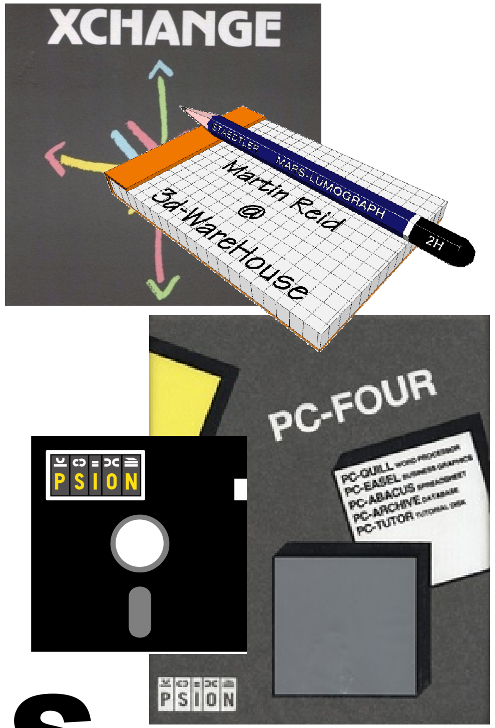
PSION Xchange / PC FOUR

Quill word processor

Abacus spreadsheet

Archive database
Easel business graphics



ABACUS

How to use this republished manual

Text in this style is a reproduction of the original manual
Text in bold like this is what is seen or typed in the Computer
Text in this font is taken from Dilwyn Jones and others tips trick and comments
Text in this font are comments from the Author (Martin)

ACKNOWLEDGEMENTS

The text in this republished manual is taken from the QL THOR Xchange manuals distributed by Gunther Strube & Erling Jacobsen on behalf of QUANTA and downloaded from Dilwyn Jones Sinclair QL Pages. Some of the Xchange references and commands are not available in the PC FOUR version but the reader will still find the manual covers the main aspects of the package.

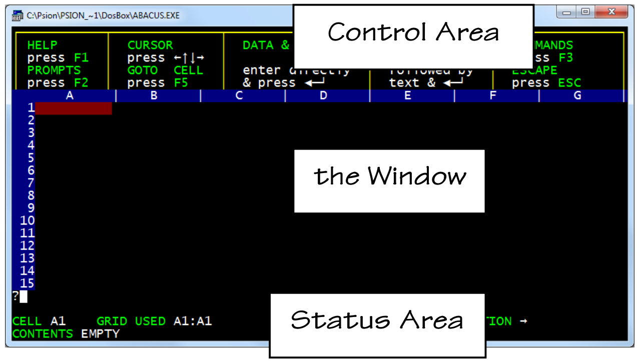
Contents

1.0 INTRODUCTION TO ABACUS

ABACUS is a spreadsheet program which can be used for planning, budgeting, tabulating data, calculation, information storage or for presenting information. The information in ABACUS is represented on a tabulated grid divided into 255 rows and 64 columns. The data area you see on the computer screen is a window through which you can see part of the grid. You can move this window rapidly across the grid. The intersections of the rows and columns represents 16000 cells in the grid. You can enter text into any cell or cells, or the cells may be used for the storage of numbers or data. The real power of ABACUS, however, comes from the use of rules, or formulae , which can connect different blocks, rows or columns of cells, or even individual cells of the grid. This means that information inserted in one area can immediately be evaluated and represented in another form elsewhere in the grid. For example, you can use twelve of the columns to represent months of the year and you can then enter sales data along a "sales" row. The next two rows can contain formulae to calculate the cost of sales (as a percentage of sales plus a fixed cost, for example) and the profit. The monthly profits will then be evaluated automatically each time you type in a sales figure. The yearly totals can also be summed by another formula, so that a change in the sales of, say, March will immediately lead to a completely different profit profile and total for the year. All the figures are evaluated automatically. ABACUS will use any text that you enter to refer to columns, rows and cells in the grid. It also automatically applies the formulae you enter to whole rows, columns or blocks of cells without your having to remember any complex command structure. ABACUS can be used in a very wide variety of planning and office tasks in finance, science, engineering, management etc.. ABACUS is an intelligent worksheet. It uses the names you use and it always prompts you with the most likely choices of values required. In addition it always informs you of what options you have, helps you to enter information and guides your decisions. The program is self-documented and includes comprehensive Help files. You may ask for Help at any stage, whatever you are doing. ABACUS is powerful, with a large range of built-in functions for operating on text as well as on numbers. In addition, as you learn more about the use of this program, you will discover facilities and commands which allow great versatility and flexibility in your work. These commands include the joining of grids, the use of multiple windows, the variation of column widths, justification of text and the use of different units (including monetary, integer, decimal and exponential forms). You may also protect your data - either at the level of protecting individual cells or by using a password to restrict access to a whole grid. You can also represent the data from ABACUS graphically in EASEL, in an ARCHIVE data file or in tabular form in a QUILL document, through the export commands. In fact in many respects ABACUS is like a visual programming language - but one which is easy to use. You may manipulate text, data, or formulae, use input and output statements and text variables. Your imagination is the only limit on the applications and possibilities of this program. The best way to learn about ABACUS is to use it. If you are not sure about something, try a simple example so that you can see how it works.

2.0 USING ABACUS

2.1 APPEARANCE


C:\Psion\PSION_~1\DosBox\ABACUS.EXE
Control Area
HELP press F1 PROMPTS press F2
CURSOR press ←↑↓→ GOTO CELL press F5
DATA & enter directly & press ↵
text & ↵
MANDS ss F3 ESCAPE press ESC
A B C D E F G
the Window
Status Area
CELL A1 GRID USED A1:A1
CONTENTS EMPTY
TION →

The Main Display. This is the appearance of the screen when you have just selected an ABACUS task. ABACUS shows 80 characters per line of the display.

The Window The central area of the screen contains the window showing part of the grid. Across the top of the window you will see a line in which a number of letters appear. These letters label vertical columns of cells making up the grid. As you can see, columns A,B,C, and so on are visible. Down the side of the window there is a series of numbers, from 1 to 15. These numbers label the rows of cells in the grid. A combination of a letter and a number will therefore identify one particular cell, and is known as a cell reference . Such a reference is, for example, A1. This refers to the cell which is in column A and row 1 (the top left hand cell).

The Status Area The bottom section of the display contain the status area, which gives information about the current state of the grid. It contains the cell reference of the current cell and its contents. This cell is, of course, empty when you have just selected an ABACUS task. In addition, the status area shows the extent of the used portion of the grid (as the cell reference of the bottom right cell of the used portion) and the name of the task.

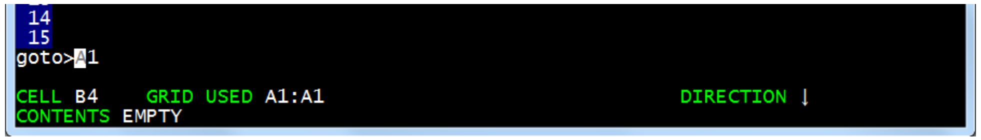
The Control Area The control area shows the normal options.

[F1]	to obtain Help
[F2]	to turn the prompts on and off
[F3]	to select a command
[F4]	to move from one split window to another
[F5]	goto a particular cell reference
[F6]	only used in Xchange - to switch tasks

In addition there are three options that are specific to ABACUS. These are to:
(a) move the cursor - use the arrow keys to move the cursor (red marker)
(b) type in data or a formula - start typing number or use =for formula
(c) type in text - start text with "

2.2 THE CURSOR

You will see that there is a cell which is different from all the others in that it is filled by a large (*red*) rectangle. This is known as the cursor and it marks the current cell, that is the cell which will receive any data you type in. The four cursor keys (the four keys marked with arrows on the numeric keypad) move the cursor around the grid. Try pressing the right cursor key once. You will see that the cursor moves one column to the right and the current cell indicator now shows B1. If you then press the left cursor key once the cursor returns to cell A1. Pressing the left cursor key again will have no effect because you are at the extreme left hand edge of the grid. Try using the four cursor keys to move the cursor around the window. You may have noticed, in your experimenting with the cursor keys, what happens when the cursor reaches the right hand side, or the bottom, of the window. If you have not, try it now. Keep pressing the right cursor key until the cursor reaches the extreme right hand side of the window. When you press the right cursor key again the cursor will not appear to move, but you will see that the letters across the top of the window will change. When you attempt to make the cursor leave the visible area of the grid the window will move across the grid so that the cursor remains in view. The window is always adjusted automatically to keep the current cell in the visible part. The cursor keys are a useful way of moving the cursor, provided you only wish to move it one or two cells. They are very inefficient for making large movements across the grid. For such large movements it is more convenient to go directly to the required cell. You can do this by pressing **F5** , to select the **GOTO** option, and then typing the required cell reference, followed by **[Enter]**. As an example of using the option to go directly to a particular cell, to move the cursor to cell D11.



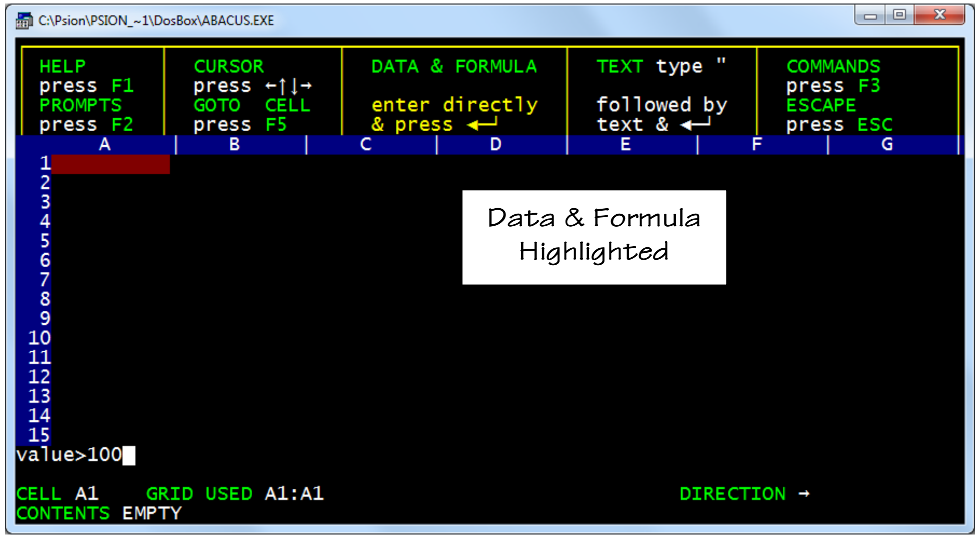
```
 14
 15
goto>A1

CELL B4    GRID USED A1:A1                         DIRECTION ↓
CONTENTS EMPTY
```

First press F5 to select the GOTO option. The words Goto>A1 will then appear in the line immediately below the window. ABACUS is suggesting that the cursor be moved to the top left hand corner of the grid. If you accept this suggestion (by just pressing [Enter]) the cursor will move to that point. To move the cursor to another cell, type in the cell reference - in this case type: **d11** and press **[Enter]** . The 'd' may be in upper or lower case. The cell reference you type in replaces that suggested and the cursor moves directly to the cell you have specified. You should now move the cursor back to the top left hand corner of the grid by using this option again. This time you can accept the suggested cell reference (A1) so all you have to type is: **F5 [Enter]** You will find that you go back to the original state of the display, with the cursor at the top left hand corner of the window, in cell A1. Try using this option to move around the grid, and finish by returning the cursor to cell A1. Now move the cursor to cell Y1, by typing in **F5 y1 [Enter]** Look at the letters labelling the columns across the top of the window and you will find what happens if you go beyond column Z. The column to the right of column Z is labelled AA, the next one is labelled AB, and so on. This enables you to refer to more than 26 columns. There are 64 columns in total and, after AZ, the columns are labelled BA, BB and so on. Move the cursor to cell AY1 to see this change in the labels. Use the right cursor key to move the cursor to the right until you can go no further; you will then have found the label BL, the last column in the grid. You can also move down the grid to find the last row but you will have to go a long way; there are 255 rows in the grid.

2.3 ENTERING NUMBERS

Return the cursor to cell A1 and then type 100 but don't press [Enter] just yet . Notice that the 'Data & Formula' option box in the control area is now highlighted, to confirm your action. The prompt value> , followed by the number 100 will also have appeared in the line immediately below the window. (This is the same place that Goto>A1 appeared when you were moving the cursor around the grid by use of F5 .)



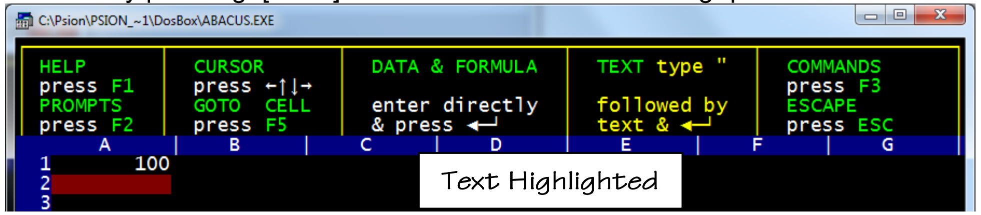
```
C:\Psion\PSION_~1\DosBox\ABACUS.EXE

HELP        CURSOR      DATA & FORMULA    TEXT type "     COMMANDS
press F1    press ←↑↓→                    followed by     press F3
PROMPTS     GOTO  CELL  enter directly    text & ←┘       ESCAPE
press F2    press F5    & press ←┘                        press ESC
     A          B          C          D          E          F          G
 1
 2
 3
 4
 5
 6
 7
 8
 9
10
11
12
13
14
15
value>100

CELL A1    GRID USED A1:A1                         DIRECTION →
CONTENTS EMPTY
```

All typed input, and the text that ABACUS shows while you are using a command, appears in this line. It is known as the input line. The small (*white*) rectangle in the input line marks where the next input character will appear, and is known as the input cursor, to distinguish it from the main cursor in the window. If you make a mistake at any time during typing to the input line, you can correct it by using the line editor. Press **[Enter]** now. When you do, the value 100 will be transferred to the current cell (A1) and the input line will clear, ready for more input. You will see that the value 100 also appears in the status area, at the bottom of the display. Although this facility might seem rather pointless at the moment, you will find it useful when you start to use formulae, as described in the chapter on Functions and Formulae. You may like, at this stage, to practise entering a few numbers at different points in the grid.

2.4 ENTERING TEXT Once you are familiar with number entry, putting text into a cell is simple. You follow exactly the same procedure, except that you start by typing quotation marks (") - you need to press the [Shift] key to get this symbol - into the input line. As soon as you type the quotation marks, ABACUS responds by emphasising the TEXT option box in the control area and showing text>" in the input line. You then type in exactly what you want to appear in the cell, followed by pressing [Enter]. There is no need for a closing quotation mark since

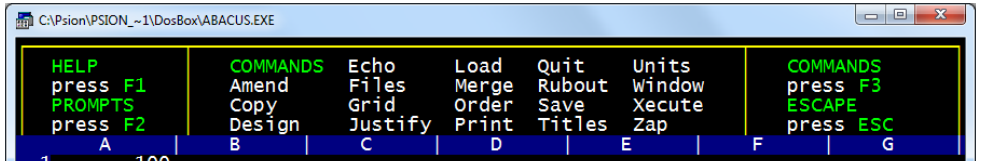


```
C:\Psion\PSION_~1\DosBox\ABACUS.EXE

 HELP        CURSOR        DATA & FORMULA    TEXT type "     COMMANDS
 press F1    press ←↑↓→                                      press F3
 PROMPTS     GOTO  CELL    enter directly    followed by     ESCAPE
 press F2    press F5      & press ↵         text & ↵        press ESC
       A          B           C          D          E          F          G
 1       100
 2
 3
```

ABACUS already knows you are typing in text. Try entering text into a few cells and, in particular, notice the difference between entering, say: 1000 [Enter] (a number) and "1000 [Enter] (text) A number is shown at the right of the cell, whereas text is placed at the left. The status area also shows the type of information; text, numeric and so on, in the current cell. If you have a large number of items to enter across or down the spreadsheet, you may find it easier to use the [Tab] key instead of [Enter]. Pressing [Tab] has the same affect as pressing [Enter] followed by the last arrow key that was pressed. The direction of the last arrow key pressed is displayed in the status area at the bottom of the screen.

2.5 THE COMMANDS As indicated by the top right hand option in the control area, you use a command by first pressing [F3].This changes the central part of the control area to show a list, or menu, of the available commands.



```
C:\Psion\PSION_~1\DosBox\ABACUS.EXE

 HELP        COMMANDS  Echo     Load    Quit    Units     COMMANDS
 press F1    Amend     Files    Merge   Rubout  Window    press F3
 PROMPTS     Copy      Grid     Order   Save    Xecute    ESCAPE
 press F2    Design    Justify  Print   Titles  Zap       press ESC
       A          B           C          D          E          F          G
```

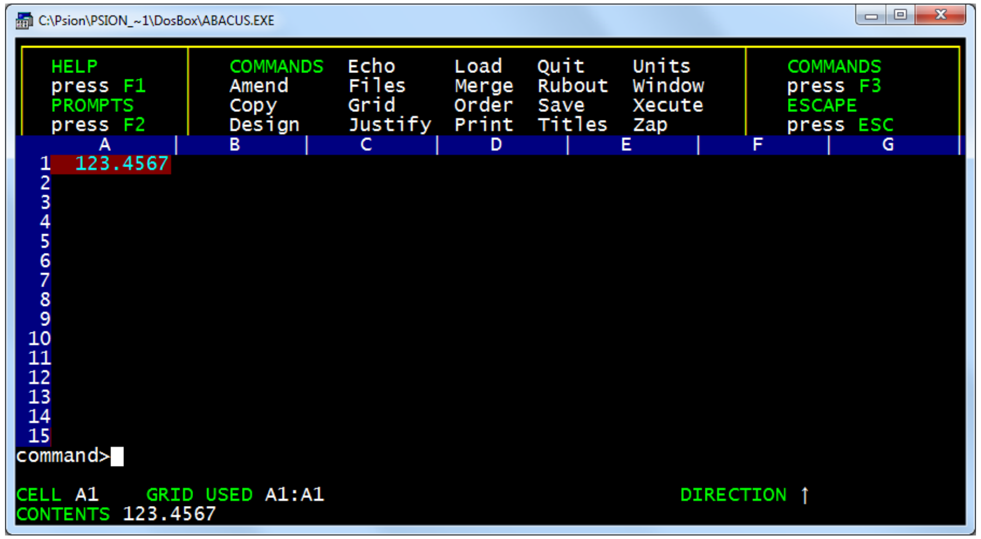
It is known as the command menu and is illustrated above. Most of the commands are described in later chapters but we can take a quick look at two of them. These are Zap, which you use to clear the whole grid, and Quit, which allows you to stop using ABACUS. Try the Zap command first. Press [F3] and locate the Zap command in the displayed menu. If you press the Z key, the word Zap will appear in the input line - you need never type more than the first letter of any command because ABACUS does most of the work for you. Also, the command box in the control area changes to show the menu for Zap, telling you that you have two options. Try pressing [Esc] first, to cancel the selection of this command and return to the main level without any deletions. Since you can not

recover the contents of a grid once you have cleared it, you will find this option useful if you select the command accidentally. (It is worth remembering that you can get out of any command, at any stage, by pressing [Esc] .) Now return to the command menu by pressing [F3] and then press Z to call the Zap command again, but this time press [Enter] next, to clear the grid. You will be left with a blank grid and with the cursor in cell A1, ready to start afresh. Whenever you want to abandon an ABACUS task, you must use the Quit command. This works in a similar way to Zap, in that you first press [F3] and then the first letter of the command Q . Quitting in this way causes you to lose the contents of your grid, so you are again given the option of cancelling the command and going back to the main level of ABACUS by pressing [Esc] . If you are sure you want to quit, press [Enter] , to exit. Some of the commands, unlike these two examples, offer you a whole range of related subsidiary options. The Files command, for example, includes the facilities to maintain your ABACUS files - to make copies, to delete a file that is no longer needed, to transfer information to another task and to rename a file. The Grid command also has a large number of options concerned with the overall appearance or functioning of the grid. These options, which offer many powerful facilities, are:

GRID - Option	***Use***
Insert, Delete	insert or delete one or more rows or columns
Width	change the width of one or more columns
Protect, Unprotect	add or remove protection against changes
Security	add or remove password security to a grid
Repeat	recalculate the grid a number of times
Goalseek	recalculate the grid until a condition is met

The last two options allow you to handle problems that require an iterative approach - ie they require a calculation to be performed several times, producing successively better approximations to the result.

