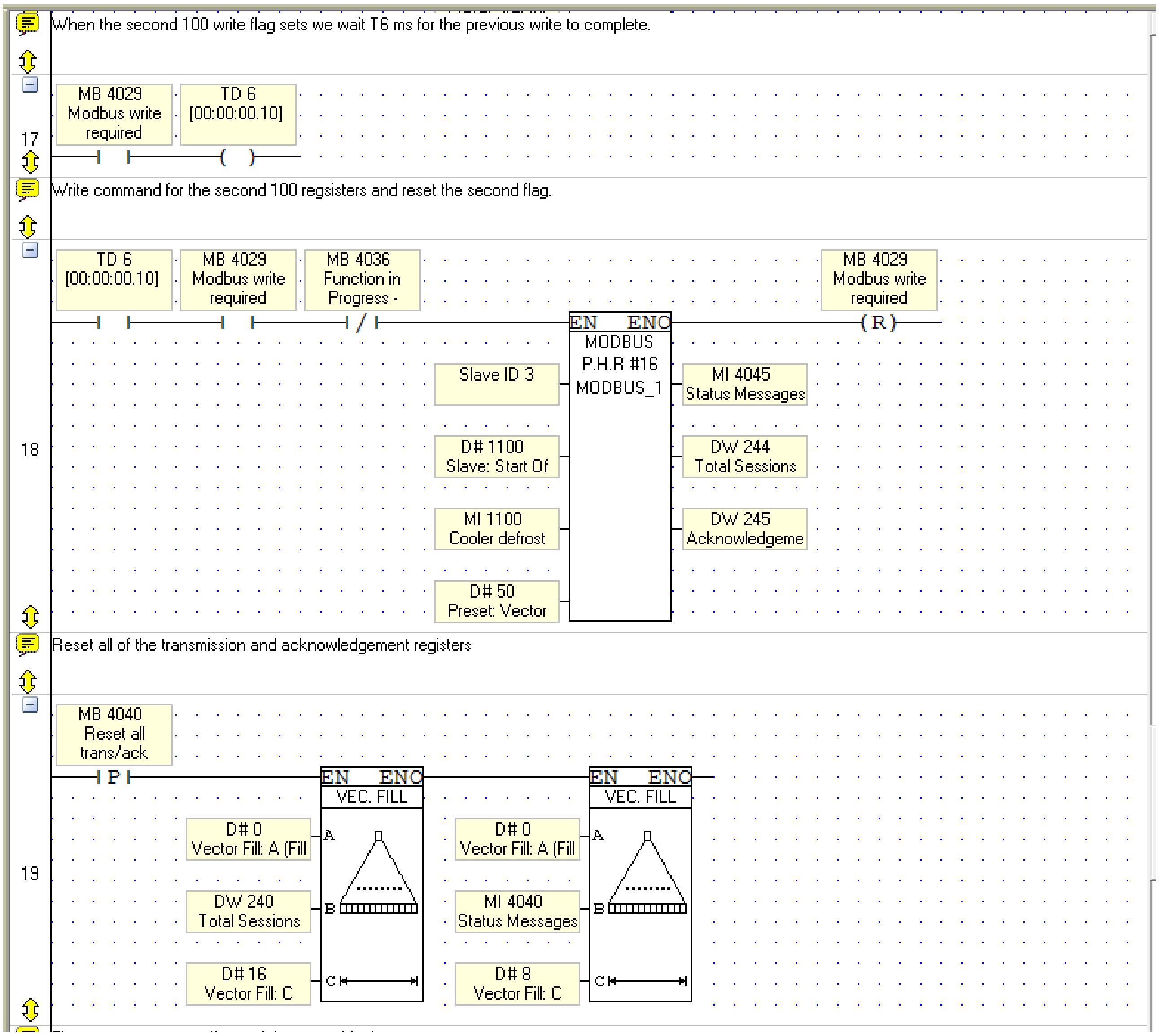
Read holding registers except if a write is required. Modbus RHR allows a max of 124 registers to be read in one go so we read 100 four times, the first two hundred are not written back to the slave, but the second two hundred are read and written.



