The protocols include error detection; e.g. if CRC/check sum is incorrect the PC is asked to repeat the last block.

Next I add code for saving the configuration, the format use was binary mode (not ASCII), which included an added CRC. Hence if the CRC fails the file is deemed corrupt.

I original sent the S19 (main code) file through the communication port in ASCII mode (as Access 4000), but put forward the idea of using binary mode as this would reduce upload time by half. My Ideal was given the go-ahead and the protocol was changed. Along with an Added 16-bit CRC instead of using Motorola's standard 8-bit check sum, giving better error detection.

Main Code: -

S224020000000046DA2000280000002040400020404000204040002040400020404000204040002040408

Use binary mode: -

SC DA NB B01 B02 B03 B04 B05 B06 …. CRCLOW CRCHIGH

SC = Start Character (ASCII character 'S' = 0x53)
DA = Digital Address (0x02)
NB = Number of bytes following, including 16-bit CRC (0x24)
B01 = Byte 1 (0x02)
B02 = Byte 2 (0x00)
B03 = Byte 3 (0x00)
….
CRCLOW = CRC low byte
CRCHIGH = CRC high byte

The original S19 file format check sum is replaced with 16-bit CRC (as used in Modbus), where the check sum is replaced with the lower CRC byte and the high byte is added to the end of the frame with the NB field being incremented. This all happens at the PC end.

Calculated the CRC

All bytes excepted the CRC bytes are used in the calculation.

Store block into an array: -

```
Unsigned char block[100];
block[00] = SC;
block[01] = DA;
block[02] = NB;
block[03] = B01;
…
block[??] = B??;
```

Then call CRC: -

```
Length = NB+1 = block[02] +1
         CRC16(block, length);
```

Why add one to NB?

NB does not count SC, DA and NB. But does count the 16-bit CRC, hence if we add one all bytes are used in the CRC calculation expect the two CRC bytes.

Changes to Controller protocol

Strtoul() or strtol() are not required, code can be simplified.

Use 16-bit CRC instead of checksum.

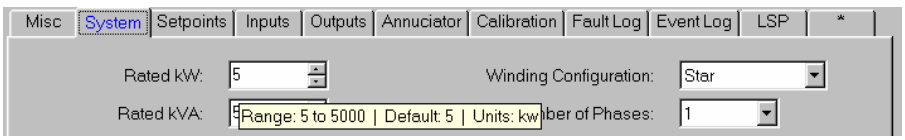
Week 45: Monday 11/06/2001 to Friday 15/06/2001

Continue working on my configuration program for the new Access 3000 controller.

Added code for Tabs 'Misc' and 'System' and tested using my test program running on a laptop connected via RS232. Change my made about the checking of the frame using ticks and 'X's, instead the following happens for invalid values: -

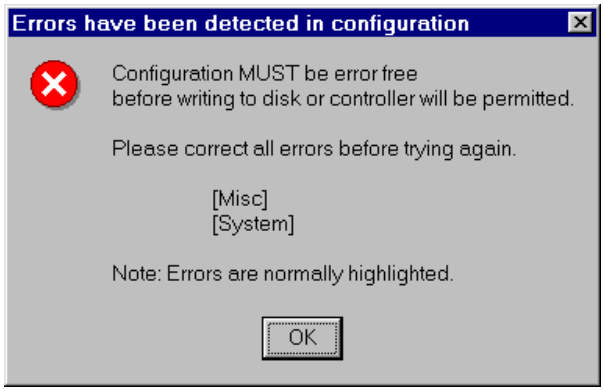
A small message pops up if that control is active and tells the user why it's wrong. Note: all invalid fields are highlighted in red. User will not be permitted to write configuration to file or controller if errors are present.

I added ToolTips, which display important information when the mouse floats over a field: -



I also added “Spin” controls to many edit boxes, allowing changes to the value using the “Up” and “Down” arrow button or using the arrow key while within the edit control (which has an attached spin control).

If errors are detected when user clicks “Write to Controller” or ties to save the configuration to disk the following dialog appears: -



Note the presents of “[Misc]” & “[System]” this tells the user that there are invalid fields in both of these tabs and all other tabs are error free.