2.6 MORE ABOUT NUMBERS AND TEXT Here is some additional information about the different ways ABACUS can display numbers and text in the grid. Because it is very detailed and uses several new commands, you may prefer to study this section after you have read the rest of the manual. ABACUS stores all numbers to an accuracy of 16 significant figures. It can display up to 14 significant figures - the extra two figures are used to make sure that the calculated value is displayed accurately. Although ABACUS calculates and stores all numbers to this accuracy, you do not have to display all 14 significant figures in the grid. To illustrate this difference, put the number 123.4567 in cell A15. The value displayed in the cell and the exact value shown in the status area will agree. Now select the Units command (by pressing **[F3]** and then the **U** key). There are two main options: Cells or Defaults. These two options are explained in the next chapter. In this case press **[Enter]** to select the (suggested) Cells option. The display will look like that shown on page 9 and you are now offered the choice of several different forms of display for numbers in the grid - let's try a few. First, press the M key (to select the Monetary form of display). ABACUS gives you the option to choose how you want it to show negative values. You can select to show them enclosed in brackets, or with a leading minus sign. ABACUS suggests that you use a minus sign and you accept this suggestion by pressing [Enter] .



```
C:\Psion\PSION_~1\DosBox\ABACUS.EXE

 HELP       COMMANDS  Echo     Load    Quit    Units     COMMANDS
 press F1   Amend     Files    Merge   Rubout  Window    press F3
 PROMPTS    Copy      Grid     Order   Save    Xecute    ESCAPE
 press F2   Design    Justify  Print   Titles  Zap       press ESC
     A     |    B     |    C     |    D     |    E     |    F     |    G
  1  123.4567
  2
  3
  4
  5
  6
  7
  8
  9
 10
 11
 12
 13
 14
 15
command>

CELL A1     GRID USED A1:A1                          DIRECTION ↑
CONTENTS 123.4567
```

If you want to show bracketed negative values you press the B key instead. In this example it does not matter which we choose, but we shall assume the minus sign option. ABACUS then asks you to specify the range of cells which are to be affected. You could reply by typing in a range reference (eg A1:B3 - see the next chapter) or just the reference to a single cell. In this case we want to change only the current cell (eg A1). You may do so simply by pressing [Enter] .The complete sequence of keypresses is: **[F3] U [Enter] M [Enter] [Enter]**

Just before you press [Enter] for the third time the input line should contain:

Command>units,cells,monetary,minus sign,range A1:A1

When you press [Enter] the display in cell A1 will change to £123.46, even though the actual value (123.4567) is still kept, and shown in the status area. ABACUS automatically takes you back to the main display. The monetary form of display always shows the number rounded to two decimal places, with a leading currency sign. (You can change the sign to $, or anything else, by using one of the options in the Design command, described at the end of the Functions and Formulae chapter). Let us now change the display in cell A1 to Integer (whole number) format, by calling the Cells option of the Units command again, but this

time press the I key. This format also allows you to choose whether to use a minus sign or brackets to show negative numbers and this time we can choose the bracket option by pressing the B key. Then type A1 and press [Enter] (again we are only affecting cell A1). The full sequence of keypresses in this case is:

[F3] U [Enter] I B [Enter]

and the input line shows: **Command>units,cells,integer,brackets,range A1:A1**
The cell display now shows 123 - the decimal point and all figures following it are not shown in integer format. We can now try Decimal format. For this, and the remaining formats, you do not have the option of displaying negative values in brackets. Instead (except for the General and Integer formats) you must specify the number of figures you want to be displayed after the decimal point; let's use five decimal places. After calling the Cells option of Units, in the usual way, press D to select Decimal format and then specify five decimal places. Finally, in response to the 'range' prompt, type A1 again, to affect only this cell. The full sequence of keypresses and the corresponding input line contents are:

[F3] U [Enter] [Enter] 5 [Enter] [Enter]

Command>units,cells,decimal,decimal places 5,range A1:A1

Cell A1 will now show 123.45670, as required. Now use the command again, but this time press the P key, to specify the Percent format. Use one decimal place and select cell A1. The full sequence of keypresses is:

[F3] U [Enter] P 1 [Enter] [Enter]

The display will now show 12345.7% - the percent option shows a number multiplied by 100 with an added % sign. Note that the stored value, as shown in the status area, is still 123.4567, regardless of the cell display. We can now try the Exponential format, with three decimal places, by typing:

[F3] U [Enter] E 3 [Enter] [Enter]

Before you press [Enter] for the final time the input line should contain:

Command>units,cells,exponent,decimal places 3,range A1:A1

and, after pressing it, the cell display will be 1.235 E+02. The exponential format is commonly used to display numbers which are too large or too small to be written in decimal format. The number is written as a value between 1 and 10, multiplied by the appropriate power of ten. The number 2,300,000,000, for example, can be written as 2.3 multiplied by 1,000,000,000. Now 1,000,000 000 is ten raised to the ninth power (nine tens multiplied together), so that 2,300,000,000 could be written in exponential format as 2.3E+09. Very small numbers are written in exponential format by using negative powers of ten. Thus, the number 0.000123, which is 1.23 divided by 10000 (ten raised to the fourth power), can be written in exponential format as 1.23E-04. The remaining option in the Cells option of the Units command is the General format, which you can see in cell A1 by typing:

[F3] U [Enter] G [Enter] The input line contains:

Command>units,cells,general,range A1:A1

This format again does not require you to specify the number of decimal places. Using the general option lets ABACUS choose a sensible format for the display of each number. It does the best it can to display each number as accurately as possible in the space that is available. Before we leave the Units command, try displaying the number in cell A1 in decimal format, with nine decimal places. Type:

[F3] U [Enter] [Enter] 9 [Enter] [Enter]

Cell A1 now shows ##### since the required display will not fit in the space available. Whenever you see a cell filled with pounds signs like this you know that there is not enough room for the number to be displayed. You should then either use the Units command to change the display format, or increase the width of that column with the Grid command. These commands are used frequently in the Examples chapter. Now clear the grid by using the Zap command and, with the cursor at cell A1, type: **"this is a long piece of text [Enter]** Although the text is too long to be contained in one cell, it is all shown - it overflows across the following cells. Now put the number **1** into cell **B1**. The text is cut off at the end of cell A1 as it is not allowed to overflow across another filled cell. Move the cursor back to cell A1 to verify that the whole of the text is still stored (see the status area) even if it is not displayed in full. Move the cursor back to cell B1 and use the **Rubout command** to erase the contents of this cell. When you call this command (press [F3] and then R) ABACUS will ask you to specify the range of cells whose contents you want to delete. In this case we only want to delete the contents of the current cell, B1, and can do so by simply pressing [Enter] . The full sequence of keys to press to rub out the contents of cell B1 is:

[F3] R [Enter]

Now that cell B1 is empty, the full text in cell A1 appears in the grid again.

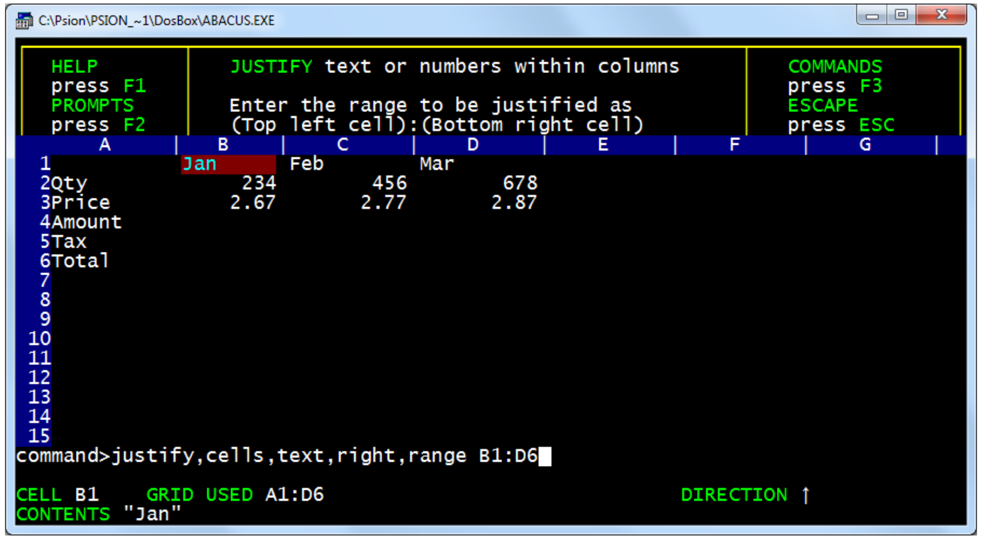
3.0 THE GRID CELLS

Much of the power of ABACUS lies in its ability to handle whole rows, columns or ranges of cells in a single operation. You do this by using simple expressions which allow you, for example, to fill all or part of a row of cells. The values in the cells may all be made the same or they may vary in a regular way. This chapter describes some of the properties of cells and the ways in which you can refer to them.

3.1 CELLS

The cell is the basic unit for holding information in ABACUS. Each cell can contain one item of information which may be text, a number or a formula. The USING ABACUS chapter tells you how to put either text or a number into a cell and the Functions and Formulae chapter explains how to use a formula. For each cell that contains information, ABACUS also keeps a record of how that information is to be displayed. You can, for example, display numbers or text at the left, centre or right of any cell, and you can display numbers in several different formats.

3.1.1 Justification You use the Justify command to change the position of the display within a cell. It allows you to select the position of numbers or of text within a cell or group of cells. Put a value of 100 in cell A1 and then use the Justify command by pressing [F3] and then the J key. ABACUS first asks you to select between a Cells and a Defaults option. These two have different effects, as we shall see later. For the moment select the Cells option by pressing [Enter] . ABACUS then asks you to choose between text (the option that ABACUS suggests) or numbers. Select numbers, by pressing the N key. Next you must select from the options of Left, Centre or Right justificaton, with Left being the one ABACUS suggests. Choose Left justification by pressing [Enter] . Finally ABACUS asks you to specify the range of cells that are to be affected. In this case just type in A1 and then press [Enter] . you will see that the value of 100 in cell A1 will move to the left hand side of the cell.



```
HELP          JUSTIFY text or numbers within columns          COMMANDS
press F1                                                      press F3
PROMPTS       Enter the range to be justified as              ESCAPE
press F2      (Top left cell):(Bottom right cell)             press ESC
      A      |    B    |    C    |    D    |    E    |    F    |    G    |
  1          Jan       Feb       Mar
  2Qty             234       456       678
  3Price          2.67      2.77      2.87
  4Amount
  5Tax
  6Total
  7
  8
  9
 10
 11
 12
 13
 14
 15
command>justify,cells,text,right,range B1:D6

CELL B1    GRID USED A1:D6                           DIRECTION ↑
CONTENTS "Jan"
```

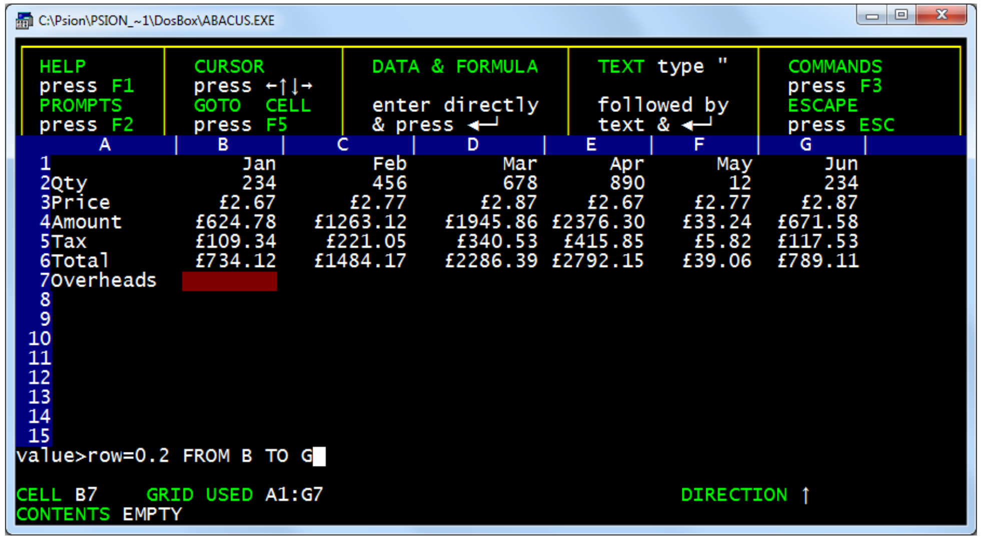
Try the other possibilities in the Cells option of the Justify command to see its effects on cells which contain text as well as those showing a numeric value. The Cells option of the Units command (to modify the display format for numbers) is described in the Using ABACUS chapter and follows a similar pattern. Note that you can change the numeric format or numeric justification of a cell which currently contains text. Nothing will appear to happen. If, however, you later change the contents of the cell to be numeric, it will be displayed with the format and justification that you gave to the cell. This also applies to a change of text justification for a cell which currently contains numeric information. ABACUS stores a numeric format, a numeric and a text justification with each cell that contains information.

3.1.2 Empty Cells Cells that contain no information do not exist as far as ABACUS is concerned, and use no memory. They can therefore have no properties. If you attempt to use the Cells option of either the Justify or the Units command on an empty cell they will have no effect. Clear the grid with the Zap command and then use the Cells option of the Units command to change the display format of cell A1 to percent format with one decimal place. [F3] U [Enter] P 1 [Enter] [Enter] Now type the number 0.5 into cell A1. You will see that it is not displayed as a percentage (you would expect to see 50.0%) but is in general format.

3.1.3 Default Cell Properties The reason why the previous example does not work is that each time you put information into a previously empty cell, it is created (memory is reserved for it) with a set of default properties (properties automatically supplied by ABACUS, without your having to specify them). When you have just selected an ABACUS task these cell defaults are that text is justified left and numbers are justified right. All numbers are displayed in General format. If you want to change these defaults you must use the Defaults option of either the Justify or the Units command (or both). For example, use the Defaults option of the Units command (press [F3] , U and then D) to select a default of percent format with one decimal place. The choices are similar to those in the Cells option, but you are not asked for a cell range. The default cell properties are used whenever you put information into a previously empty cell. Move the cursor to an empty cell and type in the number 0.5. It is now displayed as 50.0%. Every number that you put in a previously empty cell will now be displayed in percent format, until you change the Units default again. The Defaults option of the Justify command works in the same way. Again you are not asked to type in a cell range because ABACUS will use the new default each time you put information into any previously empty cell. Try using the Defaults options to make changes to the default justificaton for both numbers and text. See how they affect the numbers or text that you put into empty cells. The new default settings will remain in effect until you change them again, or until you finish using ABACUS by using Quit. Each time you start a new ABACUS task the settings take the original values. To restore the defaults to their original state - numbers justified right, text justified left and numbers displayed in General format - use the following sequences:

[F3] J D N R   [F3] J D [Enter] [Enter]   [F3] U D G

3.2 ROWS Very often you will want to fill several cells in a particular row with a particular value, or with values that vary in a regular way. ABACUS provides simple ways of doing this. One method is to refer to the cells of a row with a range identifier. There are two range identifiers, row and col . They refer to the cells of the current row or column - the row or the column that contains the cursor. As a first example, let us fill the '*Overheads*' row, from column B to column G, with the value 20. We shall use the range identifier row as follows. Place the cursor in cell A1 and then type: **row = 20 [Enter]** This means that the value 20 is to be placed in the cells of the current row (row 7), but so far we have not specified which columns are to be involved.

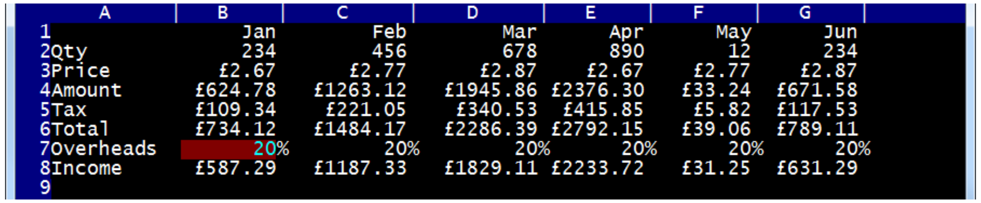

C:\Psion\PSION_~1\DosBox\ABACUS.EXE

```
 HELP           CURSOR           DATA & FORMULA    TEXT type "      COMMANDS
 press F1       press ←↑↓→                                          press F3
 PROMPTS        GOTO  CELL       enter directly    followed by      ESCAPE
 press F2       press F5         & press ←┘        text & ←┘        press ESC
         A          B        C          D         E        F         G
  1                Jan       Feb        Mar       Apr      May       Jun
  2Qty             234       456        678       890       12       234
  3Price         £2.67     £2.77      £2.87     £2.67    £2.77     £2.87
  4Amount      £624.78  £1263.12   £1945.86  £2376.30   £33.24   £671.58
  5Tax         £109.34   £221.05    £340.53   £415.85    £5.82   £117.53
  6Total       £734.12  £1484.17   £2286.39  £2792.15   £39.06   £789.11
  7Overheads
  8
  9
 10
 11
 12
 13
 14
 15
value>row=0.2 FROM B TO G

CELL B7     GRID USED A1:G7                          DIRECTION ↑
CONTENTS EMPTY
```

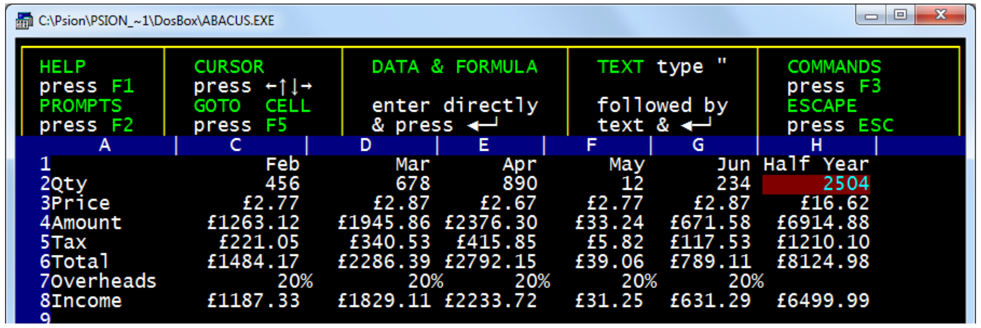
As soon as you press [Enter] a prompt appears in the input line suggesting that the row be filled starting at column A (the column containing the cursor). The system will always make a reasonable suggestion for the starting point. If this is what you want you can accept it simply by pressing [Enter] . In this case we want to start at column B so you should press: B [Enter] The input line changes to show that the filling of the row is to start at column B and a further prompt appears with a suggestion of D (the last column the last time the 'range' was used). This will have to be changed, since we want to end at column G, so you should press: G [Enter]
The instruction is now complete and will be carried out - the value 20 will appear in each of the cells from B7 to G7 inclusive and the input line will clear, ready for your next input. You must be careful to distinguish between row, used as a range identifier, and the function row(), which returns a row number (see the Functions and Formulae chapter). You use the range identifier, row, to indicate that the data following the equals sign is to apply to the cells of the current row, rather than just to a single cell. Every time you use it, ABACUS will ask for the start and end columns in the row, suggesting reasonable values based on your previous work.