Read coils from MB 0 - MB 1099

When a register write is required we stop the read commands but then wait T7 ms for the current command to complete

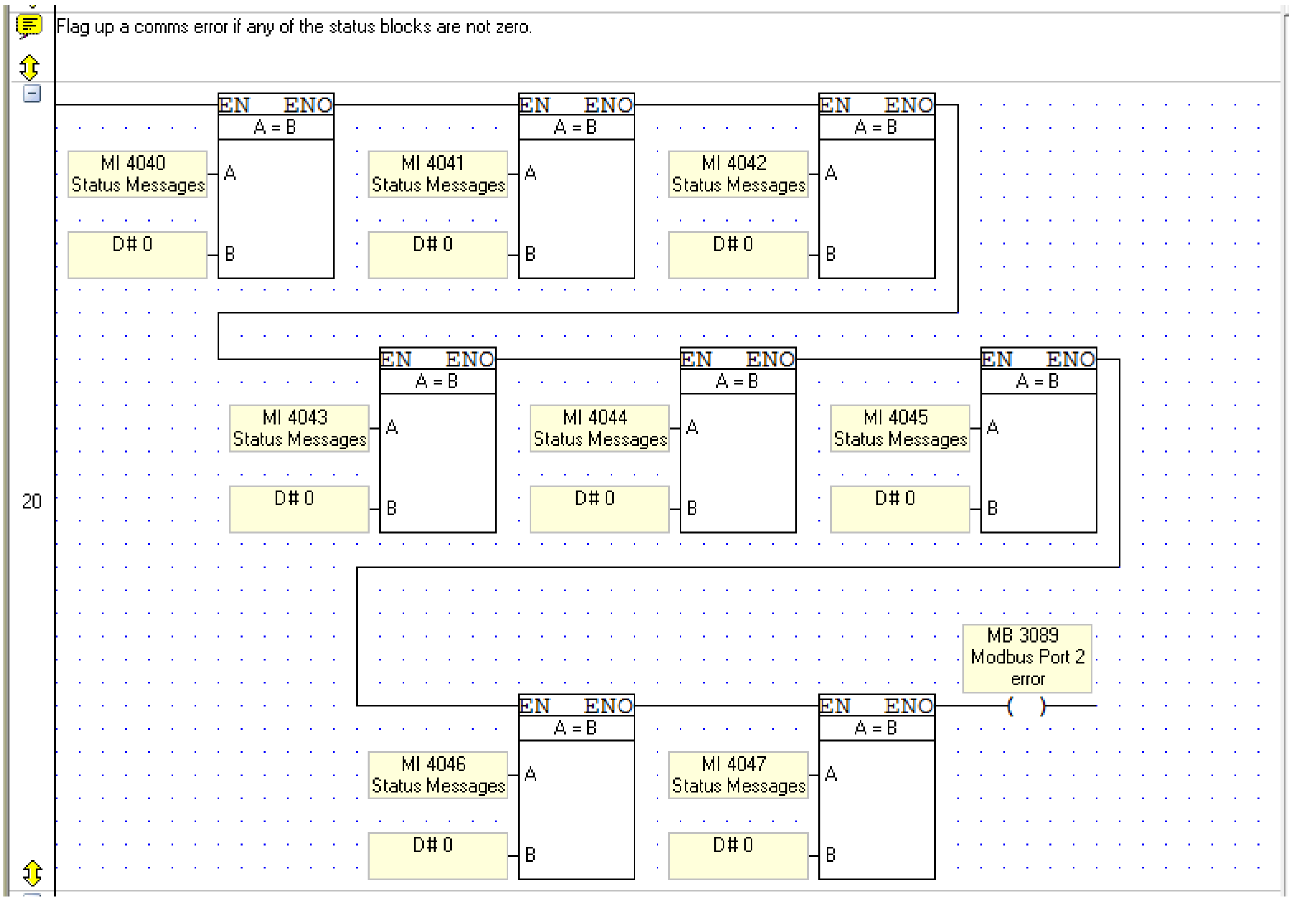
Write command for the first 100 registers (close to max allowed in Modbus). Write command is fired as soon as any current function is finished and will raise 'function in progress' flag. Having completed the first 100 writes we reset the flag and set the second 100 write flag



When the second 100 write flag sets we wait T6 ms for the previous write to complete.

