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

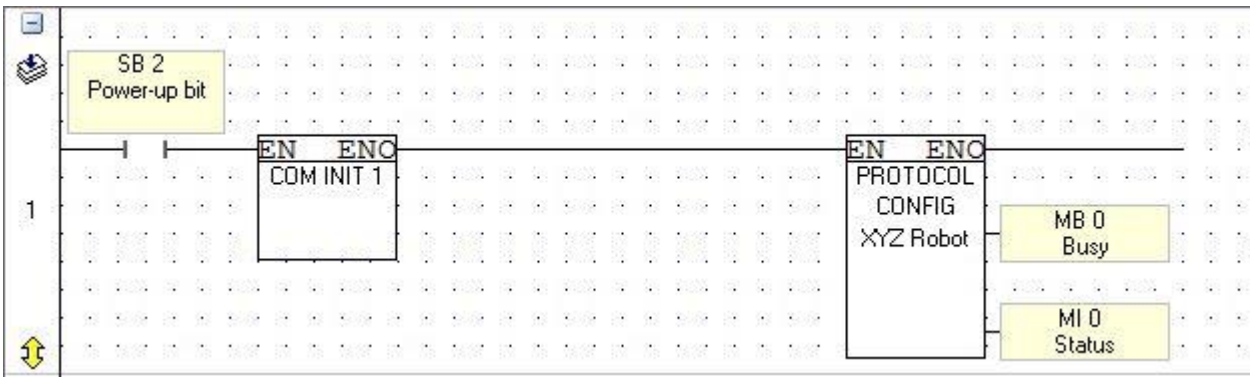
i4 Automation Ltd
Innovative Solutions for Industry

Serial Communications Made Easy

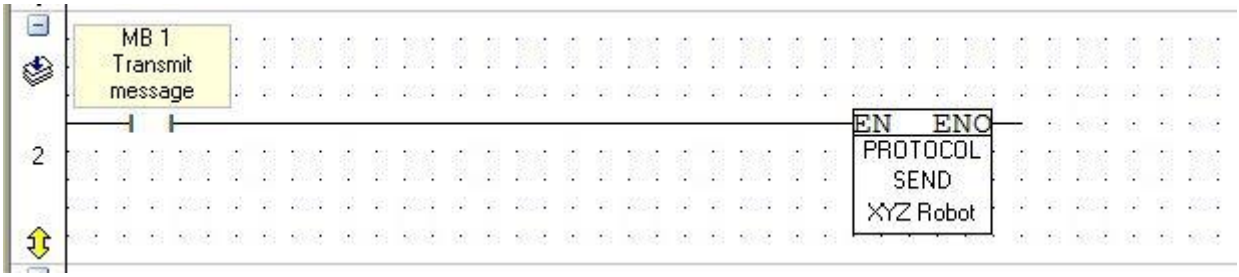
It is frequently convenient to have any two devices communicate via a serial link as opposed to connecting them together using thirty or upwards digital IO, along with all of the wiring grief such a solution creates. Invariably these devices do not 'speak' the same language so it is necessary for one of the devices to mimic the other language, which involves assembling long strings of data into registers, converting ASCII to binary, calculating checksums and finally transmitting the data in the vague hope there will be some response from the other device, all assuming you got the wiring correct in the first instance.

Unitronics engineers have worked hard to reduce the pain involved in serial communications by producing a function block that allows you to configure a message using a comprehensive set of pre-defined conversion and calculation tools. The net result is a neat little box in the middle of your program that sends out a question and returns the answer to a place of your choice.

In it's most simple form the program consists of three lines of logic, the first line simply initialising the COM port and defining the protocol name while providing a 'Busy' signal and a status register for error codes. Here we are talking to an XYZ robot.