You can accept the suggestion by pressing [Enter] or you can type in a replacement value, as described above. You can type in a one or two letter column reference or a column label (see later). If we change the 0.2 to a percentage with: [F3] U C P O [Enter] B7:G7 [Enter]. Then include an 'Income' row and calculate the income (Total less Overheads) with; row=total-(total*overheads) from B to G. then change the units to Monetary with: [F3] U C M [Enter] B8:G8 [Enter] we should get…



```
     A        B         C          D       E        F        G
1             Jan       Feb        Mar     Apr      May      Jun
2Qty          234       456        678     890       12      234
3Price      £2.67     £2.77      £2.87   £2.67    £2.77    £2.87
4Amount   £624.78  £1263.12   £1945.86 £2376.30  £33.24  £671.58
5Tax      £109.34   £221.05    £340.53  £415.85   £5.82  £117.53
6Total    £734.12  £1484.17   £2286.39 £2792.15  £39.06  £789.11
7Overheads    20%       20%        20%      20%      20%      20%
8Income   £587.29  £1187.33   £1829.11 £2233.72  £31.25  £631.29
9
```

3.3 COLUMNS Filling a column follows a very similar pattern except, of course, that you refer to a column by one or two letters rather than the number that identifies a row. Suppose we want to put the text "hello" in each of the cells of column D, from row 5 to row 11. We can do this by using the second range identifier, col . Move the cursor to cell D5 and type: col = "hello" [Enter] This time ABACUS suggests the correct starting point (row 5) as this row contains the cursor, and you can accept this suggestion by pressing [Enter] . Row 255 will then be offered as a suggested end point and you should change this by typing: 11 [Enter] in exactly the same way as in the previous example. The text will then appear in cells D5 to D11 inclusive and the input line will clear, ready for the next input. Again, you must be careful not to confuse col with the function col() , which returns a column number. Each time you use col you will be asked to specify the first and last row to be affected. You may, as usual, accept or replace the values that ABACUS suggests. You can give the start and end rows by typing either the row numbers or row labels (see later). In addition to this way of using the range identifiers row and col , you can also use them to specify the range of cells for any function that needs such a range (see the next chapter).



C:\Psion\PSION_~1\DosBox\ABACUS.EXE

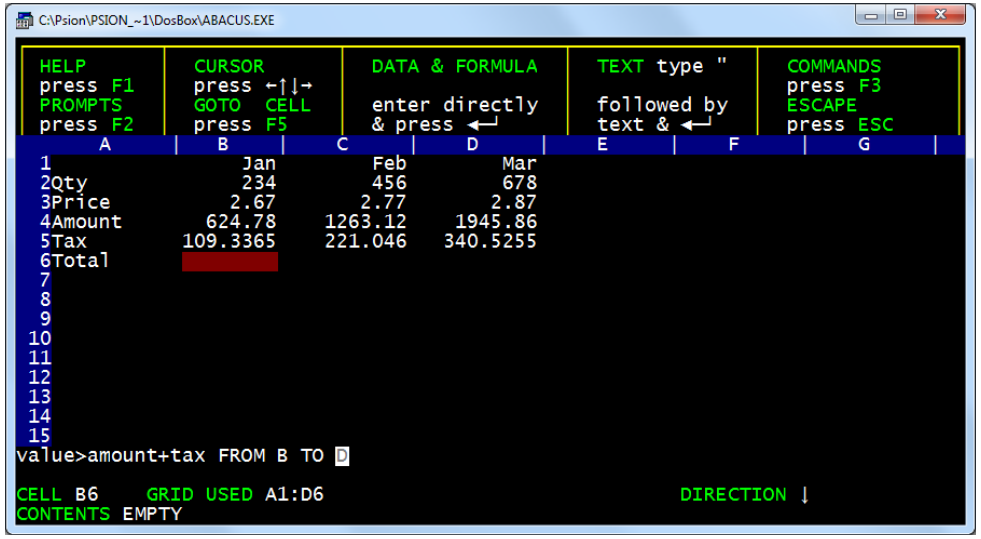
```
HELP        CURSOR       DATA & FORMULA   TEXT type "     COMMANDS
press F1    press ←↑↓→                                     press F3
PROMPTS     GOTO  CELL   enter directly   followed by     ESCAPE
press F2    press F5     & press ↵        text & ↵        press ESC
     A          C          D        E        F        G        H
1               Feb        Mar      Apr      May      Jun Half Year
2Qty            456        678      890       12      234      2504
3Price        £2.77      £2.87    £2.67    £2.77    £2.87    £16.62
4Amount    £1263.12   £1945.86 £2376.30   £33.24  £671.58  £6914.88
5Tax        £221.05    £340.53  £415.85    £5.82  £117.53  £1210.10
6Total     £1484.17   £2286.39 £2792.15   £39.06  £789.11  £8124.98
7Overheads      20%        20%      20%      20%      20%
8Income    £1187.33   £1829.11 £2233.72   £31.25  £631.29  £6499.99
9
```

col=sum(B2:G2) [Enter] from 2 [Enter] to 8 [Enter] Then [F3] R H7 [Enter] and then [F3] U C M [Enter] H3:H8 [Enter] also check out the use of Titles which can be seen above with column B hidden.

This example combines both ways of using row and col . The Examples chapter contains many examples of using row and col in both of these ways.

3.4 LABELS The previous examples referred to rows and columns by an explicit use of their number and letter cell references. An important alternative for identifying rows or columns is to use labels, that is names which you may choose yourself. These labels are then used to refer to specific rows, columns or cells. Any text that you put into a cell can be used as a label. You can use labels in any command or formula where you would otherwise use a letter and number reference to a single cell, a row or column, or a range of cells. The advantage is that it is much easier to remember names rather than numbers and letters when you want to refer to a cell or range of cells. This is an extremely powerful and flexible method which you can use to great advantage to simplify the setting out and operation of a grid. The following two sections explain how you can use these labels. The worked examples in the Examples chapter use this technique extensively but in this chapter we shall keep to simple situations.

3.4.1 Row and Column Labels A label may refer to either a row or a column, depending on the contents of the other cells in the grid.



C:\Psion\PSION_~1\DosBox\ABACUS.EXE

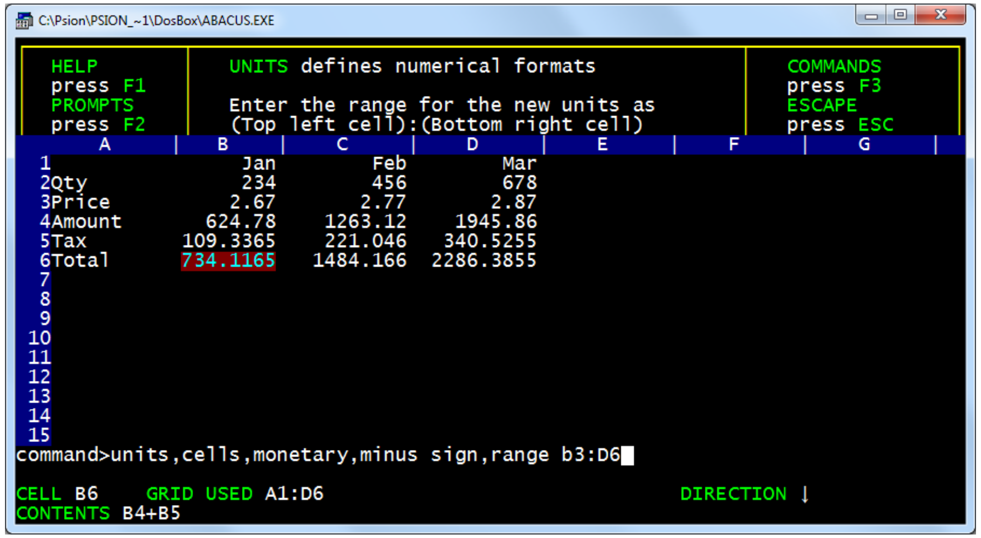
```
HELP        CURSOR       DATA & FORMULA   TEXT type "     COMMANDS
press F1    press ←↑↓→                                     press F3
PROMPTS     GOTO  CELL   enter directly   followed by     ESCAPE
press F2    press F5     & press ↵        text & ↵        press ESC
     A          B          C        D        E        F        G
1               Jan        Feb      Mar
2Qty            234        456      678
3Price         2.67       2.77     2.87
4Amount      624.78    1263.12  1945.86
5Tax       109.3365    221.046 340.5255
6Total
7
8
9
10
11
12
13
14
15
value>amount+tax FROM B TO D

CELL B6    GRID USED A1:D6                        DIRECTION ↓
CONTENTS EMPTY
```

The basic rule when you use a label to identify a row or column of figures is that ABACUS searches down the column and along the row, to the right, from the cell containing the label. The closest cell that contains a number, below or to the right of the position of the label, determines whether the label refers to a row or to a column. The example above should help make this clear.

3.4.2 Labelling Cells You can also use labels to refer to single cells, but in this case two labels are needed. In the example above the labels 'Feb' and 'Price' can be used to refer to cell C3 (containing the value £2.77). The reference is made up of the names of the two labels, separated by a full stop (eg, Feb.Price). It is not necessary to give the full names, and no distinction is made between upper and lower case letters. ABACUS needs only enough letters of each name to make sure that the identification is unique. In this example 'F.P' would be perfectly adequate. The order of the labels is also irrelevant, so you could also use 'Price.Feb' to refer to the same cell. As the worked examples will show, labels are powerful tools which result in enormous savings in time and effort during the creation or modification of an ABACUS application. They are, however, only tools and, like all tools, need a little care in their use. You must plan your grid entries carefully if you want to take maximum advantage of labels.

Confirm the setting of Units - [F3] U [Enter] M [Enter] B3:D6 [Enter]

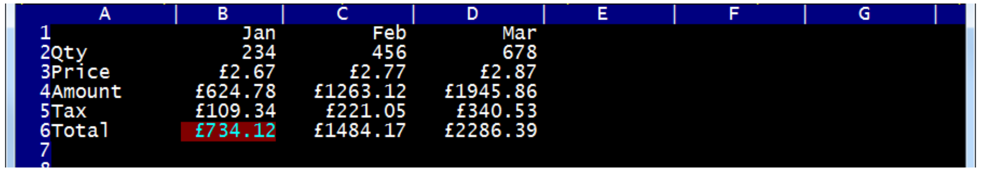


```
C:\Psion\PSION_~1\DosBox\ABACUS.EXE

 HELP          UNITS defines numerical formats              COMMANDS
 press F1                                                   press F3
 PROMPTS       Enter the range for the new units as         ESCAPE
 press F2      (Top left cell):(Bottom right cell)          press ESC
        A       |    B    |    C    |    D    |    E    |    F    |    G    |
  1                    Jan       Feb       Mar
  2Qty                 234       456       678
  3Price              2.67      2.77      2.87
  4Amount           624.78   1263.12   1945.86
  5Tax            109.3365   221.046  340.5255
  6Total          734.1165  1484.166 2286.3855
  7
  8
  9
 10
 11
 12
 13
 14
 15
command>units,cells,monetary,minus sign,range b3:D6

CELL B6     GRID USED A1:D6                             DIRECTION ↓
CONTENTS B4+B5
```

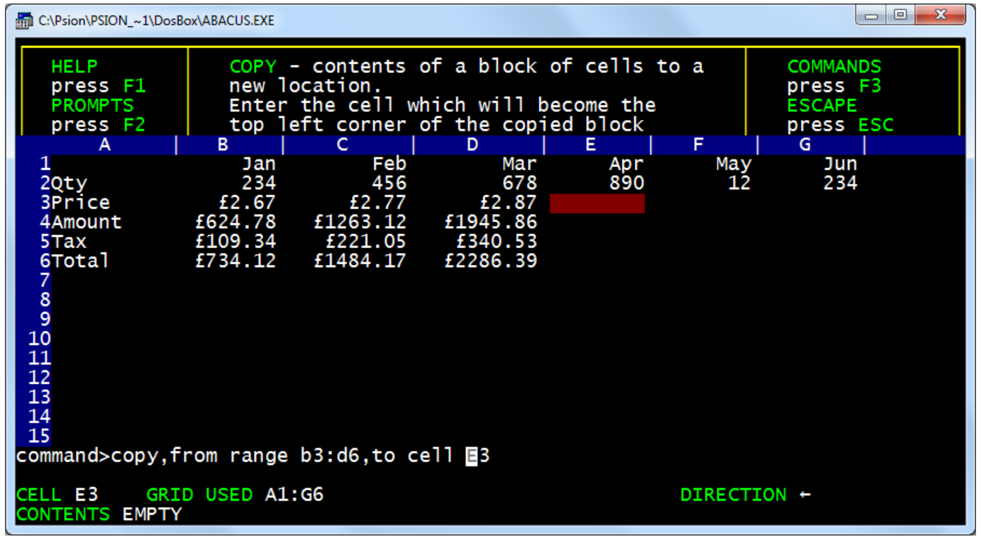
Results in range B3:D6 having 2 decimal place monetary values



```
        A       |    B    |    C    |    D    |    E    |    F    |    G    |
  1                    Jan       Feb       Mar
  2Qty                 234       456       678
  3Price             £2.67     £2.77     £2.87
  4Amount          £624.78  £1263.12  £1945.86
  5Tax             £109.34   £221.05   £340.53
  6Total           £734.12  £1484.17  £2286.39
  7
  8
```

3.5 RANGES

In addition to being able to refer to a whole row or a whole column, you can make an instruction work on a rectangular block, or range, of cells. In this case you may not use labels to replace letter and number references. A range reference is made up of two parts. The first part is the row and column reference of the top left hand cell of the range. This is separated by a colon from the second part, which is the row and column reference of the bottom right hand corner of the range. An example of the use of a range reference would be the use of the Copy command to copy the contents of a range of cells to a similar range at a different place in the grid. Below is a range reference (B3:D6) being used with the Copy command, to copy the range of cells to a range whose top left hand corner is at cell E3.

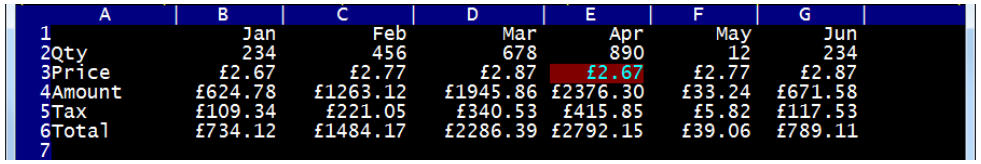


```
C:\Psion\PSION_~1\DosBox\ABACUS.EXE

 HELP          COPY - contents of a block of cells to a      COMMANDS
 press F1      new location.                                 press F3
 PROMPTS       Enter the cell which will become the          ESCAPE
 press F2      top left corner of the copied block           press ESC
        A       |    B    |    C    |    D    |    E    |    F    |    G    |
  1                    Jan       Feb       Mar       Apr       May       Jun
  2Qty                 234       456       678       890        12       234
  3Price             £2.67     £2.77     £2.87
  4Amount          £624.78  £1263.12  £1945.86
  5Tax             £109.34   £221.05   £340.53
  6Total           £734.12  £1484.17  £2286.39
  7
  8
  9
 10
 11
 12
 13
 14
 15
command>copy,from range b3:d6,to cell E3

CELL E3     GRID USED A1:G6                             DIRECTION ←
CONTENTS EMPTY
```

Resulting in



```
        A       |    B    |    C    |    D    |    E    |    F    |    G    |
  1                    Jan       Feb       Mar       Apr       May       Jun
  2Qty                 234       456       678       890        12       234
  3Price             £2.67     £2.77     £2.87     £2.67     £2.77     £2.87
  4Amount          £624.78  £1263.12  £1945.86  £2376.30    £33.24   £671.58
  5Tax             £109.34   £221.05   £340.53   £415.85     £5.82   £117.53
  6Total           £734.12  £1484.17  £2286.39  £2792.15    £39.06   £789.11
  7
```

Many of the commands ask you to type in a range reference, to identify the cells on which they are to work. You can specify the range in one of two ways. These are:

1) With explicit row and column numbers and letters, eg B3:D6
2) If the range you require is within the row or column that contains the cursor you can type one of the range identifiers, row or col. In this case, ABACUS suggests suitable start and end points, as described earlier.

4.0 FUNCTIONS AND FORMULAE

4.1 FUNCTIONS ABACUS contains a number of pre-defined functions which are used to perform specific calculations on the contents of one or more cells. A function takes a number of input values, known as arguments, and from them calculates a specific result . The result is said to be the value that the function returns.In ABACUS you must supply the arguments in brackets after the name of the function and, if there is more than one argument, you must separate them with commas. Most of the functions provided return a numeric value, for example, the function sum() which we met briefly in the previous chapter. This takes, as an argument, a range reference and returns a numeric value equal to the sum of the numeric values contained in all the cells within the range. Some functions, such as month() , return a text value - month(1), for example, returns the text "January". A few functions require no arguments, but you must still include the brackets. An example of such a function is pi() which returns the numerical value of the mathematical constant pi (approximately 3.14). Two particularly useful functions are col() and row(). These return the number of the column (or row) which intersect at the cell that contains the function. They are used extensively in the worked examples in the next chapter. For example, col() will return a value of 1 from column A, 2 from column B, and so on. The function row() simply returns the row number. As an example we can use the two functions month() and col() to label columns of the grid. The object will be to place the headings January, February, and so on at the top of columns B to M. We use the function col() to supply the number that month() needs as its argument, so that it gets a different value in every column. Type in:

row = month(col()) and then press [Enter].

Select the range from B to M when ABACUS asks for the start and end columns. You will see that the result is not quite what we want in that, although the labels start at column B, the first label is February and not January. This is because, in column B, col() returns the value 2 and month(2) is the text "February". We can correct this mistake by making sure that the argument given to the function month() is 1 when in column B, 2 when in column C, and so on. All we have to do is to alter the instruction so that 1 is subtracted from the value returned by col() , before calculating the month. Type in:

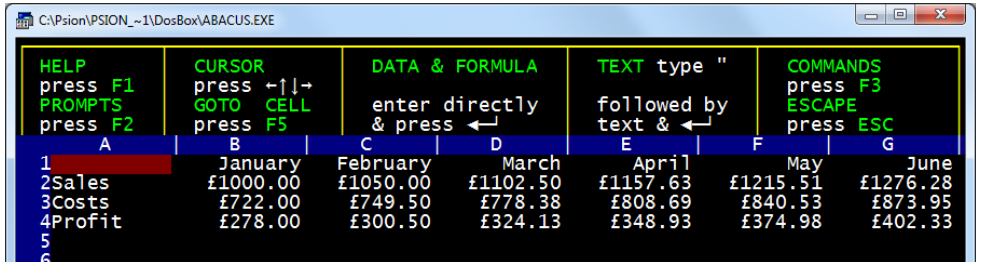
row = month(col()-1) [Enter]

Select the column range from B to M, as before. This now gives the correct result, with the names of the months starting with January in column B.