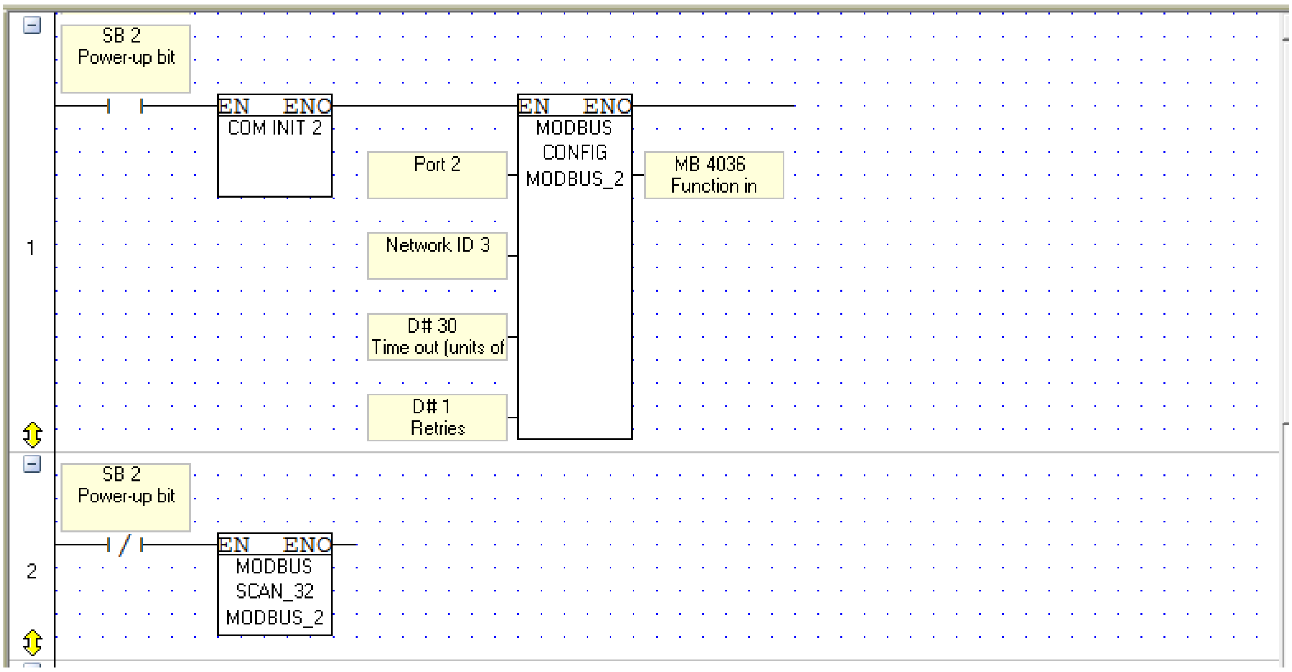
Write command for the second 100 regsisters and reset the second flag.

Reset all of the transmission and acknowledgement registers



Flag up a comms error if any of the status blocks are not zero.

Slave Program – The slave program is very short and can be a subroutine which is called all of the time.


1
SB 2
Power-up bit
EN ENO
COM INIT 2
EN ENO
MODBUS
CONFIG
MODBUS_2
Port 2
Network ID 3
D# 30
Time out (units of
D# 1
Retries
MB 4036
Function in
2
SB 2
Power-up bit
EN ENO
MODBUS
SCAN_32
MODBUS_2