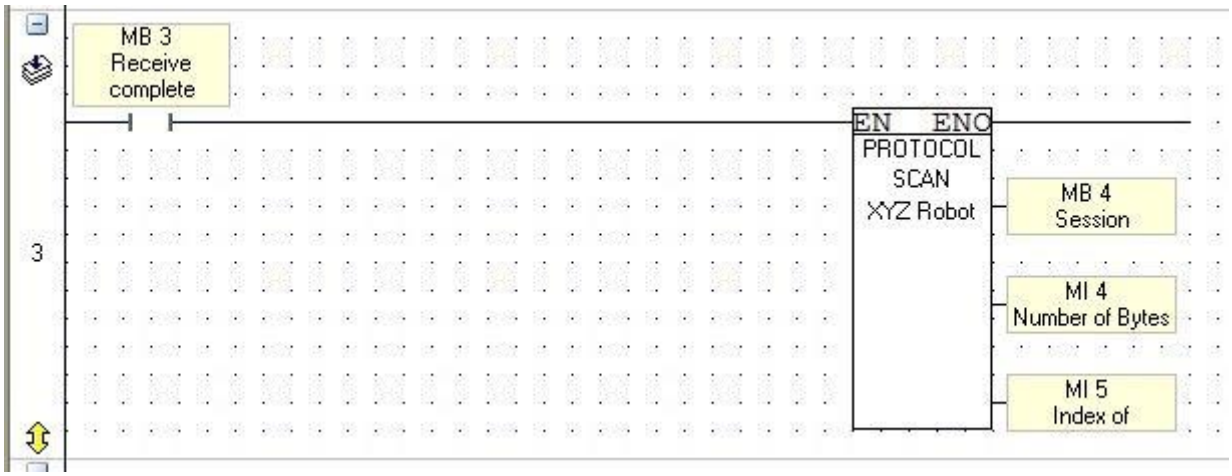


Having set up the COM port we then transmit the message that has been pre-assembled by the SEND function block.



The PLC then monitors the COM port for a reply from the XYX robot and disassembles the message to place only the required data where you want it.

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



The really NEAT bit of this message exchange is what happens inside the PROTOCOL SEND & PROTOCOL SCAN function blocks. This is where the message strings are assembled and disassembled with ease because we use the pre-defined conversion and calculation tools.
What does a message string normally consist of?

Start transmission | Fixed command | Variable data | Checksum |End transmission

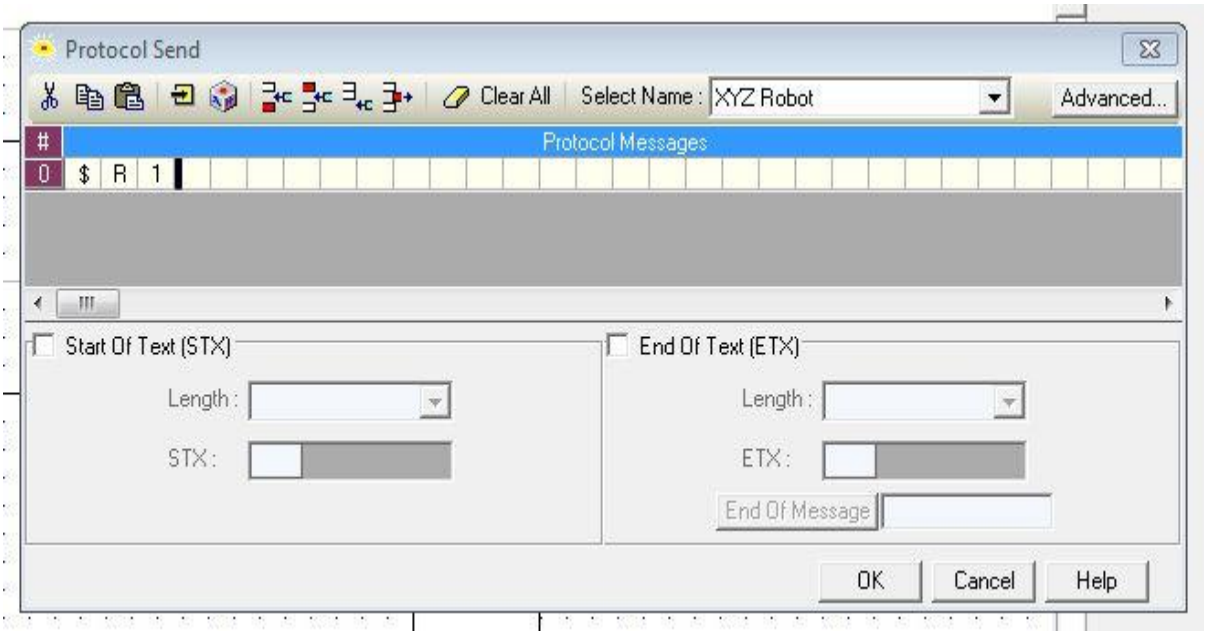
Take the command string needed to change the program number on a Janome robot, it is shown below.