4.2 FORMULAE A formula is usually used to relate the contents of one cell to the contents of one or more of the other cells in the grid. The idea of formulae is very important in the use of ABACUS as it allows you to describe even the most complicated calculations in a simple way. You enter a formula into a cell using the same method employed for entering numbers, that is, by moving the cursor to the cell, typing it and then pressing [Enter] . ABACUS assumes that anything it does not recognise as a number (starting with a numeric digit) or a text value (starting with quotation marks) is a formula. (The examples of the previous section all put formulae into the cells of the grid, so it is not such a new idea after all.) Let us first try a very simple example. Move the cursor to cell B3 and enter the number 100, move the cursor to cell C3 and enter 200. Now move the cursor to cell D3 and type in the following formula B3 + C3 When you press [Enter] you will see two things happen. First the value 300 will appear in cell D3; the formula's result has been calculated by adding together the contents of cell B3 and cell C3 and the total placed in cell D3. In addition you will see that the status area at the bottom of the screen shows the formula used to calculate the value in this cell. A cell which contains a formula will always show the result of the calculation. If you position the cursor on the cell then ABACUS will show the formula itself in the status area at the bottom of the screen. The rest of the examples using formulae make use of the labelling facility and the row and col range identifiers first described in the previous chapter. They allow much more efficient methods of entering information into the grid than the direct use of letter and number cell references, such as in the last example. Note that any numeric formula that does not contain any cell references is not stored as a formula. In such a case ABACUS calculates its value and stores the result as a pure number. For example, 32+10/20 is stored as the value 32.5, and not as the original formula.

4.3 A SIMPLE CASH FLOW EXAMPLE

We aim to produce a grid which looks like that this Simple Cash Flow Analysis.



```
HELP          CURSOR         DATA & FORMULA    TEXT type "      COMMANDS
press F1      press ←↑↓→                                        press F3
PROMPTS       GOTO  CELL     enter directly    followed by      ESCAPE
press F2      press F5       & press ↵         text & ↵         press ESC
     A             B          C          D          E          F          G
1                  January   February      March      April        May       June
2Sales            £1000.00   £1050.00   £1102.50   £1157.63   £1215.51   £1276.28
3Costs             £722.00    £749.50    £778.38    £808.69    £840.53    £873.95
4Profit            £278.00    £300.50    £324.13    £348.93    £374.98    £402.33
5
6
```

Start this example with a grid containing month headings in cells B1 to M1. If you have anything else in the grid you should clear it with the Zap command and enter the month headings, as described in the first section of this chapter.

row=month(col()-1) [Enter] B to M then [F3] J C T R B1:M1 [Enter]

Now move the cursor to cell A2 and type: "Sales [Enter] and then put the value 1000 in cell B2. Now move the cursor to cell C2 and type in the formula: row=sales.january*1.05 [Enter] Accept the range selection given by ABACUS (column C to column M) by pressing [Enter] twice. Note that ABACUS knows the end of the row is at column M because that is where the previous row ended. When you press [Enter] a second time you will see a whole series of values appearing in row 2, from column C onwards, and the formula B2*1.05 will appear in the status area at the bottom of the screen. If you move the cursor along row 2 you will see that the formula for each cell is slightly different. In each case the formula takes the contents of the cell on the immediate left and multiplies it by 1.05 to obtain the value to place in the current cell. For example, the formula in cell E2 refers to cell D2, and the formula in cell H2 refers to cell G2, and so on. This process is completely automatic. ABACUS remembered that the original definition of the formula was in cell C2 (ie at the position of the cursor) and referred to the contents of cell B2 ('sales.january') one column to the left. The relative separation, from the cursor position to the cell referred to in the formula, is used in all the cells of the row. In ABACUS all formulae work in this way unless you specify otherwise. Each formula remembers the relative separation from the cell containing the formula to each of the cells to which it refers. When such a formula is used in more than one cell the references are adjusted to maintain such relative cell references. The multiplication table and auto-scaling bar chart developed in the Examples chapter explain how you can change this normal behaviour. The value in cell B2, to the right of the label "Sales", makes this label a row reference. The formulae use this value to calculate all the other values in the grid.

Now position the cursor at cell A3 and type the row label: "Costs [Enter] Without moving the cursor, type in the formula: costs = sales * 0.55 + 172 [Enter] This formula calculates the cost from two components. They can be regarded as manufacturing costs (55% of sales) and fixed costs totalling 172.00. Use the suggested start and end points of column B and column M. Since the contents of the row is defined in terms of the row reference "Sales", the label "Costs" will also be taken as a row reference, with the same range as "Sales". Again you should move the cursor along the row, examining the different formulae shown at the bottom of the screen, in order to understand how the results have been calculated.

Finally, put the label "Profit" in cell A4, as in the previous two cases, and type in a further formula: profit = sales - costs [Enter] with the same range selection as before (i.e. columns B to M). ABACUS will do all the rest of the work for you, producing a simple, but complete, example. If you now change the display to monetary format with the command (remember to press [F3]):

Units,Cells,Monetary,Minus sign,Range B2:M4

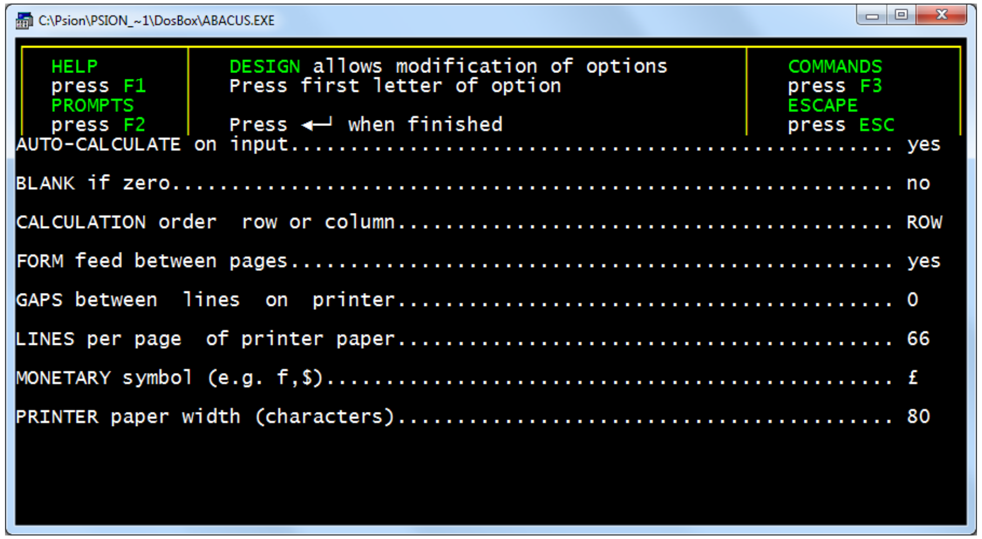
you should find that the first few columns appear as in the example on page 20.

4.4 AUTO-CALCULATION

When you have typed in the simple cash flow application described in the previous section, try changing the number in cell B2 (Sales.January). Move the cursor to this cell - the easiest method is to press F5 and then type in the cell reference (either B2 or sal.jan) followed by [Enter]. Now type in any number you like. When you press [Enter] you will see that all the numbers in the grid will change! The reason is that the values of all the formulae in the cells of the grid are recalculated automatically each time you make an entry to a cell. Since all the formulae in this example refer, directly or indirectly, to the value held in cell B2, all their values will change when you alter the contents of this cell. (Remember that we assumed that sales would increase by 5% per month, based on the January figure.) This facility makes it very easy to use ABACUS as an aid to making management decisions. You can alter a value and see immediately what effect it has on the rest of the figures in the grid. Even in this simple cash flow example, you can see how changes in sales, manufacturing costs and fixed costs will affect the profits.

4.5 THE DESIGN COMMAND

You can switch off the auto-calculate facility by means of one of the options in the Design command. This is useful, for example, when you have many complicated formulae in the grid and do not want to wait for a recalculation each time you change a single value. This feature is used in the Fourier Analysis example in the next chapter. Try switching off the auto-calculate facility by pressing [F3] and then the D key, to call the Design command.



```
 HELP            DESIGN allows modification of options              COMMANDS
 press F1        Press first letter of option                       press F3
 PROMPTS                                                            ESCAPE
 press F2        Press ◄┘ when finished                             press ESC
AUTO-CALCULATE on input.............................................. yes

BLANK if zero........................................................ no

CALCULATION order  row or column..................................... ROW

FORM feed between pages.............................................. yes

GAPS between  lines  on  printer..................................... 0

LINES per page  of printer paper..................................... 66

MONETARY symbol (e.g. f,$)........................................... £

PRINTER paper width (characters)..................................... 80
```

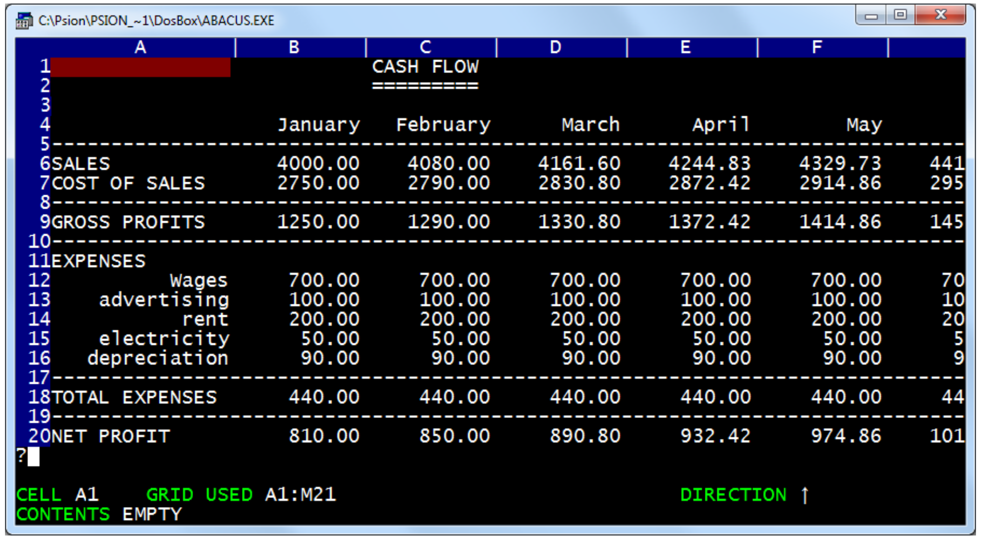
The display changes to show a list of the options. You can select any one of these options by typing its first letter. Select the Auto-calculate option by pressing A . You will see that the auto-calculate state changes automatically. You leave the command by pressing [Enter] . ABACUS returns you to the main display. If you now change the contents of cell B3 you will see that there is no change in the contents of any of the other cells. You can also force a recalculation of all the formulae in the grid at any time by using the Xecute command. While you have the auto-calculate turned off, try using this command. Make sure that the command menu is displayed in the control area (press [F3]) and then press the X key. The values in the cells of the grid will be recalculated, just as for a normal auto-calculate. The Xecute command returns you, as usual, from the command menu to the main display. Before you go any further you should restore the auto-calculate facility by using the Design command again. Select the Auto-calculate option by pressing the A key, as before, and leave the command by pressing [Enter] .

5.0 THE EXAMPLES

The following sections illustrate the use of ABACUS by developing a number of examples. In addition to explaining the way a number of features work, the examples have been chosen to show some of ABACUS's wide range of applications. The best way to learn about ABACUS is to use it. The examples have been written with this in mind, rather than with the intention of their being serious, professional applications. You are recommended to work through all the examples yourself, typing them in as you go along. Each contains some additional information, as well as giving more practice with the topics covered in earlier examples. At a later stage you might like to modify or extend them to match your needs and they should give you ideas about how to construct applications of your own. Above all, you are encouraged to experiment. You cannot do any harm, either to the computer or to ABACUS, and the more things you try out, the faster you will learn. In all the examples in this chapter text, numbers and formulae are shown exactly as you would type them in. If a cell range is required it will be given in brackets at the end of the line. In many cases the range you need will be the one that ABACUS suggests and you can select it simply by pressing [Enter] . In other cases you will have to type in the range yourself, in the format ABACUS requests. If the cursor needs to be positioned on a particular cell, its cell reference is shown in square brackets at the beginning of the line - do not type in any such cell reference. For example, the line: [A4] row=month(col()-1) (columns B to M) should be read as: move the cursor to cell A4, and then type in: row=month(col()-1) [Enter] if necessary, modifying the range suggested by ABACUS to be from column B to column M. Where you have to type in an explicit range reference, eg b3:e15, it will be given in that form. You do not have to worry whether you type the letters in upper or lower case - ABACUS will accept either. Choose upper or lower case letters to give the best appearance to the text in the grid. The commands used in the examples are shown exactly as they will appear in the input line. Remember that you only need to type in the first letter of each option - the rest is filled in by ABACUS. If you want to use the default option (the one suggested by ABACUS) you should just press [Enter]. Each example assumes that you start with a completely blank grid. If necessary clear the grid with the Zap command before starting to type in the example.

5.1 CASH FLOW MODELLING

This is a more complete version of the simple cash flow example of the previous chapter. When you have finished the grid it should look like. F2 Hides and Shows the Control Area



```
          A          |    B    |    C    |    D    |    E    |    F    |
 1                          CASH FLOW
 2                          =========
 3
 4                  January  February     March     April       May
 5-----------------------------------------------------------------------
 6SALES              4000.00  4080.00   4161.60   4244.83   4329.73   441
 7COST OF SALES      2750.00  2790.00   2830.80   2872.42   2914.86   295
 8-----------------------------------------------------------------------
 9GROSS PROFITS      1250.00  1290.00   1330.80   1372.42   1414.86   145
10-----------------------------------------------------------------------
11EXPENSES
12            Wages   700.00   700.00    700.00    700.00    700.00    70
13      advertising   100.00   100.00    100.00    100.00    100.00    10
14             rent   200.00   200.00    200.00    200.00    200.00    20
15      electricity    50.00    50.00     50.00     50.00     50.00     5
16     depreciation    90.00    90.00     90.00     90.00     90.00     9
17-----------------------------------------------------------------------
18TOTAL EXPENSES      440.00   440.00    440.00    440.00    440.00    44
19-----------------------------------------------------------------------
20NET PROFIT          810.00   850.00    890.80    932.42    974.86   101
?

CELL A1     GRID USED A1:M21                     DIRECTION ↑
CONTENTS EMPTY
```

The Completed Cash Flow Grid. (first five columns)

The first two cell entries produce an underlined title for the grid.

[C1] "CASH FLOW

[C2] rept("=",len(c1)) We shall use such a heading for each example. The second entry uses the rept() function. This needs two arguments. The first is text, or a reference to a cell which shows a text value and the second is numeric. The function produces that number of repetitions of the first character of the text. In this case it underlines the title with '=' signs, to the exact length of the title. If you decide to change the title there is no need to alter the formula in cell C2 since it uses the len() function to read the length of the text in cell C1.

[A4] row=month(col()-1) (columns B to M)

[A5] row=rept("-",width()+1) (columns A to M) These row entries produce month headings, as described in the previous chapter, and rule a line across the whole of the used part of the grid. The function width() gives the width, in character spaces, of each column. It can therefore be used to rule lines across a grid with columns of different widths. There is one extra character space separating each column of the grid, which is why the additional +1 is needed.

[A6] "SALES

[B6] 4000

[C6] row=sal.jan*1.02 (columns C to M)

These entries fill in the sales figures for the year, assuming that the January sales were £4000 and that sales are increasing at 2% per month.

[A7] "COST OF SALES cos=sal*0.5+750 (columns B to M)

(The costs are assumed to be half of the selling price plus a fixed amount of £750.00.)
[A8] row=a5 (columns A to M)
[A9] "GROSS PROFIT gro=sal-cos (columns B to M)
This rules off the grid again and calculates the monthly gross profit figures.
[A10] row = a5 (columns A to M)
[A11] "EXPENSES

[A12]	"wages	row=700 (columns B to M)
[A13]	"advertising	row=100 (columns B to M)
[A14]	"rent	row=200 (columns B to M)
[A15]	"electricity	row=50 (columns B to M)
[A16]	"depreciation	row=90 (columns B to M)

These entries fill in the expense figures, assuming them to be constant throughout the year. You can, of course, change the expense headings and amounts to suit yourself. You can include more or fewer entries, as long as you make the necessary changes to the cell references in the rest of the example. You may want to have different values for each month, but it is faster to set up the table with fixed values and modify them later. A simple way to change any value is given at the end of the example.
[A17] row = a5 (columns A to M)
[A18] "TOTAL EXPENSES
[B18] row=sum(col) (rows 12 to 16, columns B to M)
[A19] row=a5 (columns A to M)

You now have the totals of the monthly expenses. The sum() function adds the contents of all the numeric cells in the range specified as its argument. All empty cells, together with those containing text, are ignored. The range could be given as an explicit range reference - B12:B16 for example. In this case, however, each range is only a single column so we have used the range specifier col. All you need to do is to answer the range questions asked by ABACUS, and just press [Enter] if the suggested range is what you want. Note that this formula uses the range identifiers row and col in the two different ways that were mentioned in the Grid Cells chapter. Firstly, row is used to indicate that the formula is to be placed in several cells of the current row. Secondly, col is used to specify the range of cells over which the addition should take place. Both of the range identifiers need you to confirm (or change) their beginning and end points. In this case ABACUS deals with the range for the sum() function first.
[A20] "NET PROFIT   net=gross-tot (columns B to M)
[A21] row= rept("=",width()+1) (columns A to M)

The table is now complete, with the net profit figures calculated as the difference between the gross profits and the total expenses. All you have to do now is to adjust the appearance of the table by using a few commands. Remember to press [F3] each time you want to use a command. First we change the width of column A (note that the Grid command has its own menu of options).

[F3] Grid >Width, 15 FROM a TO a
Then we change the justification and numeric display format for a few cells:
[F3] Justify,Cells,Text,Right,Range a4:m4
[F3] Justify,Cells,Text,Right,Range a12:a16
[F3] Units,Cells,Decimal,Decimal places 2,Range a1:m21

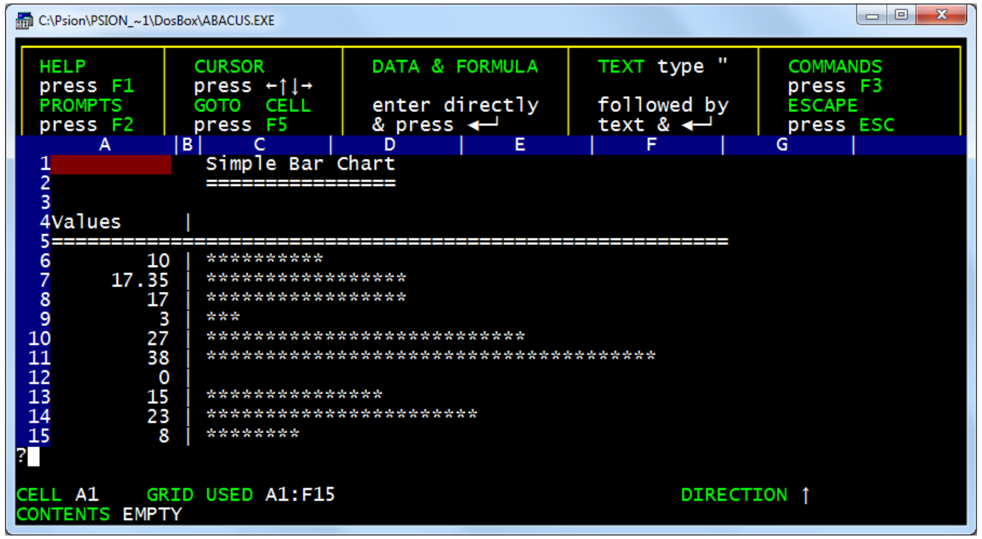
We have chosen to display the figures in decimal format, with two decimal places. If you prefer the pound sign to appear you should replace the last command by

[F3] Units,Cells,Monetary,Minus sign,Range a1:m21

It is very simple to alter any of the figures. Suppose you want to increase the February advertising figure. All you have to do is press F5 (go to a cell) and type the cell reference feb.adv The cursor will move to that cell and you can type a new value. Remember that the sales figures were calculated by a formula which assumed a 2% increase each month. If you change one of these cells to a numeric value you will destroy the formula in that cell. The formulae in the other cells of the row will, however, be unchanged. The amounts in the following cells will still increase by 2% per month, starting from the new value.