Values information about the configuration is stored in an standard window INI file, if this file does not exist default values was are hard coded into program are use (only a valid INI file will override this hard coded values), the follow is part of the INI file to date: -

```
[WON]
min = 0
max = 4294967294
default = 0
tooltip = Range: 0 to 4294967294  |  Default: 0  |  Units: none
validchar = 0123456789
LimitText = 10


[WON - last number]
min = 1
max = 255
default = 1
tooltip = Range: 1 to 255  |  Default: 0  |  Units: none
validchar = 0123456789
LimitText = 3


[Spare 1]
min = 0
max = 255
default = 0
tooltip = Range: 0 to 255  |  Default: 0  |  Units: none
validchar = 0123456789
LimitText = 3
Label = Spare 1:


[Spare 2]
min = 0
max = 255
default = 0
tooltip = Range: 0 to 255  |  Default: 0  |  Units: none
validchar = 0123456789
LimitText = 3
Label = Spare 2:


[Spare 3]
min = 0
```

```
max = 255
default = 0
tooltip = Range: 0 to 255  |  Default: 0  |  Units: none
validchar = 0123456789
LimitText = 3
Label = Spare 3:


[Spare 4]
min = 0
max = 255
default = 0
tooltip = Range: 0 to 255  |  Default: 0  |  Units: none
validchar = 0123456789
LimitText = 3
Label = Spare 4:


[Unit Address]
min = 1
max = 255
default = 1
tooltip = Range: 1 to 255  |  Default: 1  |  Units: none
validchar = 0123456789
LimitText = 3


[Protocol]
default = 0
tooltip = Range: GenAccess or Modbus  |  Default: GenAccess  |  Units: none
Note    = 0 --> GenAccess, 1 --> Modbus


[Data Speed]
default = 0
tooltip = Range: 9600 to 38400  |  Default: 9600  |  Units: bps
Note     = 0 --> 9600, 1 --> 14400, 2 --> 19200. 3 --> 32768, 4 --> 38400


[Remote Config Flag]
default = 0
tooltip = Range: 0 or 1  |  Default: 0 |  Units: none


[Spare 5]
min = 0
max = 255
default = 0
tooltip = Range: 0 to 255  |  Default: 0  |  Units: none
validchar = 0123456789
LimitText = 3
Label = Spare 5:
```

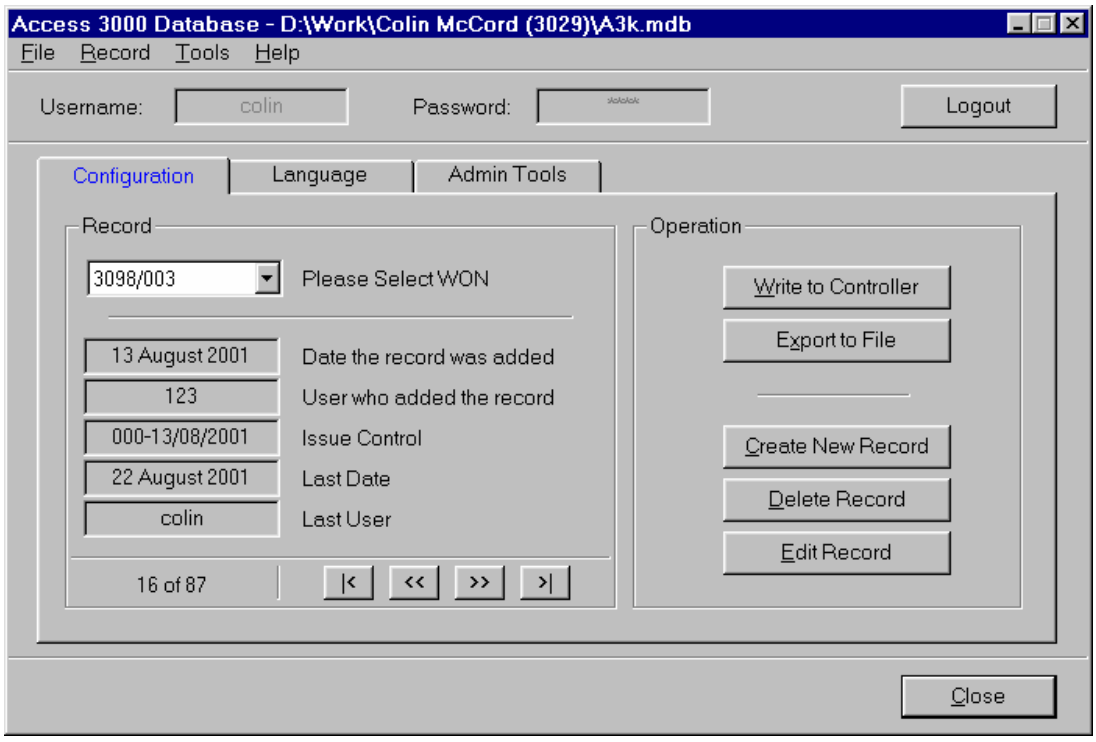
Note the tab [*] is for are use only and a master password will be set for access. This feature will not be documented and only a selected few will have access. In unless the master password has been entered on startup this tab will not be included.