e.g. To Change the Program Number to 6

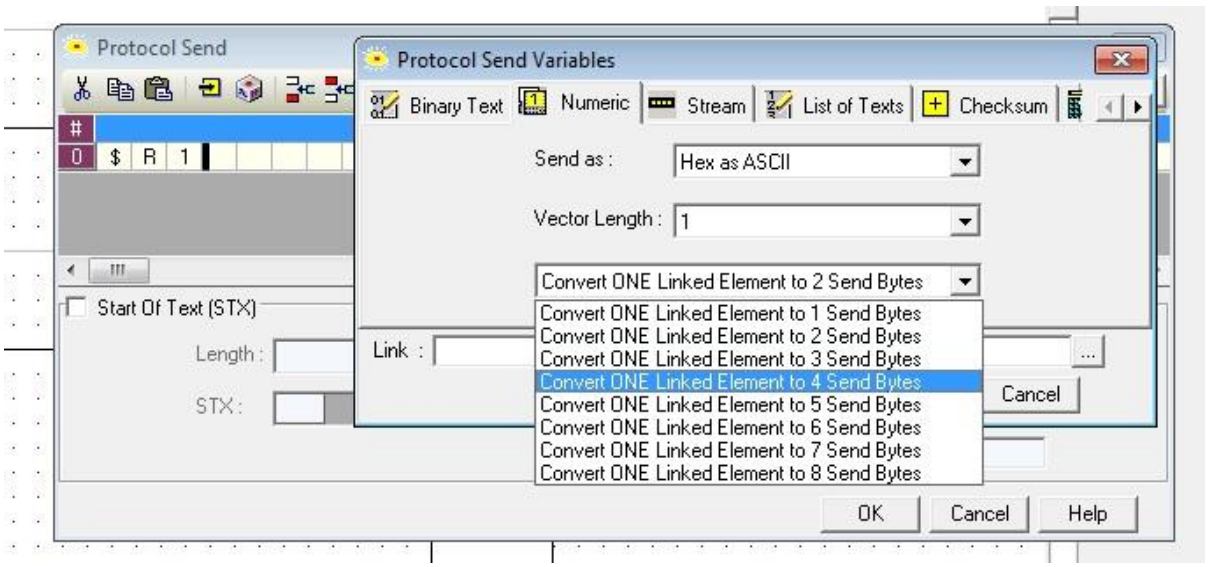
N	HEX	ASC	Description
1	24	$	$: Transmission Start Code
2	52	R	Command Code
3	31	1	Subcommand Code (1: Program Number Change)
4	30	0	Program Number 76 = 004CH
5	30	0	
6	34	4	
7	43	C	
8	35	5	SUM 52H+31H+30H+30H+34H+43H=15AH
9	41	A	
10	0D		CR: Transmission End Code

The first three characters consist of the start code followed by a two character command code $R1. Double click on the PROTOCOL SEND box to open the message definition dialog box and then enter these as fixed values into the message string as shown.