5.2 A SIMPLE BAR CHART

This is a short example to produce a simple graphic representation of a set of figures in bar chart form. You have seen most of the formulae before, so you will not need too many comments. This shows the appearance of the chart with a few numbers added.



```
HELP           CURSOR        DATA & FORMULA    TEXT type "     COMMANDS
press F1       press ←↑↓→                                      press F3
PROMPTS        GOTO  CELL    enter directly    followed by     ESCAPE
press F2       press F5      & press ←┘        text & ←┘       press ESC
        A      |B|    C     |     D     |     E     |     F     |     G     |
 1                 Simple Bar Chart
 2                 ================
 3
 4Values        |
 5==========================================================
 6            10 | **********
 7         17.35 | *****************
 8            17 | *****************
 9             3 | ***
10            27 | ***************************
11            38 | **************************************
12             0 |
13            15 | ***************
14            23 | ***********************
15             8 | ********
?█

CELL A1    GRID USED A1:F15                         DIRECTION ↑
CONTENTS EMPTY
```

Remember to clear the grid with the Zap command before starting to type it in.
[C1] "SIMPLE BAR CHART
[C2] rept("=",len(c1))

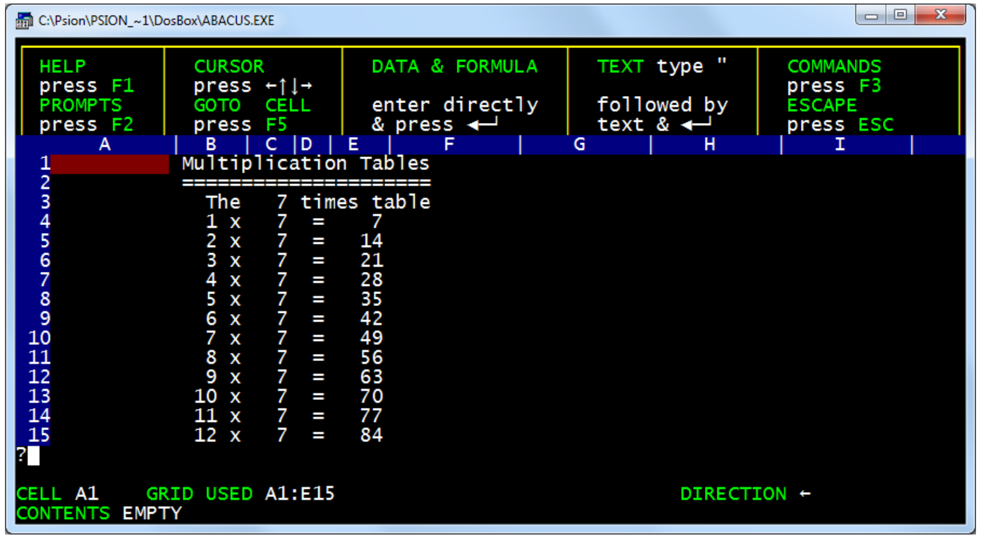
[A4] "Values
[B4] col="!" (rows 4 to 15)
[A5] row=rept("=",width()+1) (columns A to F)
[C6] col=rept(" *",a6) (rows 6 to 15)
These cells in column C are set to display a row of asterisks. The number of asterisks shown in each cell is governed by the value in the cell in column A of the same row. The cell reference (a6) in this formula is a relative one. Look at the formulae in the cells of column C. The cell reference in each one is different - it has been adjusted to refer to the correct row of column A. ABACUS treats all cell references like this unless you specify otherwise. How to use different types of cell references is described in the Multiplication Table example. We might as well reduce the width of column B to one character, since it is only dividing the values from their display. Press [F3] and use the command:

Grid >Width, 1, FROM b TO b

To use this bar chart, all you have to do is to put numbers in the cells of column A, between A6 and A15 inclusive. You will find that the rept() function will accept a negative value for the number of repetitions, but ignores the minus sign. Of course, if you use too large a number you won't see the end of the bar, as it will disappear out of the end of the window! You should not use a number greater than 255, which is the largest number of repetitions that rept() will accept.

5.3 MULTIPLICATION TABLES

This simple example may prove useful to a child who wants to learn the multiplication tables. It lets you request a particular table and then displays it. When you have typed in the example you use it by forcing a recalculation of the grid with the Xecute command, ie you type:
[F3] X ABACUS then asks you to type in a number (don't forget to press [Enter]) and displays the corresponding multiplication table.



```
C:\Psion\PSION_~1\DosBox\ABACUS.EXE

HELP       CURSOR      DATA & FORMULA   TEXT type "    COMMANDS
press F1   press ←↑↓→                                  press F3
PROMPTS    GOTO  CELL  enter directly   followed by    ESCAPE
press F2   press F5    & press ↵        text & ↵       press ESC
     A      | B | C |D | E      F     |    G    |    H    |    I    |
 1          Multiplication Tables
 2          =====================
 3            The    7 times table
 4            1 x    7  =    7
 5            2 x    7  =   14
 6            3 x    7  =   21
 7            4 x    7  =   28
 8            5 x    7  =   35
 9            6 x    7  =   42
10            7 x    7  =   49
11            8 x    7  =   56
12            9 x    7  =   63
13           10 x    7  =   70
14           11 x    7  =   77
15           12 x    7  =   84
?

CELL A1    GRID USED A1:E15                      DIRECTION ←
CONTENTS EMPTY
```

First title the application as normal:
[B1] "MULTIPLICATION TABLES
[B2] rept("=",len(b1))
The next three lines give a heading to the table.
[B3] "The
[C3] askn("Which multiplication table do you want")
[D3] "times table Here we have used the askn() function to request input; it allows you to choose which table you want, by typing in a number. This function takes a text string as its argument and displays the text in the input line, followed by a question mark. It then waits for you to type in a number, ending with . The number that you type in will be displayed in the cell (C3 in this case) which contains askn(). Note that askn() will not wait for input during a normal auto-calculation of the grid. It will only display the message and request input when you first put the formula into the cell, or when you force a recalculation of the grid by using the Xecute command. Once you have input a value to the cell it will be retained until the next time you force a recalculation with the Xecute command.
The remaining grid entries use the column-filling facility to produce the body of the multiplication table.
[B4] col=str(row()-3,2,0)+" x" (rows 4 to 15) This is the most complicated formula of the example. It is used to display the multiplier in each row of the table. The number is converted to a text string so that we can combine it with the multiplication sign and display them both in a single cell. The str() function converts a number to the equivalent string of digits. It takes three values; the number to be converted, a code for the format (0 = decimal, 1 = exponential, 2 = integer, 3 = general) in which the number is to be displayed, and the number of decimal places to be shown. In this case the number is converted to integer (whole number) format. In this case the value is obtained from the expression 'row()-3', whose value is 1 in row four, 2 in row five, and so on, up to 12 in row fifteen. The next value (2) selects display as an integer (whole number). The third number normally specifies how many decimal places are to be used. Its value must always be given but is ignored for integers which, by definition, have no fractional part. It has been given a value of zero (any other value could have been used). Finally the result is concatenated (the technical term for adding one text string to another) with the string " x", so that both the multiplier and the multiplication sign are displayed in a single cell.
[C4] col=$c3 (rows 4 to 15)
Column C contains copies of the value typed in in answer to the askn() function. The cell reference is preceded by a $ sign to make it an absolute cell reference. When you have entered the formula, move the cursor up and down the cells of column C and look at their contents. You will see that they all contain the cell reference $C3. The reference has not been adjusted in each row, unlike the example in the previous chapter. An absolute cell reference always refers to one particular cell, from any position in the grid. You can make any cell reference absolute by adding a leading $ sign.
[D4] col="=" (rows 4 to 15)
[E4] col=$c3 *(row()-3) (rows 4 to 15)

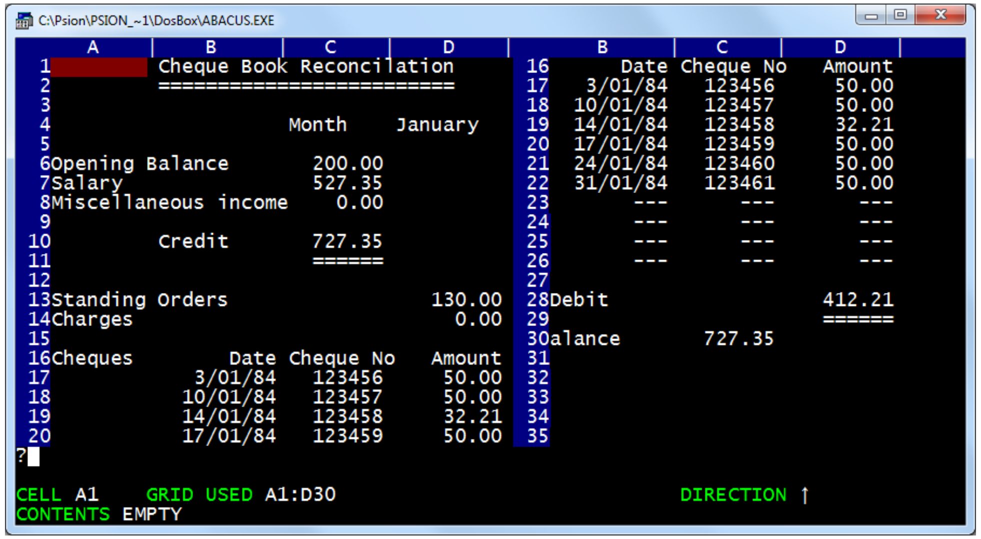
These last two column entries are almost self-explanatory. They are used to produce the equals sign and the answer for each row of the table. The last formula multiplies the value from the askn() function in cell C3 (another absolute cell reference) by the 'row()-3' expression which, as we saw earlier, gives a value of 1 in row four, 2 in row five, up to 12 in row fifteen. We now need to use a few commands to change the display of the table to a more convenient form. Use the following commands:

```
Justify,Cells,Text,Right,Range b3:b15
Justify,Cells,Text,Right,Range d4:d15
Justify,Cells,Numbers,Centre,Range c3
Grid >Width, 5 FROM b TO b
Grid >Width, 3 FROM c TO c
Grid >Width, 2 FROM d TO d
Grid >Width, 4 FROM e TO e
```

You use the table by forcing a recalculation of the grid with the Xecute command. You are prompted for input - the text of the askn() function will appear in the input line - and should type in a number.

5.4 CHEQUE BOOK RECONCILIATION

This example allows you to keep a check on your bank account. You enter details of your cheques in the spaces provided. At the end of the month you add the details of your salary, standing orders etc, by use of the Xecute command. You are then provided with a balance which you can compare with your bank statements.



```
       A          B          C          D                B          C          D
 1          Cheque Book Reconciliation         16        Date  Cheque No    Amount
 2          ==========================         17      3/01/84    123456     50.00
 3                                             18     10/01/84    123457     50.00
 4                     Month     January       19     14/01/84    123458     32.21
 5                                             20     17/01/84    123459     50.00
 6Opening Balance        200.00                21     24/01/84    123460     50.00
 7Salary                 527.35                22     31/01/84    123461     50.00
 8Miscellaneous income     0.00                23          ---       ---       ---
 9                                             24          ---       ---       ---
10          Credit       727.35                25          ---       ---       ---
11                       ======                26          ---       ---       ---
12                                             27
13Standing Orders                  130.00      28Debit                      412.21
14Charges                            0.00      29                           ======
15                                             30alance          727.35
16Cheques          Date Cheque No  Amount      31
17              3/01/84   123456   50.00      32
18             10/01/84   123457   50.00      33
19             14/01/84   123458   32.21      34
20             17/01/84   123459   50.00      35
?
CELL A1    GRID USED A1:D30                         DIRECTION ↑
CONTENTS EMPTY
```

[B1] "CHEQUE BOOK RECONCILIATION
[B2] rept("=",len(b1))
[C4] "Month

[D4] askt("Enter month") The askt() function works in the same way as askn() , but the expected input is text instead of a number. When you use Xecute ABACUS will display the message in the input line and then wait for you to type in some text. You should type in the name of the month for your balance. In this example we have used January.

[A6] "Opening balance
[A7] "Salary
[A8] "Miscellaneous income

Now we want ABACUS to ask for the amount of the opening balance for the month. We use askn() , but this time we can construct the text for the prompt by combining the text from cell A6, D4 and the literal text " for " (the spaces are important). The complete prompt will therefore be "Opening balance for January".

[C6] askn(a6+" for "+$d4) Note that the first cell reference is a relative one and the second is absolute. The reason is that we can now add the prompts for the remaining two entries very simply. Use the Echo command to copy the formula from cell C6 into cells C7 and C8. Instead of typing the range reference C7:C8, we can use the range identifier col. Echo,cell c6,over range col (rows 7 to 8)

[B10] "CREDIT
[C10] sum(col) (rows 6 to 8) Cell C10 is used to contain the total of all credits for the month. This cell is labelled; its reference is 'credit.month'. The cell's contents are calculated using the sum() function which we met in the first example in this chapter. This function adds the numeric contents of all cells in the range specified by its argument. Remember that it ignores any cell in the range that is empty or that contains text. In this case we have again used it as sum(col) , which specifies that the cells to be summed lie in the current column. As normal, ABACUS asks you to specify the exact range, suggesting reasonable values based on your previous work.

[C11] rept("=",len(str(credit.month,0,2))) Cell C11 underlines the total, using the usual rept() and len() functions. In this case, however, the formula is more complicated because ABACUS has to calculate how many characters to underline. Suppose cell C10 (credit.month) contains the number 100.00. If we simply used the formula rept("=",credit.month) the underlining would be 100 characters long, instead of the six characters we need. We therefore have to convert the number to a string of characters with the str() function (which, in this case, assumes that it is to be shown in decimal format, with two decimal places). The length of this string gives the correct number of characters to underline. Note that, as yet, the underlining does not line up with the numbers. This is because the underlining is text and is therefore left justified. It will be corrected later when we modify the justification.

[A13] "Standing orders

[A14] "Charges
[D13] askn(a13+" for "+$d4)
[D14] askn(a14+" for "+$d4)   These allow you to enter the monthly debits in response to prompts, using askn() in the same way as described earlier.
[A16] "CHEQUES
[B16] "Date
[C16] "Cheque no
[D16] "Amount
[B17] row="---" (columns B to D) These cells set up an area of the grid which you will later use to enter the details of your cheques.
[B28] "DEBIT
[D28] sum(col) (rows 13 to 26) This calculates the total of the debits. Remember that sum() only adds numeric values in the cells of the specified range. Cells containing text, and empty cells, are not included. The sum will therefore ignore all unused entries in the list of cheques, as well as the table heading in column D.
[A30] "Closing balance
[C30] credit.month - debit.amount   The calculation of the closing balance completes the grid entries.

You should now use the commands to tidy up the appearance of the application. First we can use the Echo command to fill the rest of the cheque table and complete the underlining of the totals. This command makes copies of the contents of a single cell into all the cells in a range. The first of the following three uses, for example, copies the contents of cell B17 into all the cells in a rectangle whose top left and bottom right corners are B18 and D26 respectively.
Echo,Cell b17,over range b18:d26
Echo,Cell c11,over range d29
Echo,Cell c11,over range c31

Next we need to set the numeric display to decimal, with two decimal places, for the whole of the application, with integer format for the cheque numbers:

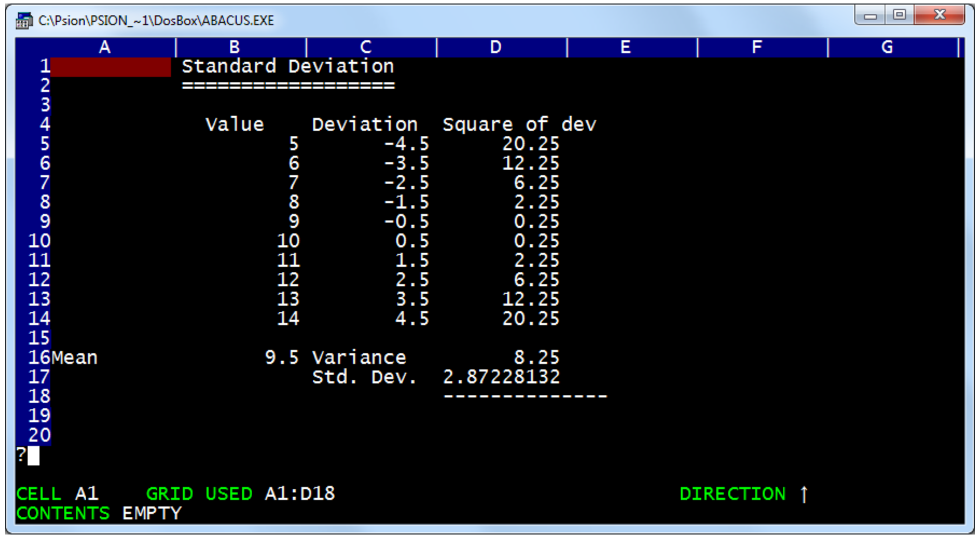
Units,Cells,Decimal,Decimal places 2,Range a1:d30
Units,Cells,Integer,Minus sign,Range c17:c26 In the Grid Cells chapter we explained that empty cells do not exist as far as ABACUS is concerned and they do not take up any space in memory. The change to decimal format (or to any other format) will therefore only affect non-empty cells. We fill the cheque table with "---" before making the change to decimal format so that the change can affect them. An alternative method is to change the default format, as described in the Grid Cells chapter. Finally we can modify the justification of the text, including the underlining, to improve the final appearance.

Justify,Cells,Text,Right,Range b16:d26
Justify,Cells,Text,Right,Range c11
Justify,Cells,Text,Right,Range d29
Justify,Cells,Text,Right,Range c31

The part of the grid that is used is too large for it all to be visible in the window at once. In order to see the final results, together with the values entered via the askt() and askn() functions, you might like to use the split window facility. The Window command splits the window, either vertically or horizontally, into two windows. It uses the position of the cursor in the window to determine the position of the split. A vertical split is most suitable for this grid and you can set it up by moving the cursor to the centre of the window and then using the command: Window,Vertical,Separate The split will occur at the position of the cell containing the cursor (so that you can divide the screen in any proportion that you want). You can move the cursor from one window to the other by pressing F4 . For this example you should use the cursor to adjust the left hand window to show cell A1 at its top left corner, and cell B15 at the top left corner of the right hand window.