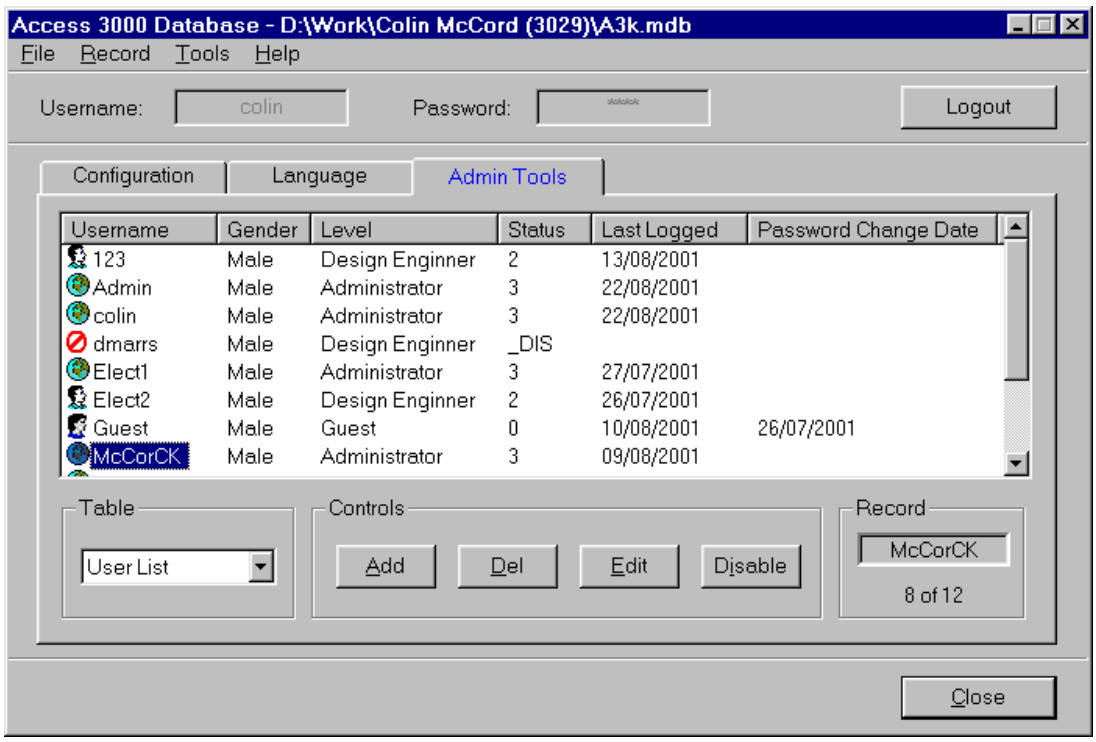
This section gives the user direct access to the Address Allocation Table, which can be written directly to the Access 3000 control, no error checks are carried out and the user must have detailed technical knowledge and is not designed for ease of use. Items like “Hours Run” can be changed to any value. FG Wilson does not want the customers getting access to this section, so great care has to be taken not to leak any passwords, this is FG Confidential RED.

The code for the Address Allocation Table is complete.

Weeks 46 to 56: Monday 18/06/2001 to Friday 31/08/2001

Finished configuration program for the new Access 3000 controller, including database section: -





Created a help file, wrote user manual and detailed technical report, which included: -

CONTENTS

Also carried out work on many other small projects.

THE END