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

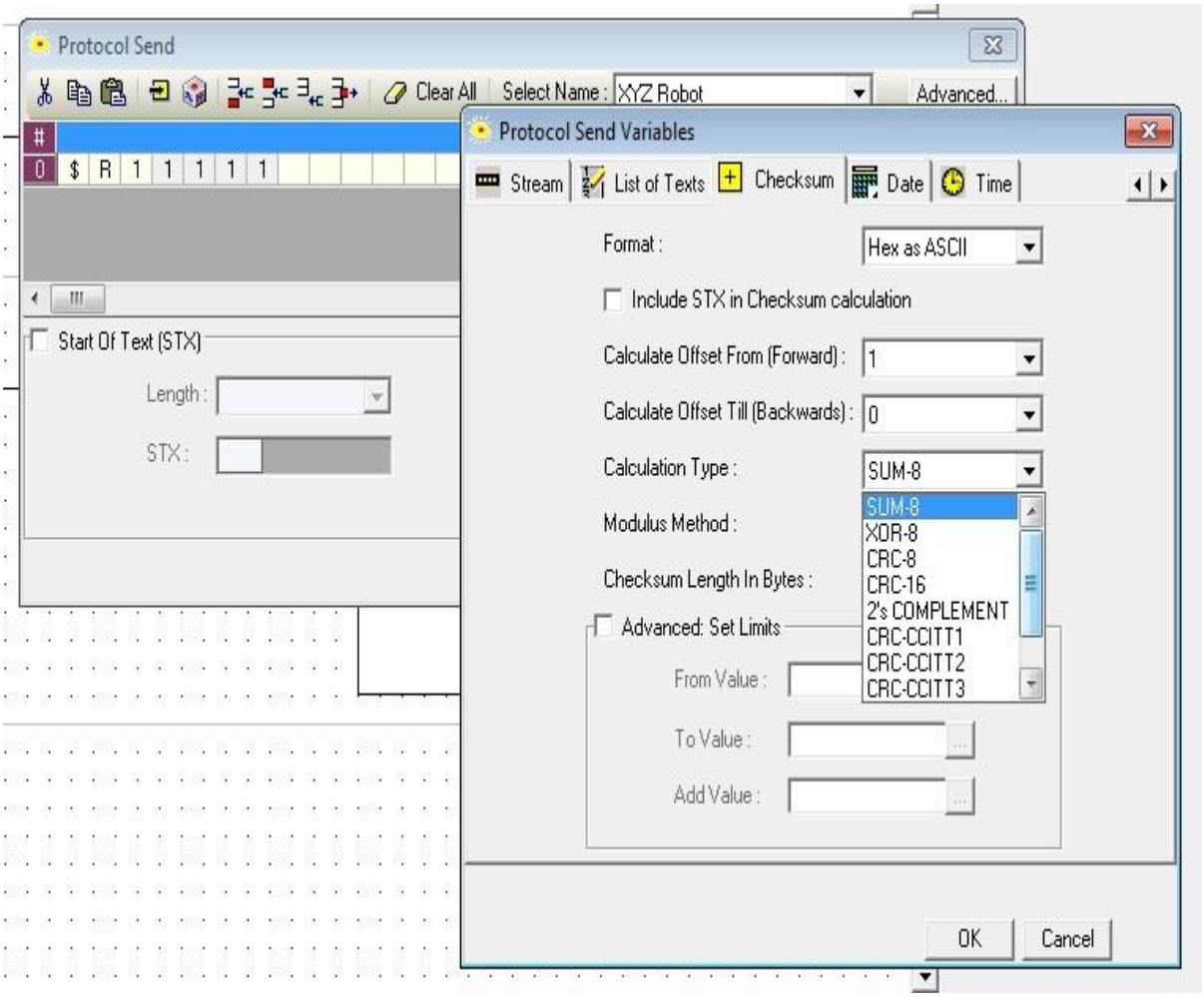


The next four characters represent the program number but the operator will only type 76 into the program number register, so we now have to convert one register into four characters, how? Create/Edit Variable and select it from the drop down box!



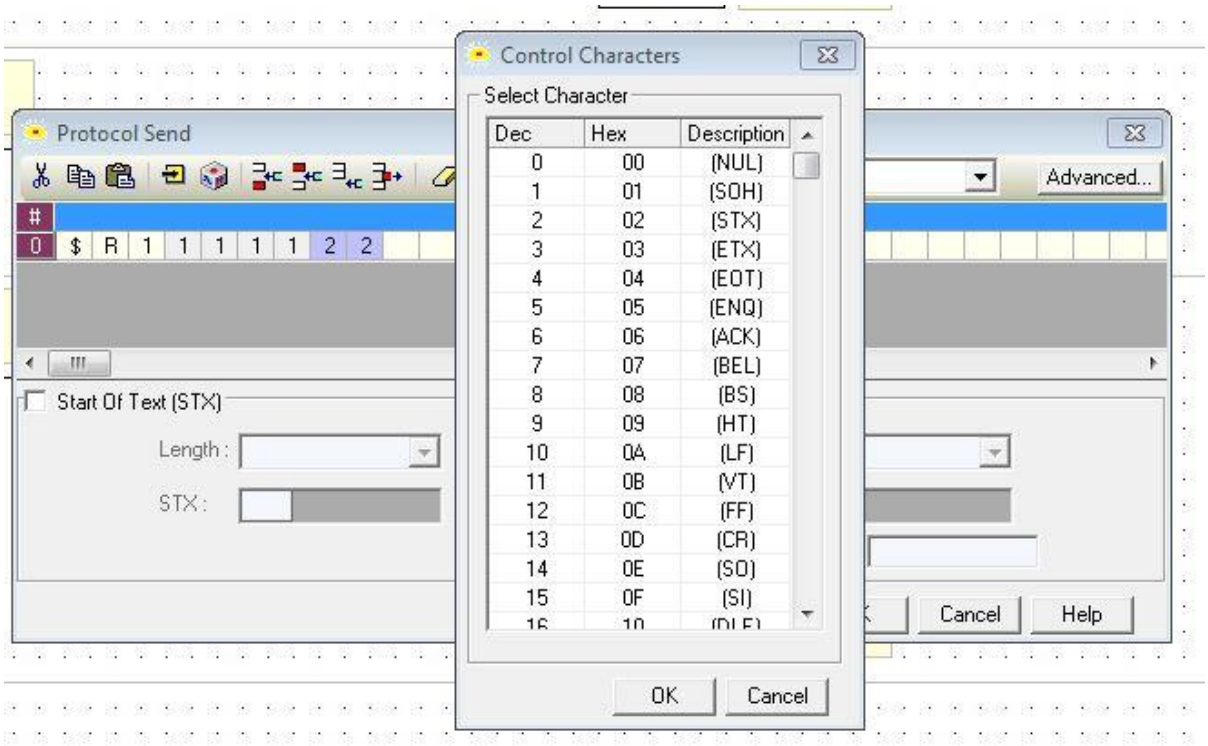
Then all that is left is to calculate the checksum and add in the end of transmission character, there are a number of algorithms for calculating the checksum (as shown) which in this case is a simple SUM, which bytes are summed being specified by the offsets.