5.5 STANDARD DEVIATION

This example calculates the mean and standard deviation of a set of numbers. It makes use of the labelling facilities of ABACUS so that the formulae used in the calculations are mostly self-explanatory



```
C:\Psion\PSION_~1\DosBox\ABACUS.EXE
         A         B         C         D         E         F         G
  1           Standard Deviation
  2           ==================
  3
  4             Value    Deviation  Square of dev
  5                 5       -4.5      20.25
  6                 6       -3.5      12.25
  7                 7       -2.5       6.25
  8                 8       -1.5       2.25
  9                 9       -0.5       0.25
 10                10        0.5       0.25
 11                11        1.5       2.25
 12                12        2.5       6.25
 13                13        3.5      12.25
 14                14        4.5      20.25
 15
 16Mean           9.5 Variance          8.25
 17                   Std. Dev.  2.87228132
 18                              --------------
 19
 20
?
CELL A1    GRID USED A1:D18                              DIRECTION ↑
CONTENTS EMPTY
```

In addition it uses a grid layout which requires calculation in column order, rather than the normal row order. In general, a formula should only refer to cells that are in the region above and to the left of the cell containing the formula - including the row and column containing the formula. If you do not follow this rule, as in this example, it is likely that the results may be incorrect. In most cases you can obtain a correct result by forcing a recalculation of the grid with the Xecute command or, as in this case, calculating the grid in column order.

[B1] "STANDARD DEVIATION
[B2] rept("=",len(b1))
[B4] "Value

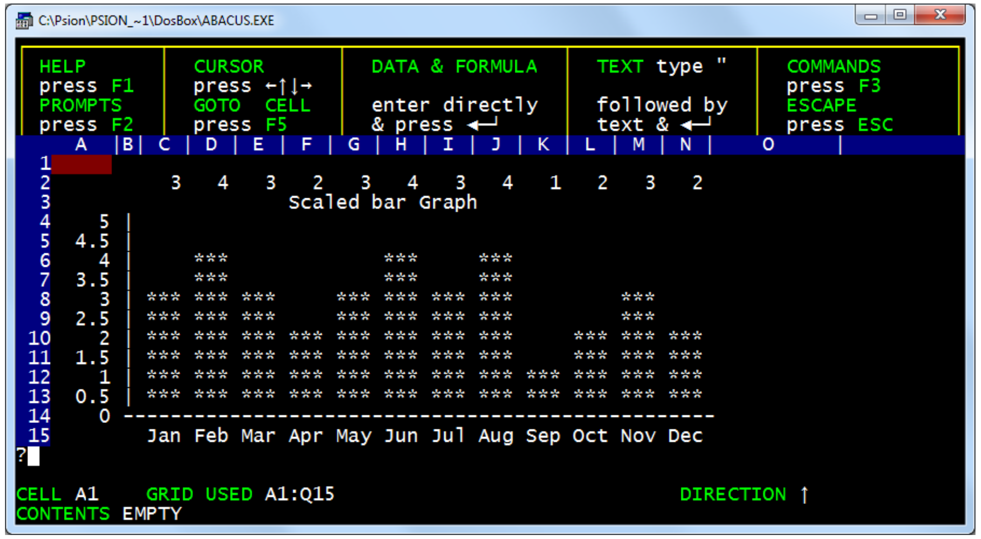
[C4] "Deviation
[D4] "Square of dev.
[B5] col=row() (rows 5 to 14) This last formula inserts a set of dummy values in the cells of column B for testing the application. When the grid entries are complete you can replace them with other values. The table described in this example will only hold ten values - you can change this to cope with more if you want.

[A16] "Mean
[B16] ave(value) (rows 5 to 14)
deviation=value-$mean.value (rows 5 to 14)
square=dev *dev (rows 5 to 14)
[C16] "Variance
[D16] ave(square) (rows 5 to 14) These formulae show that the variance of a set of numbers is defined as the average of the squares of the deviations from the mean,
[C17] "Std. Dev.
[D17] sqr(variance) (columns d to d) and that the standard deviation can be calculated as the square root of the variance.
[D18] rept("-",len(str(std.sq,3,0)))

The numbers in this example are left in general format so that it can handle any range of values. The underlining therefore uses the length of the text string corresponding to the number in the cell above (with cell reference 'std.sq') expressed in general format. You can improve the appearance of the display by changing to centre justification for the text in the range B4:D4, and using left justified numbers in the range B16:D17. If you try using this example by putting different values in the cells of column B, you will find that it does not give the correct answers. The reason is that the recalculation of the grid is performed row by row, from the top downwards. Any alteration you make will therefore be worked out on the basis of an incorrect mean value (since the new mean will not be calculated until after the deviations from the mean). The solution is to make the recalculation of the grid be in column order, from left to right. You do this with the Design command. Try it now, using the C option to change to column order. Leave the command by pressing , as indicated in the control area. When you next change a value in column B, the calculation will be correct, since the new mean is now calculated before the deviations. Although this ability to change the order of calculation is very useful, you should not get into the habit of using it too often - calculating in column order is much slower than row order. If you save a grid to a disk file, the current settings of all the Design options are saved with it. They are used to set these options whenever you reload the file, so you do not have to change them yourself each time you use it. All you need to do is to set the options to the values you want before saving your application with the Save command.

5.6 AN AUTO-SCALING BAR CHART

This example presents a more sophisticated bar chart display than that produced by the Simple Bar Chart example of this chapter. In addition to extending the use of absolute and relative cell references, it introduces several new functions. The chart displays twelve values, labelled by month. The values are read from twelve cells above the chart. The vertical scale is adjusted automatically to make sure that all values will fit the display. It is only suited to displaying positive values



```
 HELP        CURSOR        DATA & FORMULA     TEXT type "     COMMANDS
 press F1    press ←↑↓→                                       press F3
 PROMPTS     GOTO  CELL    enter directly     followed by     ESCAPE
 press F2    press F5      & press ←┘         text & ←┘       press ESC
      A   |B| C | D | E | F | G | H | I | J | K | L | M | N |    O    |
 1
 2              3   4   3   2   3   4   3   4   1   2   3   2
 3                      Scaled bar Graph
 4     5 |
 5   4.5 |
 6     4 |     ***             ***     ***
 7   3.5 |     ***             ***     ***
 8     3 | *** *** ***     *** *** *** ***         ***
 9   2.5 | *** *** ***     *** *** *** ***         ***
10     2 | *** *** *** *** *** *** *** ***     *** *** ***
11   1.5 | *** *** *** *** *** *** *** ***     *** *** ***
12     1 | *** *** *** *** *** *** *** *** *** *** *** ***
13   0.5 | *** *** *** *** *** *** *** *** *** *** *** ***
14     0 ------------------------------------------------
15         Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
?█

CELL A1    GRID USED A1:Q15                         DIRECTION ↑
CONTENTS EMPTY
```

First you should set the column widths to five in column A, one in column B and three in columns C to N, using the Width option of the Grid command.

[C2] row=0 (columns C to N) Row two will contain the values to be displayed - for the moment it is filled with zeroes, as dummy values, later to be replaced by real data.
[F3] "SCALED BAR GRAPH
[P2] int(max(c2:n2)/5+1)*5
[Q2] int(min(c2:n2)/5)*5

Cells P2 and Q2 contain the maximum and minimum values for the vertical scale of the graph. These cells are chosen so that they do not appear in the final display of the chart. Their initial values (when all the displayed numbers are zero) are five and zero respectively. The max() function finds the maximum, or largest, numerical value in the range of cells specified by its argument. Similarly, the min() function finds the minimum, or smallest, value in the range. Let us first examine the formula in cell Q2. The min() function finds the minimum, or smallest, value in the specified range and this is then divided by five. The int() function then removes the fractional part of the result of the division. If, for example, the minimum value is 13, dividing by 5 gives a value 2.6, and int(2.6) is 2. When this is multiplied by 5 we end up with a value of 10, which is the largest multiple of 5 that is less than the minimum. The formula in cell P2 is similar, except that it finds the largest value in the range and adds 1 to the number before the final multiplication by 5. If, as an example, we assume that the maximum value is 21, you can verify that the formula will give a value of 25 - the smallest multiple of 5 that is greater than the maximum. The two values in these cells will therefore always bracket the values in the cells from C2 to N2. Their difference is always a multiple of five. The next formula displays the vertical scale of the graph in column A.

[A4] col=$q2+(14-row()) *($p2-$q2)/10 (rows 4 to 14) The interval between successive numbers in the scale is (P2-Q2)/10. Note that we made the difference between the contents of P2 and Q2 a multiple of five so that this interval always has a simple value. This interval is multiplied by a number (14-row()) which starts at zero in row fourteen and increases by steps of one to a value of ten in row four. The result is added to the smallest value, from cell Q2, to produce the number for each cell. The net result is that the value in cell Q2 is displayed in A14, the value from P2 is displayed in A4 and the intervening cells contain a set of equally spaced values between these two limits.

[B4] col="!" (rows 4 to 14)
[B14] row=rept("-",width()+1) (columns B to N)
[C15] row=month(col()-2) (to 3) (columns C to N) These draw the axes for the chart and add the horizontal axis labels, using the months of the year. Note that we have used the string slicing operator to display only the first three characters of each month.

[C4] if(index(1,row()) > index(col(),2) ,"","***") This is the formula that does all the work of producing the bars themselves. It must be copied into every cell in the display area: Echo, Cell c4,over range c4:n13 The formula itself needs some explanation. It uses the if() function to decide whether to display part of a bar. The if() function takes three arguments. The first is an expression which must give a numeric result. If this result is non-zero the cell displays the second argument, which may be text or numeric. If, however, the result is zero the third argument is displayed in the cell. Again this may be text or numeric. In each cell the formula compares the number in column one of that row (the value labelling the vertical axis) with the number in row two of that column (the value to be displayed in the graph). If the axis label is greater than the display value, the condition is true (it evaluates to 1) and nothing is displayed. If the axis label is less than or equal to the display value, the condition results in a value of zero, and three asterisks are shown in the cell. The net result is that a bar is drawn to the correct height in each column. Since a single formula is used for all the cells in the display, the cell reference can be neither absolute nor relative. The reference to the display values must change as we move from column to column (ie it must be relative along a column) but must always refer to row two as we move down, from row to row. We need a form of cell reference which is relative with respect to

columns, but absolute with respect to rows. Fortunately the index() function can be used to produce this effect. It takes two parameters, a column number and then a row number, returning the contents of the specified cell. With this we can construct any combination of absolute and relative references, as the following examples show:

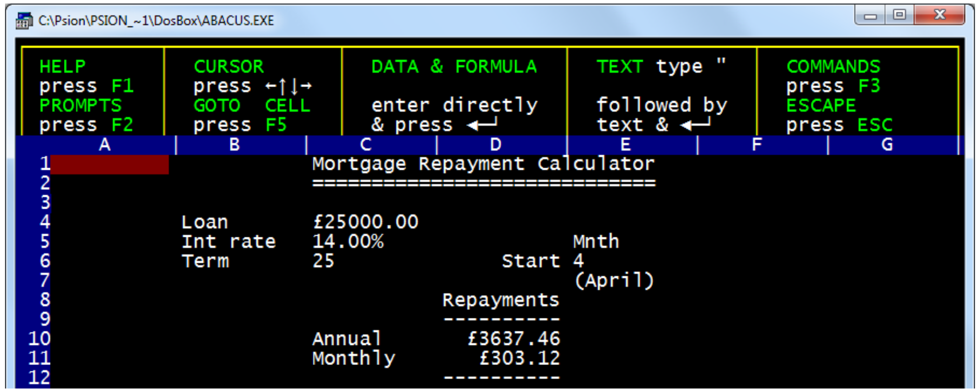
Function	Column Ref	Row Ref
index(5,5)	absolute	absolute
index(col(),5)	relative	absolute
index(5,row())	absolute	relative
index(col(),row())	relative	relative

The function index(col(),2) therefore returns the contents of the cell in row two of the current column, and index(1,row()) returns the contents of the cell in column one (A) of the current row. Try putting different values in cells C2 to N2 and see what effect they have on the display.

5.7 MORTGAGE CALCULATOR

This example enables you to calculate the monthly payments due on a repayment mortgage. You are asked to type the amount of the loan, the interest rate, the length of the loan in years and the month of the first payment. The required payments are calculated and displayed, together with a complete repayment table for the whole period of the loan. This table shows you the outstanding sum at the beginning of each month until the loan is repaid. Several of the calculations in the grid make use of values that are input by use of the askn() function.

5.7.1 Mortgage Repayment Calculations



```
HELP          CURSOR        DATA & FORMULA    TEXT type "      COMMANDS
press F1      press ←↑↓→                                       press F3
PROMPTS       GOTO  CELL    enter directly    followed by      ESCAPE
press F2      press F5      & press ←┘        text & ←┘        press ESC
        A            B             C          D           E           F           G
 1                          Mortgage Repayment Calculator
 2                          =============================
 3
 4            Loan          £25000.00
 5            Int rate      14.00%                        Mnth
 6            Term          25                      Start 4
 7                                                        (April)
 8                                     Repayments
 9                                     ----------
10                          Annual       £3637.46
11                          Monthly       £303.12
12                                     ----------
```

In this section we shall produce the part of the grid that accepts your input and calculates the monthly repayments. When you have typed in the formulae and added a few figures in response to the askn() functions it should look like this.

[C1] "MORTGAGE REPAYMENT CALCULATOR
[C2] rept("=",len(c1))
[B4] "Loan
[C4] askn("Amount of loan") The next three entries request the input of the interest rate. The original input is to cell (H4) well away from the displayed portion of the grid so that you do not normally see it. You type in a percentage value, e.g. you type 12 to mean 12%. The value needed by the rest of the formulae is a fractional value (e.g. 12% must be converted to 0.12) and this is calculated from the input value by the formula in cell C5.

[H4] askn("Percentage interest rate")
[B5] "Int rate
[C5] h4/100
[B6] "Term
[C6] askn("Period of loan in years [maximum 35]")
[E5] "Mnth
[D6] "Start
[E6] askn("Month of first payment [Jan1, Feb2, etc]")
[E7] '(' + month(e6) + ')' In this last formula we enclose the literal text with single quotation marks. If the first character had been a double quote, ABACUS would have interpreted the following characters as text input rather than a formula.
[D8] "REPAYMENTS
[D9] rept("-",len(d8))
[C10] "Annual
[D10] mor.loan*mor.int/(1-(1+mor.int)^(-mor.term)) This formula which calculates the annual repayment, assumes that the interest is calculated and added to the loan before the twelve monthly repayments are made.
[C11] "Monthly
[D11] ann.rep/12
[D12] d9

The grid is now sufficiently complete to calculate mortgage repayments. Try using the Xecute command and entering the figures requested, so that you can see it working. To make the example look better, we can change the format of some of the numbers with the Units command. In this example there is no need to alter the default numeric format (See Chapter 3) since you do not need to make new entries in any grid cell once the application is completed.

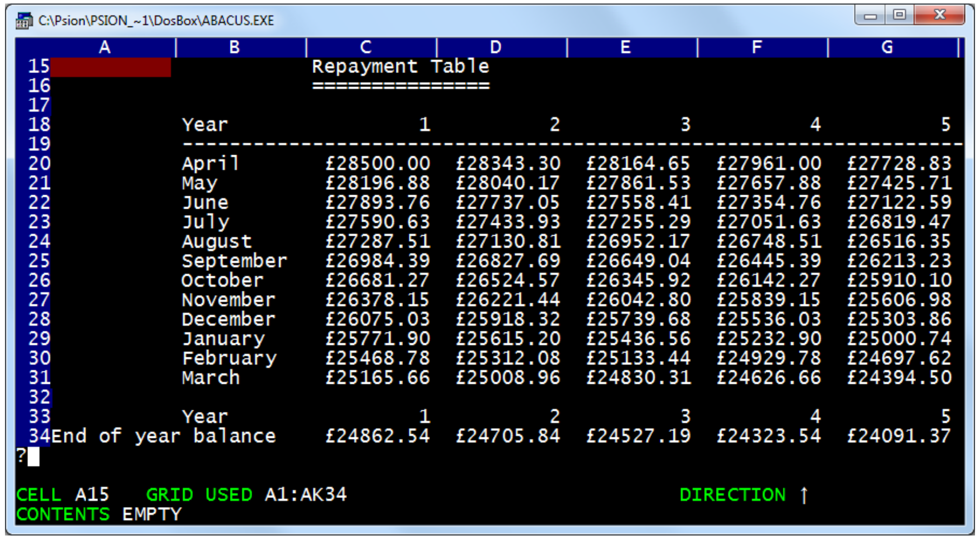
```
Units,Cells,Percent,Decimal paces 2,Range c5
Units,Cells,Monetary,Minus sign,Range c4
Units,Cells,Monetary,Minus sign,Range d10:d11
```

In addition it improves the appearance if we move the numbers in rows 4,5 and 6 to the left hand side of the cells:

Justify,Cells,Numeric,Left,Range c4:e6

5.7.2 Mortgage Repayment Table

This section describes how you can add a repayment table to the mortgage calculator. The first part of a repayment table for the values in section 5.7.1



```
         A        |    B     |    C     |    D     |    E     |    F     |    G     |
15                          Repayment Table
16                          ===============
17
18              Year                1          2          3          4          5
19              -----------------------------------------------------------------
20              April       £28500.00  £28343.30  £28164.65  £27961.00  £27728.83
21              May         £28196.88  £28040.17  £27861.53  £27657.88  £27425.71
22              June        £27893.76  £27737.05  £27558.41  £27354.76  £27122.59
23              July        £27590.63  £27433.93  £27255.29  £27051.63  £26819.47
24              August      £27287.51  £27130.81  £26952.17  £26748.51  £26516.35
25              September   £26984.39  £26827.69  £26649.04  £26445.39  £26213.23
26              October     £26681.27  £26524.57  £26345.92  £26142.27  £25910.10
27              November    £26378.15  £26221.44  £26042.80  £25839.15  £25606.98
28              December    £26075.03  £25918.32  £25739.68  £25536.03  £25303.86
29              January     £25771.90  £25615.20  £25436.56  £25232.90  £25000.74
30              February    £25468.78  £25312.08  £25133.44  £24929.78  £24697.62
31              March       £25165.66  £25008.96  £24830.31  £24626.66  £24394.50
32
33              Year                1          2          3          4          5
34End of year balance       £24862.54  £24705.84  £24527.19  £24323.54  £24091.37
?

CELL A15    GRID USED A1:AK34                              DIRECTION ↑
CONTENTS EMPTY
```

If you have a mortgage, type in your own figures. Don't spend too much time over the results for the first few years - they make rather depressing reading!
[C15] "REPAYMENT TABLE
[C16] rept("=",len(c15))
[B18] "Year
[C18] row=col()-2 (columns C to AK)
[B19] row=rept("-",width()+1) (columns B to AK)
[B20] col=month(row()-20+$mnth.start) (rows 20 to 31)

These entries set up the headers for the table: now we must add the formulae that will calculate the values. We start with the first item which is the initial amount due. It is calculated by adding the first year's interest to the amount of the loan.

[C20] mor.loan*(1+mor.int) Then the rest of the first row is calculated by subtracting the yearly payment and adding the interest for the current year. These values should not be calculated beyond the year in which the loan is repaid and we allow for this by using the if() function. If the year number (given by col()-2) is greater than the term of the mortgage, zero is placed in the cell.

[D20] row=if((col()-2)>$mor.term,0,(c20-$ann.rep)*(1+$mor.int)) - (columns D to AK)

The remainder of the table can be filled with a single formula. We fill the first cell with a formula which just subtracts the monthly repayment from the amount in the cell above. Again we use the if() function to prevent the calculations extending beyond the year in which the loan is repaid.

