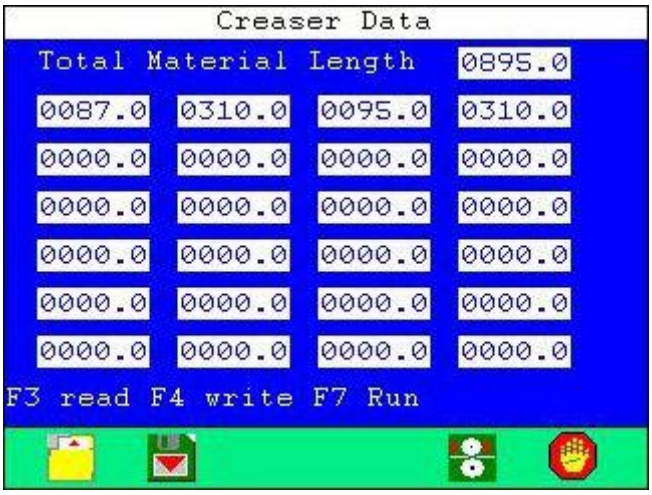
Making a data entry table is normally quite a difficult task in most programming languages. In MAP it can be very easy to write a data entry table. Please see the example below:



This is a typical data entry table

MAP makes the task simpler because it has dedicated commands that simplify producing tables. We believe that no other programming language can produce a menu like the one shown above with so little effort and so few lines of code. Shown below is the code used to produce the above menu. Map supports C++ style comments which are drawn using two slashes //, this signifies that from this point on all text is a comment and is ignored.

```
// title Creaser menu
RegType     data[0]R000     data[1]R009             // set registers 0 to 9 as integer
RegType     data[0]R010     data[1]R050     CC2     // set registers 10 to 50 as floating point
SetColors   data[0]0x001F   data[1]0xFFFF           // set the back ground color to Blue and the fore ground to peak white
DrawCanv    data[0]0                                // draw a clear canvas
DrawRect    data[0]0        data[1]0        data[2]100.00   data[3]08.00
DrawPos     data[0]33.00    data[1]0
SetColors   data[0]0xFFFF   data[1]0x0000
Text        str[Creaser Data] CC1
SetColors   data[0]0x001F   data[1]0xFFFF
DrawPos     data[0]05.00    data[1]10.00
Text        str[Total Material Length]  CC1
DrawPos     data[0]0        data[1]90.00
Text        str[F3 Read F4 Save  F7 Run] CC2
// draw the icons at the bottom of the screen
DrawBMP     str[c:openfile16.bmp]   CC2             // draw the F3 Folder Icon
DrawBMP     str[c:Savefile16.bmp]   CC3             // draw the F4 Save Icon
DrawBMP     str[c:continuous.bmp]   CC6             // draw the F7 continuous Icon
// set up the limits and draw the menu
TabLimit    data[0]2000.00  data[1]0        data[2]04.10    data[3]R002
TabPars     data[0]05.00    data[1]10.00    data[2]05.00    data[3]05.00
Table       data[0]4        data[1]25       data[2]R010     data[3]4
OnKey       str[a:readdata.mcp]     CC2         // on key F3 call the read data function
OnKey       str[a:savedata.mcp]     CC3         // on key F4 call the save data function
OnKey       str[a:runcreaser.mcp]   CC6         // on key F7 call the run creaser function
BrancFalse  data[0]R000     data[1]0        data[2]-21 CC11
End
```

(Fig 1.) Shows a data entry table as shown in Notepad+ on a PC

Programs can be written either on the controller or using an editor on your PC (using Notepad++ for example). TRM provide an XML file which is used to highlight the language making it easier to read and program. Looking at the top of the program you will see two commands RegType these commands are used to set a range of registers in a particular type, the first command sets 10 registers (0 to 9) to be an integer type. Integers in this case are of 32-bit long values that contain whole numbers that may be signed (can be positive or negative values). The second register type command sets registers 10 to 50 as floating point. Floating point variables are used to hold numbers that can contain a fraction such as 21.5 these numbers can also be signed, this means that they can

hold positive or negative values. The third command is the SetColors command, in this case it sets the background colour to blue on the foreground colour to white these colours are used to draw the canvas. In MAP, a screen is called a “canvas” the idea being that you can paint on the canvas and map controls all of this for you. Line 4 shows a DrawCanv (Draw canvas) command, it will draw a background in this case blue and if draws green box icon tray with some of the standard icons at the bottom of the screen.

On the menu shown above, there is a white box at the top of the menu with the title of the menu written in it, to make this bar we use 4 commands these are the DrawRect (draw rectangle command), the DrawPos (set draw position command), the SetColors command to set the background colour to White and the foreground colour to black and then the Text command to write the text. Since we set the colours online three, when the rectangle is drawn it will use the foreground colour that we have already set. Therefore the draw rectangle command will produce a white rectangle at position X (set by data[0]), Y (set by data[1]) and of width set by data[2], in this case 100% and finally the height of the box is set by the value in data[3] which is set to 8% in our example. The lines preceding the rectangle is a draw position command, like all other canvas commands this command works in percentage of screen size, so that the same program will work on different screen sizes making programmes easier to write. To set the position for the text “Creaser Data” we need to position this text at 33% in the X domain and zero in the Y domain to do this we use the draw position command and set the X position in data[0] to 33 and in data[1] zero. After positioning the draw position a SetColors command is used to change the colours to White and Black, before drawing the text with a Text command.