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



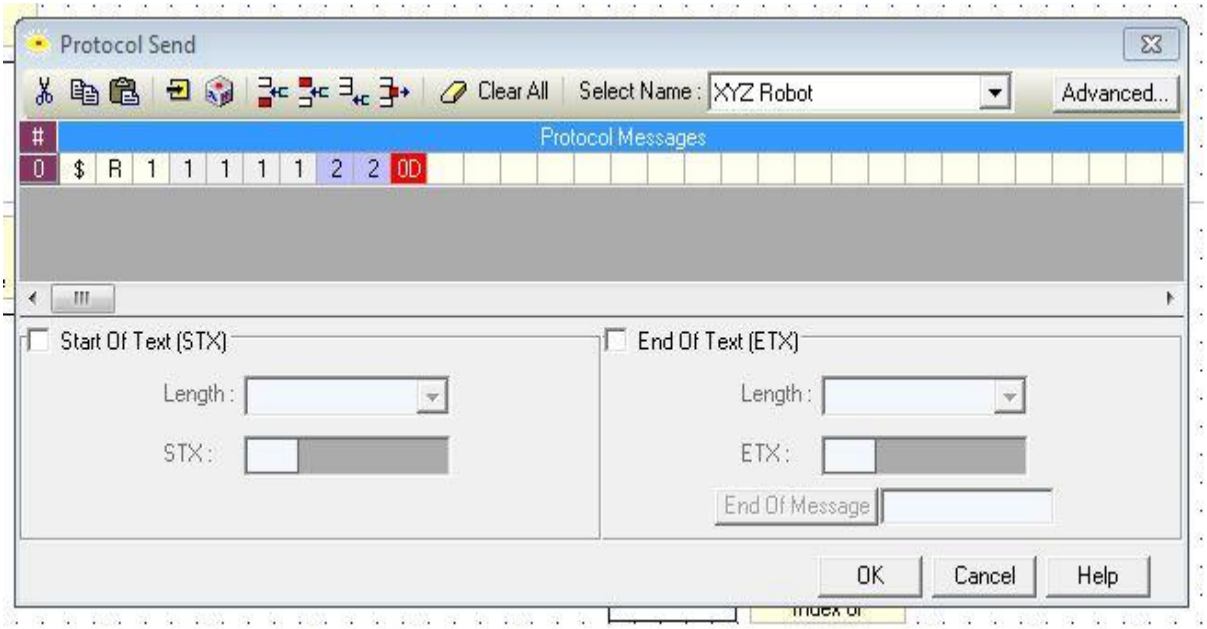
Select a carriage return (CR) as being the end of transmission character.....

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



.... and the whole message comes together in a simple to use function block!

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



The device on the receiving end of the transmission will reply with an acknowledgement or data depending on the message type. This reply will be in a similar format to the transmitted message and the PROTOCOL SCAN function block will disassemble this reply in much the same way as the PROTOCOL SEND function block assembled it.

Serial communication messages written in a modern PLC often take up to 40-50 lines of code to assemble, convert and transmit, clogging up the memory and distracting from the original objective of controlling the machine. Using a Unitronics function block not only makes the whole process simpler and easier to construct but it removes the clutter of incidental code leaving the control program nicely structured.