[C21] if((col()-2)>$mor.term,0,c20-$mon.rep)

You can then use the Echo command to copy the formula from cell C21 to the range D21:AK31. [F3] Echo, Cell c21,over range c21:ak31

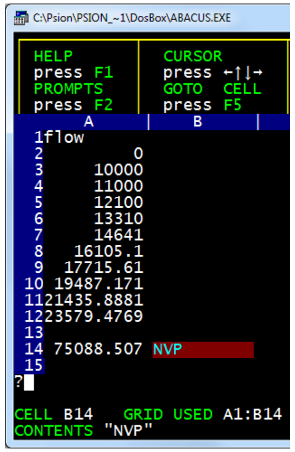
We can now complete the table by adding a final row to give the outstanding balance at the end of each year. It is probably a good idea to add a copy of the year, from row 18, for easy reference.

[B33] row=year.term (columns B to AK)
[A34] "End of year balance
[C34] row=if((col()-2)>$mor.term,0,c31-$mon.rep) (columns C to AK)

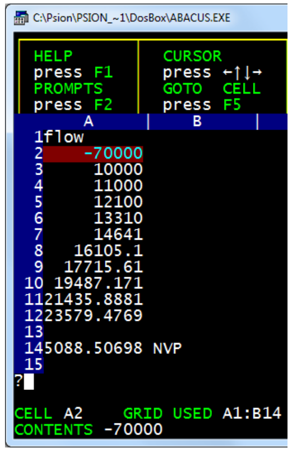
The entire table, and the end of year balances should be set to either monetary format or to decimal format with two places of decimals. The ranges for these changes are C20:AK31 and C34:AK34 respectively.

5.8 NPV() and IRR() The Net Present Value function reduces a series of cash flows, forecast to occur over a period of time, into a single net value now. The technique used is called discounting. A sum of £1.00 invested now at an interest rate of 10% would grow to £1.10 in one year's time. Conversely, £1.10 one year from now has a present value of £1.00, assuming an investment rate - or more accurately in this case a discount rate - of 10%. We can say that £1.00 now is the equivalent of £1.10 in one year's time. The npv() function discounts each of a series of future cash flows to its equivalent present day value and adds them together. The result is the net present day equivalent value of the future series of cash flows. This technique is widely used as an aid in making decisions regarding expenditure proposals. It is best illustrated by means of an example. Suppose you are given the opportunity to buy, for a single payment of seventy thousand pounds, a ten-year lease on a shop which is currently producing a yearly net income of ten thousand pounds. You expect the income to increase by 10% per year. The prevailing interest rate is 14% so, if you did not buy the shop, your seventy thousand pounds could earn 14% interest. What should you do? You should calculate the net present value of the projected income and compare it with the sum you are asked to pay:

[A1] "flow [A2] 0
[A3] 10000
[A4] col=a3*1.1 (rows 4 to 12)
[A14] npv(flow,12,14) (rows 2 to 12)
[B14] "NPV


```
C:\Psion\PSION_~1\DosBox\ABACUS.EXE
 HELP           CURSOR
 press F1       press ←↑↓→
 PROMPTS        GOTO  CELL
 press F2       press F5
        A      |     B      |
1flow
2          0
3      10000
4      11000
5      12100
6      13310
7      14641
8    16105.1
9   17715.61
10 19487.171
1121435.8881
1223579.4769
13
14 75088.507 NVP
15
?
CELL B14    GRID USED A1:B14
CONTENTS "NVP"
```

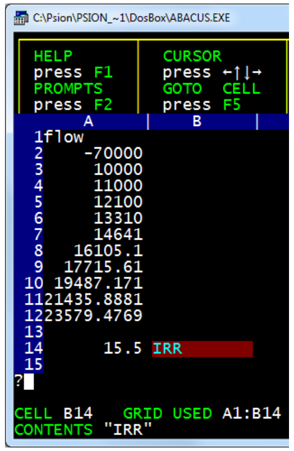
The npv() function requires three arguments. The first is a reference to a range of cells - in this case we use the column label "flow". The second is the number of months in each period and the third is the annual percentage discount rate. The net present value (in cell A14) of the cash flow from the shop is more than the asking price. This indicates that it would be more profitable to buy the shop than to invest your money at 14%. Note that the annual interest is assumed to be a flat rate, rather than being compounded, so a 12% interest rate with periods of length 3 months would be interpreted as 3% per quarter. The first item in the list is for period zero and is not discounted, the next is for period one and is discounted by one period, and so on. This is consistent with the assumption, made by the function, that the returns are received at the end of each period. You therefore have to wait for one period before you obtain any return on your investment. In a real situation of this type you would probably work on a quarterly or monthly basis, rather than with yearly periods. An alternative way to tackle the problem is to include your payment as a (negative) cash flow for the period zero cash flow. The only difference between this example and the previous one is:
[A2] -70000
In this form a positive result (in cell A14) shows that the deal would be profitable.


```
C:\Psion\PSION_~1\DosBox\ABACUS.EXE
 HELP           CURSOR
 press F1       press ←↑↓→
 PROMPTS        GOTO  CELL
 press F2       press F5
        A      |     B      |
1flow
2     -70000
3      10000
4      11000
5      12100
6      13310
7      14641
8    16105.1
9   17715.61
10 19487.171
1121435.8881
1223579.4769
13
145088.50698 NVP
15
?
CELL A2     GRID USED A1:B14
CONTENTS -70000
```

Net Present Value - Method 2. The irr() - internal rate of return - function calculates the investment (discount) rate necessary to reduce the NPV of a series of present and future cash flows to zero. This gives you a measure of the expected rate of return on your investment in a project, as opposed to the npv() function which gives you a measure of the projected absolute cash return. Thus irr() is particularly useful to distinguish between two cash flows yielding identical NPV's but requiring different levels of investment. Again, the function is best explained by means of an example. We can continue the example and calculate the IRR of the series of cash flows. We can again refer to the range of the data by the label "flow" - and the interval between successive periods is twelve months:

[A15] irr(flow,12) (rows 2 to 12)
[B15] "IRR

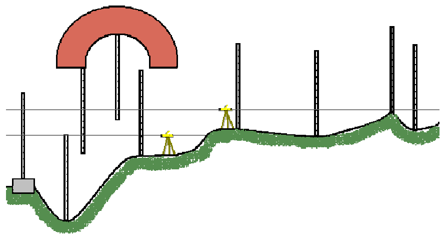
The completed grid shows that the internal rate of return is 15.5%. If you can invest your seventy thousand pounds at a higher rate of interest it would be more profitable to do so, rather than buy the shop.

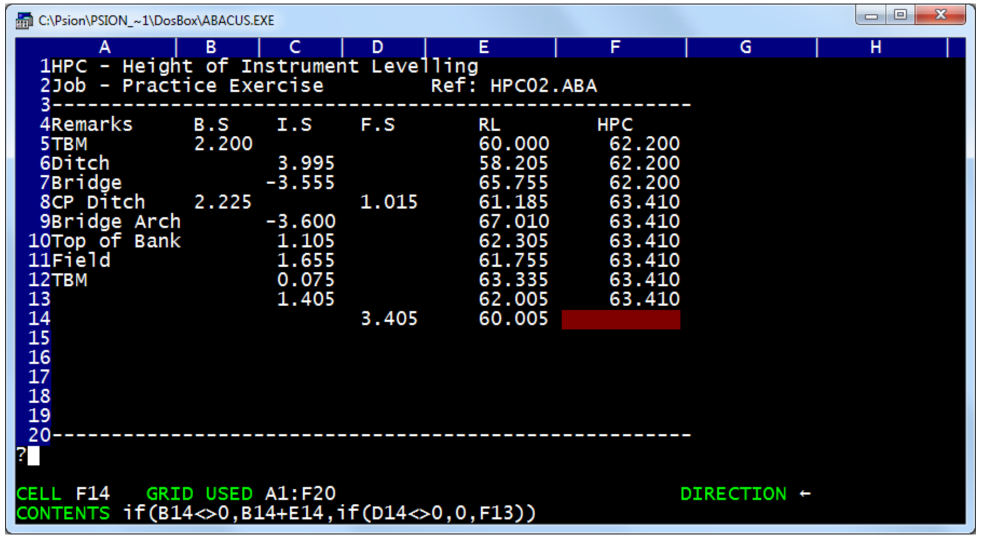

```
C:\Psion\PSION_~1\DosBox\ABACUS.EXE
 HELP           CURSOR
 press F1       press ←↑↓→
 PROMPTS        GOTO  CELL
 press F2       press F5
        A      |     B      |
1flow
2     -70000
3      10000
4      11000
5      12100
6      13310
7      14641
8    16105.1
9   17715.61
10 19487.171
1121435.8881
1223579.4769
13
14      15.5 IRR
15
?
CELL B14    GRID USED A1:B14
CONTENTS "IRR"
```

5.9 Optical Levelling HPC Spreadsheet

Builders and land surveyors using optical levelling instruments calculate the level at certain points by taking staff readings at those points and calculating the level. This calculated level is known as a 'Reduced Level'. A spreadsheet can automate these calculations.

[F3]Units Defaults Decimal decimal places 3
[A1]"HPC - Height of Instrument Levelling
[A2]"Job - Practice Exercise
[E2]"Ref: HPC02.ABA
[A3]row=(rept("=",width()+1))   A to F
[A4]"Remarks
[B4]"B.S
[C4]"I.S
[D4]"F.S
[E4]"R.L
[F4]"HPC   [F3] Justify Cells Centre B4:F4
[E6]IF(C6<>0,F5-C5,IF(D6<>0,F5-D6,0))   -   [F3] Echo E6 - E7:E19
[F5]IF(B5<>0,B5+E5,IF(D5<>0,0,0))
[F6]IF(B6<>0,B6+E6.IF(D6<>0,0,F5))   -   [F3] Echo F6 - F7:F19
[A20]A3 from A to F   -   [F3] save HPCblank
Enter details as below -Remarks then reading at BS, IS & FS




C:\Psion\PSION_~1\DosBox\ABACUS.EXE

```
     A        B       C       D         E         F          G          H
 1HPC - Height of Instrument Levelling
 2Job - Practice Exercise           Ref: HPC02.ABA
 3------------------------------------------------------------
 4Remarks     B.S     I.S     F.S       RL        HPC
 5TBM         2.200                     60.000    62.200
 6Ditch               3.995             58.205    62.200
 7Bridge             -3.555             65.755    62.200
 8CP Ditch    2.225           1.015     61.185    63.410
 9Bridge Arch        -3.600             67.010    63.410
10Top of Bank         1.105             62.305    63.410
11Field               1.655             61.755    63.410
12TBM                 0.075             63.335    63.410
13                    1.405             62.005    63.410
14                            3.405     60.005
15
16
17
18
19
20------------------------------------------------------------
?
CELL  F14    GRID USED A1:F20                            DIRECTION ←
CONTENTS  if(B14<>0,B14+E14,if(D14<>0,0,F13))
```

Used extensively in the 80's & 90's by construction students on their surveying camp. Utilising the PSION Organiser II for data logging and Sinclair QL Abacus for calculating the RL's and printing.

ABACUS REFERENCE 6.0

THE FUNCTION KEYS 6.1

ABACUS uses the function keys shown in the following list.

Key	Plus	Action
[F1]		To obtain Help
[F2]		Turn the prompts on and off
[F3]		Call the command menu
[F4]		to move from one split window to another
[F5]		goto a particular cell reference
[F6]		Freeze the task return to Xchange - Not used in PC FOUR
[F4]	CTRL	Move the windows, if using the Pointer Environment with extended resolution - Not used in PC FOUR

6.2 CURSOR MOVEMENT The cell cursor is moved by using the four arrow keys. Alternatively, you can press F5 followed by a cell reference to go directly to that cell.

6.3 DATA ENTRY Numbers and formulae are typed in directly. Text must be preceded by quotation marks("). There are two methods of entering data into a cell.
6.3.1 Using the [Enter] key. This places the value in the cell and leaves the cursor on that cell.
6.3.2 Using the [Tab] key. This places the value in the cell and moves the cursor in the direction of the arrow shown in the status area. The direction of the cell cursor is set by its last movement. You can refer to single cells, rows, columns or ranges either by using explicit letter and number references or by using text labels.

6.4 SINGLE CELLS A reference to a single cell consists of two parts, a column and a row reference. There are 64 columns in the grid and they are labelled from A to BL. There are 255 rows, numbered from 1 to 255. Typical cell references are: A1 AC13 BD200

6.5 RANGE REFERENCES A range reference is made up of two cell references, separated by a colon. You must always type in the colon to separate the two parts of the reference. The first cell reference specifies the top left hand corner of the block and the second one identifies the bottom right hand corner. Examples of range references are: B5:D9.B AZ23:BA155

6.6 ROW AND COLUMN REFERENCES A part of a row or column can be considered as a range that is only one column wide (or one row deep). You

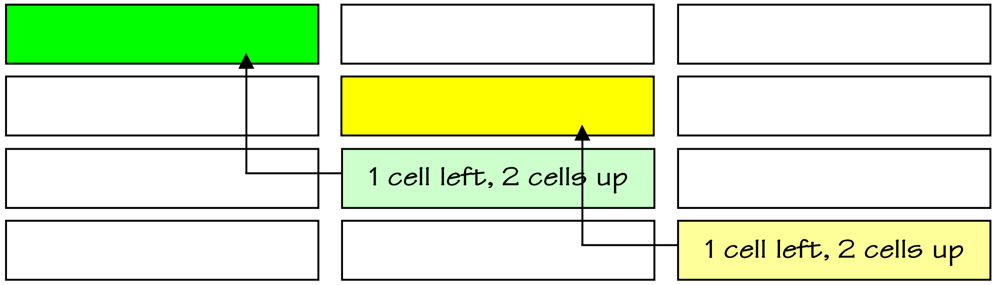
can therefore use a range reference to specify part of a row or column, such as:
A3:L3 (cells A to L of row 3) D7:D11 (cells 7 to 11 of column D)

6.7 RANGE IDENTIFIERS

There are two range identifiers: row and col. They refer to the cells of the current row or the current column respectively (those that intersect at the cell containing the range identifier). Each time you use one of them in a formula you will be asked to specify the exact range of cells within the row or column. ABACUS will suggest reasonable starting and ending points for the range and you can either accept this choice or change it. There are two ways in which you can use range identifiers. You can fill the current row or column by use of either row = (formula) col = (formula) You can also use them as the argument for any function that requires a range, for example, count(row). You can, of course, only use them in this way when you just want to refer to the cells of a single row or column. You can mix the two methods freely, for example, col = ave(row) Each occurrence in a formula will result in ABACUS asking you for a particular range.

6.8 RELATIVE AND ABSOLUTE CELL REFERENCES

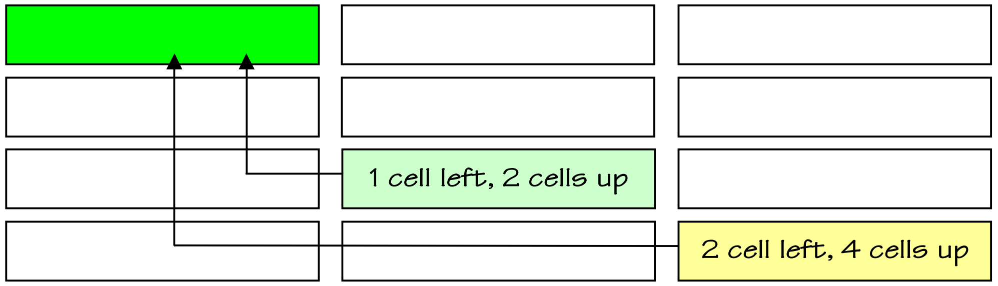
ABACUS normally assumes that all cell references are relative, ie that the difference in position between the cell containing the reference and the cell to which it refers is the important thing. When you copy such a reference into another cell, the references are modified to keep this relative difference. For example, imagine that a formula in cell B3 contains a reference to cell A1 (one column to the left and two rows above). If the formula in cell B3 is copied into cell C4 it will, in this new location, refer to cell B2 (again one column to the left and two rows above). This is illustrated in below. A formula in the light green cell contains a reference to the dark green cell. If this formula is copied to the light yellow cell Y it then refers to dark yellow cell. The two cells in each pair have the same relative positions.



Suppose we put the formula A1*2 into cell A2, and then use the echo command to copy the formula into cells in the range B2:G2. Examining the cells of row 2 will show that they have the following contents:

CELL:	A2	B2	C2	D2	E2	F2	G2
Contents:	A1*2	B1*2	C1*2	D1*2	E1*2	F1*2	G1*2

You can make any cell reference absolute by prefixing it with a $ sign. Such a reference will not be modified when the formula is copied to other cells. For example, if a reference in cell B2 was to $A1, any copy of the formula will also contain the reference $A1. You can also use labels to give an absolute cell reference (eg $march.costs).
A formula in the light green cell contains an absolute reference to the dark green cell. A copy of the formula in the light yellow cell refers to the same cell. Let us try the previous example, but this time we shall use an absolute reference.



Put the formula $A1*2 in cell A2 and echo it to cells B2 to G2 inclusive. You will then find that the cells contain the following:

CELL:	A2	B2	C2	D2	E2	F2	G2
Contents:	$A1*2	$A1*2	$A1*2	$A1*2	$A1*2	$A1*2	$A1*2

See also the index() function (which is used in the Auto-Scaling Bar Chart example of Chapter 5.6). Cell ranges in any form (including the range identifiers, row and col) are always relative.

6.9 LABELS

6.9.1 Row and Column Labels A label is a cell containing text. The text must only include letters and digits - other characters (eg '*') are not allowed. Any such cell can be used to identify a row or column in the grid. You can also use labels to refer to a single cell, but you may not use them to replace a range reference in order to refer to a rectangular block of cells. Whenever you refer to a label in an expression or formula, ABACUS uses a set of rules to determine whether it refers to a row, a column or a cell. The rules for rows and columns are: The row and column intersecting at the label are scanned (to the right and below) to find a numeric entry.

a) If only a row entry is found, the label refers to the row, starting at the found entry.
b) If only a column entry is found, the label refers to the column, starting at the found entry.
c) If entries are found in both the row and the column the entry closest to the labelled cell is used to make the choice.

If no decision can be made under the above rules, but the label is used on the left hand side of an expression, it will be given the type of any label(s) used on the right hand side. For example, if "Costs" is a row label: Sales = Costs * 0.5 then "Sales" will also be a row label. If both of these rules fail, you are told that ABACUS cannot decide the meaning of the label.

6.9.2 Cell Labels You need to use two labels to identify a single cell and you make the cell reference by giving both labels, separated by a full stop. For example, if you have two labels "fruit" and "apples", you can refer to a cell as fruit. apples (or by any unique abbreviation, such as "fr.ap"). The order of the two labels is unimportant so you could also use "apples.fruit", "ap.fr" and so on. Such a reference refers to the cell at the intersection of the rows and columns containing the labels.

6.10 FORMULAE

A formula is any allowed combination of functions, cell references, labels and arithmetic operators. Examples are: A1*B3
month(col()-1) if(instr(B6,"is"),1,0) rept("=",len(G23))+":"

6.10.1 Master Formulae Each new formula, in addition to being used in one or more grid cells, is stored separately in a list of master formulae. Each master formula may therefore appear in one cell or in many. When you fill cells by use of the row and column fill operations, or by using the copy or echo commands, all the filled cells share a single master formula. If a master formula contains relative cell references they are adjusted, in each cell using the formula, to be valid for that particular location. The formulae may therefore appear superficially different but are all based on the one master formula. There are several advantages to this approach, the most important being that less memory is used than if the formula in every cell is stored separately. This allows you to construct much more complex grids without running out of memory. A further advantage is that you can modify all copies of the formula by editing only one of the copies. If you use the amend command to change any copy of a master formula, the master is also modified and all copies are changed simultaneously. This is why you should use the row and column operations, and the echo and copy commands whenever possible, rather than filling each cell separately. One case, however, needs explanation to remove a possible source of confusion. When you use the grid command to insert or delete a row (or column) then any relative cell references in the formulae in the following cells are modified to take their new positions into account. ABACUS locates the first formula in the affected portion and uses this as a model to work out the necessary changes to the remaining formulae. All relevant master formulae are modified. If you have used copies of a master formula on both sides of the change to the grid it is possible, in some circumstances, for the formulae in cells before the position of the insertion or deletion to be made incorrect. This is unlikely to happen if you plan your grid correctly. If, however, you do find that this has happened, you can correct it by typing a replacement formula into the affected cells (so that they use a different master formula). You must not use the amend command, as this will simply change the master formula and make the formulae in later cells incorrect for their new positions.