A Text has implementation options, these options are called “conditions”. There are more than 30 commands that have multiple conditions, these conditions are set in a variable called the “condition code” or CC when written on a PC using a text editor. You will notice from the above program that the text command has a CC of one written as CC1. When you write a map program on the controller, to make life easier, pulldown menus are used to make it simple to select an option. For example for a Text command would give three options, to use the command as a comment (this does not print on the screen), “**printable text**” to print on the screen or “Continue Print” (this allows long messages to be printed). If the program is being written on a PC, then these options are shown in the manual and start at 0 that is why we use CC1 to print the message we need on the screen. Incidentally if the value is zero it is possible not to bother writing the field, for example if we want to set the draw pointer to 33% in the X and zero the Y we can simply write DrawPos data[0]33 and the other data fields will automatically be set to 0.

The rest of the lines follow involve changing the colour to draw text on the screen where the text is set to white on a blue background, since we have already described printing text on the screen it is not necessary for us to describe the remainder, save to say that it is possible to write one text command that has text longer than the field with. Normally the maximum length of a text string is set to 16 characters, however in the editor these can simply be written as one line and on data importing the controller will automatically split this up into multiple text commands. To continue writing text use CC2 this allows long messages to be split up into multiple text commands. From the above you can see the comment shown by // draw the icons at the bottom of the screen this allows you to write icons into the icon tray at the bottom of the screen. The DrawBMP (draw BMP command) the data field of this command gives the file name of the bitmap file we want to draw. The command supports 16 or 256 colour bitmaps. These can be drawn with various apps for the PC such as MS paint. Condition code is used to indicate over which function key the image should be

drawn where zero gives function key F1. From the above program you can see that the draw bitmap command is used to draw three icons with condition codes of 2, 3 and 6 therefore the bitmaps are drawn above F3, F4 and F7 function key. It is also possible to use this command to put an image on the canvas by setting the CC to 8.

The next part of the program is used to draw the table itself there is a comment //set up the limits and draw the menu the first command in this block is the TabLimits (table limits command) the first data field, Data[0] is used to set the maximum value that can be input on the table, Data[1] gives the minimum value that can be entered into the table, for no limits enter zero in both fields. Data[2] gives the number of digits and precision that can be import or displayed on the table for example 4.1 shows four digits and one decimal point on the table. The next field is for future use. The table parameters command, TabPars is used to set display parameters for the menus that will be drawn, the first data field Data[0] is used to set the X starting point of the table in percent. The second data field Data [1] is used to set the Y starting position on the table in percent. The third field Data [2] is used to set the X spacing between the cells of the table, and the final field Data[3] is the Y spacing for the cells in the vertical direction. Drawing the table is achieved with the Table command, the first data field gives the number of columns that will be drawn on the table the second data field gives the number of members in the table (this is the number of inputs that you want the user to enter data for). This is an important data field to consider, to consider. From the above image you will see that that there is an odd number of data members and the table command has drawn a single field to the right of the image on its own. If we write -25 for the number of members the data field will be drawn with the odd member drawn at the end of the table to the left-hand side, this is useful so that you can split members out of the table. In this case the single data field is used to store the total length and this field is drawn on the right, allowing a text description to be used for this one member which separates it from the rest of the data entry field. The third data field of the table command is used to state the start register for data entry. In this case register 10 to register 35 are used for our data. The fourth data field of the table command is called the *check code*, this used to store the number of lines that will be run after various keypresses. For example function keys cause a table command to pass control to the line following the table command, however the table command operates rather like a DO/While loop and after the number of commands listed in the fourth data field has been executed control will be returned to the table command. Normally the commands following the table command will be used to check data or to select items from the menu such as to run the program.

Following the table command in our example there are three OnKey commands, these commands are used to call another program if the Key press by the user matches the key specified in the condition code of the command, where a CC value of zero represents F1 and seven represents F8. After executing an OnKey command control will be passed to the next instruction unless that is the last instruction specified by the check code variable.

The final command in our project is a BrancFalse command, which is used to test a condition and branch if that was it was not true. In this case the Branch instruction takes the branch if an OnKey command called a function. If no OnKey functions were active the branch instruction does nothing and the table command takes control returning control to the Table command. The Branch instruction causes a branch of -21 if an OnKey command was active the command will jump back to line 2 the SetColors command.

Keywords used:

A keyword is a special command that is the name for a command.

BrancFalse:	Branch if false, used to test a condition and branch if it is false.
CCn:	Condition code this is an option that controls features of a command.
DrawBMP:	Draws a bitmap image in the icon tray or on the canvas.
Data[n]:	There are a maximum 4 data fields that a command can use to store information.
DrawCanv:	Draws a canvas on the screen of the controller.
DrawRect:	Draws a rectangle in the canvas.
DrawPos:	Sets the draw position on the canvas.
OnKey:	Invoke a MAP function if a keypress matches a value stored in the condition code.
SetColors:	Allows the programmer to set the background and foreground colours.
TabLimits:	Used to set the limits of data and data displayed by the table command.
TabPars:	Used to set data parameters for the table command.
Table:	Table is used to create a table on the Canvas.
Text:	This command is used to provide a comment or write text the screen.