6.10.2 String Slicing You may use a string slicing operation on any formula that results in a text value. This operation allows you to extract any sequence of one or more characters from the text. You may add one of the following string slicing operators at the end of any formula that has a text value.

(n)	select the nth character.
(n to m)	select all characters from the nth to mth character inclusive.
(n to)	select from character n to end.
(to m)	select from the beginning to the mth character.

For example, if the cell A1 contains the text "January":

A1 (to 3)	will show "Jan"
A1 (2 to 3)	will show "an"
A1 (5 to)	will show "ary"

6.11 THE COMMANDS

This section contains a full description of all the commands available in ABACUS. All commands are selected by pressing [F3] and then the first letter of the command.

6.11.1 AMEND Changes the contents of a cell. The contents of the cell containing the cursor are copied to the input line, ready for editing with the line editor. When you press [Enter] the edited version replaces the original cell contents. Remember that you will also edit the master formula, so that any change to a formula in a cell will cause a corresponding change in all other cells that share the same formula.

6.11.2 COPY Copies a range of cells from one area of the grid to a similar range in another place. Compare this with the echo command which copies the contents of a single cell to all the cells in a range. ABACUS first asks you to give the range reference of the cells to be copied, eg A1:B3, and you should then press [Enter] . ABACUS next asks you to specify the cell reference for the top left hand corner of the area to which the range of cells is to be copied. When you then press [Enter] the range will be copied to the new location. If any of the formulae that are being copied contain relative cell references, these references will be adjusted during the copy to be valid in the new area of the grid. This adjustment assumes that the reference is to another cell within the range of cells being copied. Any formula that refers to a cell outside the range being copied is likely to refer to the wrong cell from the new copy.

6.11.3 DESIGN Allows you to modify a number of the features of ABACUS that affect the working of the whole grid, and the appearance of the grid on printed output. The choices remain in force until you modify them again, or until you delete the ABACUS task in which they were set. When you save an application these choices are saved with it so that they are used every time you load the application. Changing the design options within an ABACUS task has no effect on the options in any other task. In the design command a list of options replaces the grid display. Press the key corresponding to the first letter of the option you require. You may then be asked to specify further information, which will be either a number or a single letter, as described in the following list. At the end of each default selection you are returned to the options list so that you can make further changes. When you have finished you return to the main display by pressing [Enter] . The options are:

AUTO-CALCULATE - on input used to specify auto-calculate or no auto-calculate. Each time you press the **A** key the auto-calculate option switches between YES and NO. If you choose YES, the whole spreadsheet will be recalculated after each entry. Selecting NO, however, means that the spreadsheet will only be recalculated when you use the **xecute** command. The initial value is YES.

BLANK if zero - switches between two ways of treating zero values in the grid. The original option is to display the value zero in the appropriate format for that cell. You may select the alternative, which is to display a blank cell if its contents evaluate to zero. Note that, in this option, a blank cell will only be shown if the value is truly zero. Suppose you have selected decimal display format, with two decimal places, and the value in such a cell is 0.003. The cell will show 0.00, rather than being blank, since the true value is non-zero.

CALCULATION order - selects between calculating the spreadsheet in ROW or COLumn order. The option changes each time you press the C key. The specified order will be used for both auto-calculate and the xecute command. The initial value is for row order.

FORM feed between pages - selects whether or not a form-feed is issued at the end of each page of printed output. The initial value is YES.

GAPS between lines on printer - sets the line spacing on printed output by specifying the number of gaps between the lines of text. You are asked to type in 0,1 or 2 (no [Enter] is necessary). You can set ordinary double-spaced printer output, for example, by specifying one gap between each line. The initial value is zero.

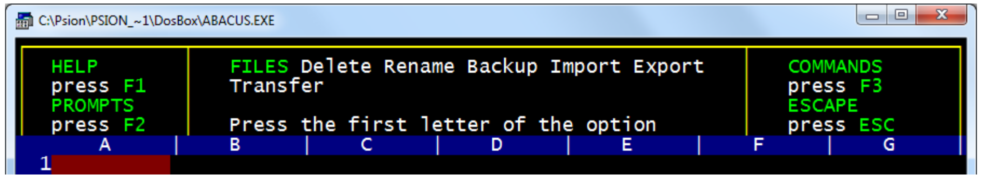
LINES per page - specifies how many lines on a page of printed output. You should type in a number, followed by [Enter] . The initial value is 66 and the maximum is 255.

MONETARY sign - specifies the currency sign to be used in the display of monetary values. You should type in the single character that you want (no [Enter] is necessary). The initial value is the pound sign.

PRINTER paper width - sets the number of characters per line of printed output. You should type in a number, followed by [Enter] . The initial value is 80 and the maximum is 255.

6.11.4 ECHO (E) Copies the data or formula in a particular cell to all the cells in a specified range. Compare this with the copy command, which copies the contents of all the cells in a specified range. This command gives you the option of specifying the cell reference of the cell to be copied, or of pressing [Enter] to copy the current cell. You then should type in the range over which the cell contents are to be copied, followed by [Enter] .

6.11.5 FILES Allows you to use a set of file options concerned with maintaining ABACUS files previously saved on disk.


```
C:\Psion\PSION_~1\DosBox\ABACUS.EXE

 HELP         FILES Delete Rename Backup Import Export    COMMANDS
 press F1     Transfer                                    press F3
 PROMPTS                                                  ESCAPE
 press F2     Press the first letter of the option        press ESC
        A    |    B    |    C    |    D    |    E    |    F    |    G
 1
```

Press the first letter of an option to select it. Whenever an option asks you for a file name you can press ? for a list of the files on a given drive. You are offered the following options:

Files> **D**elete - deletes a named file from a disk. Note that this command is not reversible and should therefore be used with great care.

Files> **E**xport - exports a named file. The file is saved in a form suitable for begin imported by ARCHIVE, EASEL or QUILL. There is a full description of Export in Appendix A. ABACUS first asks you whether you want to export to QUILL, ARCHIVE or EASEL. Accept the suggestion of export to QUILL by pressing [Enter] , or select export to ARCHIVE or EASEL by pressing either the A key or the E key. In all cases you are then asked to type in the range reference for the section of the grid that you want to export, ending your input by pressing [Enter]. If you have chosen to export to ARCHIVE or EASEL you can export the file by rows or by columns. ABACUS asks you to press [Enter] to accept the suggestion of exporting by rows, or to press the C key to choose export by columns. You are not given this option if you choose to export to QUILL. In this case the data is always exported by rows. ABACUS finally asks you to type in a name for the exported file. If you do not specify a file name extension ABACUS will supply an extension of .exp.

Files> **I**mport - imports a named file. It allows ABACUS to read files exported from Archive or Easel. There is a full description of import in Appendix A. You may import a file in either row or column order, and are asked to select which one. You are also asked for the cell reference of the top left hand corner of the area of the grid into which the data is to be imported. If you do not specify a file extension ABACUS will assume an extension of .exp.

Files> **T**ransfer - used to save and load ABACUS spreadsheets in a transferable format. Transfer files have two important properties:

- They do not depend on a particular machine.
- They consist of printable characters.

These properties allow you to send Transfer files as text files via 7-bit RS232 communications links to other computers. There are two types of Transfer files that you can use, Psion and DIF (Data Interchange Format) files.

Psion Transfer files save all features currently in use at the time that they are saved, so that the spreadsheet can be reproduced exactly.
DIF files only save the values and text in the spreadsheet grid. They can, however, be loaded into any of the variety of software packages that support DIF files.

To save or load a Transfer file press **T** for the Transfer files option followed by **S** , if you wish to save a Transfer file, or **L** , if you wish to load one. ABACUS will then ask you to specify the type of Transfer file - a **Psion** Transfer file or a **DIF** file.

ABACUS offers Psion Transfer files as the default, and you can accept this by pressing [Enter] , or you can select DIF files by pressing **D** . The PSION transfer format is described in Appendix B. Having made your choice, type in the file name for the Transfer file. If you do not specify a file name extension, ABACUS will assume an extension of .dif for DIF files and .abt for Psion Transfer files.

6.11.6 GRID This allows you to make changes which affect the entire spreadsheet. The options, selectable by their first letters, are:

Grid> **D**elete - allows you to delete one or more rows or columns from the grid. You are first asked if you want to delete rows (press [Enter]) or columns (press C). You are then asked to give the reference of the starting row (or column) of the region you want to delete, followed by [Enter] . You are then asked for the row (or column) reference of the end of the region. When you then press [Enter] the selected region is deleted and the following rows (or columns) close up to fill the gap. Empty rows (or columns) will be inserted at the bottom (or at the extreme right) of the grid. In both of these options all formulae in the rows or columns that are moved will be adjusted to correct them for their new positions.

Grid> **G**oalseek - recalculates the grid until a specified condition is met. ABACUS asks you to type in the reference to a cell containing, as a formula, the condition to be met. This formula must give a numeric result. When you press [Enter] , ABACUS will recalculate the grid repeatedly until the result of the formula is zero. For example, suppose we wish to continue recalculating the formulae in the grid until the difference between the numbers in cells J9 and K12 is equal to or less than 0.01. We can do this by typing the formula: abs(j9-k12)>0.01 into a cell;

suppose this is cell B3. This formula gives a true (non-zero) value as long as the difference is greater than 0.01. When you select the Goalseek option by:

[F3] G G B3 [Enter]

ABACUS will keep recalculating the grid until the difference is no longer greater than 0.01, when the formula in cell B3 gives a false (zero) result. You may stop the recalculation at any time by pressing [Esc] . In this case the result shown in the grid will be unreliable, since you may have interrupted the process part of the way through a recalculation.

Grid> **I**nsert - allows you to insert empty rows or columns into the grid. You are first asked if you want to insert rows (press [Enter]) or columns (press C). You are then asked to give a row (or column) reference for the insertion position and the number of rows or columns to insert. When you then press [Enter] empty rows (or columns) are inserted before the one specified. The last rows (or columns) will be lost from the grid. If, for example, you insert three rows, the last three rows of the old grid will be lost. This would not affect you unless you had data in those last three rows.

Grid> **P**rotect - asks you to specify a range of cells to be protected against the two most common ways of accidentally changing the contents of the wrong cell(s) - ie by accidentally typing a new value with the cursor at the wrong position, and using the amend command on a master formula shared by one or more of the protected cells. You are still allowed to make deliberate changes to the contents the cell by means of any command other than amend - eg copy and rubout - since they require an explicit reference to the range of cells to be affected. Protected cells are shown either in green, or in a different intensity from normal unprotected cells, depending on whether you are using a colour or a monochrome monitor.

Grid> **R**epeat - recalculates the grid a specified number of times. ABACUS asks you to type in the number of recalculations. For example: [F3] G R 5 [Enter] causes ABACUS to recalculate the grid five times before displaying the result. The number of recalculations must be between 0 and 255 inclusive. You may stop the recalculation at any time by pressing [Esc] . In this case the result shown in the grid will be unreliable, since you may have interrupted the process part of the way through a recalculation.

Grid> **S**ecurity - allows you to supply, change or delete a password for an ABACUS grid. You will not be able to load a file protected by a password unless you give the correct password. The command first asks you to type in the old password. If the file does not yet have a password you should respond by pressing [Enter] . You must then type the new password, terminated by pressing [Enter] . You are then asked to confirm the password by typing it again before it is accepted. The password may be up to eight characters long and may contain any character, except [Esc] (which aborts the command) and those characters (eg <-)

recognised by the line editor. Upper and lower case letters are regarded as being different. You can remove a password by pressing [Enter] when asked for the new password and for the confirmation. In all cases you should then save the ABACUS file. When you load a protected file you will be asked to type the password. If you type the wrong password then the file will fail to load.

Grid> **U**nprotect - allows you to remove any protection from the range of cells that you specify.

Grid> **W**idth - allows you to change the width (number of characters) of one or more columns. You are first asked to specify the number of characters in a column, and then to specify the starting and ending columns over which you wish the change to take effect.

6.11.7 JUSTIFY Modifies the positioning of text and numbers in a range of cells. It has two main options: to modify existing cells, or to set the default justification that ABACUS will use when you put data in a cell which is currently empty. You should press [Enter] to select the Cells option, or the D key to select the Defaults option. You are then asked to specify whether you want to modify the justification of text (by pressing [Enter]) or of numbers (by pressing the N key). In either case you can then select left ([Enter]), right (R) or centre (C) justification. In the case of the Cells option you are finally asked to give the range over which the change is to act. You do not have to give a range in the Defaults option. The new default will apply to all newly-created cells, at any point in the grid, until you make a further change in the default justification.

6.11.8 LOAD Loads a file from the disk. You are first asked to specify the file name; pressing the ? key at this point gives you a list the files on the default drive. If you do not include an extension in the file name you type in, ABACUS will assume an extension of .aba.

6.11.9 MERGE Combines or consolidates data from a previously saved file with the data in the current grid. You are first asked for the name of the file to be merged from the disk and will then have to indicate whether the data in this file is to be added to (press [Enter]) or subtracted from (press S) the data in the current grid. Whenever a cell (in the file) containing a number or a formula matches a corresponding data cell in the grid, the value from the file will be added to or subtracted from the grid data. The contents of any other cells are not affected. The command will not have any effect on grid cells containing text, which are therefore protected against alteration. Nor will it change any cells that you have protected against alteration with the Protect option of the grid command. The resulting grid contains purely numeric values in each cell that have been affected by the merge. The formulae that produced these values in the original grid cells will be destroyed. These formulae would not have any meaning in the consolidated grid. This command offers a fast and easy method of combining the data in two similar models. It is, of course, essential that you have laid out the two

grids in exactly the same way, using the same cell locations, for the results of the command to make sense.

6.11.10 ORDER Sorts the rows of the grid into ascending order, based on the contents of one particular column. You are first asked to specify the column on which the sorting is to be based. You are then asked for the first and last rows to be sorted. The rows are rearranged so that the contents of the cells of the chosen column are in ascending order with increasing row number. The exact ordering sequence that is used depends on the collation order. The default collation sequence is:

sequence is:	Empty cells
	Numeric cells, in ascending numeric order
	Text cells in alphabetic order

Only use the order command on areas of the grid which contain numeric data. It will invalidate any formulae present in the affected portion of the grid, as they are not adjusted for their new locations.

6.11.11 PRINT Sends a selected portion of the grid to a printer or to a disk file. You are first asked whether you want the printed grid to show the values or the formulae in each cell. Press [Enter] to show the values, or press the F key to show the formulae. ABACUS next asks you to specify the range of cells which you want printed. Then you are asked whether you want the grid border to be included (press [Enter]) or not (press the N key). Following this you should specify whether the output should be sent to the printer (press [Enter]) or to a disk file (press the F key). If you choose to send the output to a file, you are also asked to type in a file name (ending with [Enter]). The selected portion of the grid will be sent to the chosen destination. If you have chosen to direct the output to the printer you can stop the printing at any time by pressing [Esc]. If you have asked for a display of the formulae, ABACUS will first print a numbered list of all the formulae used in the grid. It then prints the grid itself. The formula number is shown in any cell that contains a formula. In the case of the option to print to a file, if you do not specify an extension when you type in the file name, ABACUS will assume an extension of .lis.

6.11.12 QUIT Leaves ABACUS and returns. When you leave ABACUS the current grid contents are lost. You are asked to confirm your request so that you have the chance to change your mind. You can cancel the command and return to your spreadsheet by pressing [Esc] . If you press [Enter] you will confirm your wish to cancel this ABACUS task.

6.11.13 RUBOUT Rubs out (deletes) the contents of one or more cells in the grid. When you use this command you will be asked to specify a range of cells. All the cells in that range will be cleared.

6.11.14 SAVE Saves a grid to a file on a disk. You are first asked to specify the file name; pressing the ? key at this point gives you a list of the files on the default drive. If you do not include an extension when you type in the file name, ABACUS will assume an extension of .aba.

6.11.15 TITLES Makes a number of rows and columns become an extension of the border of the grid. Such rows and columns scroll in the same way as the border as you move the window around the grid. The size of the window is reduced by a corresponding amount and you are not allowed to move the cursor into the title area by means of the cursor keys. Press [Enter] to set the titles - ie attach all rows and columns above and to the left of the position of the cursor to the border. If you select this command when titles are set, ABACUS offers you the option to unset the titles - ie restore the title rows and columns to their normal behaviour. You may also unset the titles by moving the cursor - with the Goto cell key (F5) - above and to the left of the title rows or columns.

5.11.16 UNITS Changes the way that numbers are displayed within a cell, or group of cells. It does not affect the values of the numbers in any way. You are first asked to select whether you want the command to affect existing cells (just press [Enter]) or to set the default format that ABACUS will use for all subsequently created cells (press the D key). In either case you are then asked to choose the display format from the following list (the list assumes the Cells option):

[F3] Units,Cells, Decimal - numbers are displayed in fixed-point decimal notation, ie all numbers are shown in the same way, with a fixed number of decimal places. Numbers which actually contain more decimal places than are displayed will be rounded up or down as necessary. The option asks you to type in the number of decimal places you want to be shown. It will not accept a value greater than 14.

[F3] Units,Cells, Exponent - numbers are displayed in exponential, or scientific notation. The option asks you to type in the number of decimal places you want to be shown. It will not accept a value greater than 14. Again the displayed number is rounded as necessary, to the number of decimal places that you select.

[F3] Units,Cells, Percent - this displays numbers as percentages so that, for example, the value 0.55 is displayed as 55%. The option asks you to type in the number of decimal places you want to be shown. It will not accept a value greater than 14.

[F3] Units,Cells, Integer - numbers are shown as integers, or whole numbers, as for the int() function. You are given the option for negative values to be enclosed in brackets, rather than with a leading minus sign. Press the B key for bracketed negative values, or [Enter] for a leading minus sign.

[F3] Units,Cells, General - this is a general numeric format in which any of the previous formats (except percent) is chosen, depending on the value of the number, to make best use of the space available in the cell.

[F3] Units,Cells, Monetary - numbers are displayed in fixed-point decimal format, with two decimal places and a leading currency symbol (which can be set up using the design command). You are given the option for negative values to be enclosed in brackets, rather than with a leading minus sign. Press the B key for bracketed negative values, or [Enter] for a leading minus sign. In the case of the Cells option ABACUS asks you, at the end of any of the above choices, to specify the range over which the change is to act. You can type in any form of cell or range reference (including labels or range identifiers). Press [Enter] to mark the end of the reference. ABACUS does not ask you to specify a range if you selected the Defaults option. In this case the selected format will be used for all new cells, as they are created.

6.11.17 WINDOW Controls whether the display is a single window or is split into two windows which can be used to show two separate portions of the grid. You are first asked to choose between a vertical (V) split, a horizontal (H) split, or to join (J) a split display back into a single window. If you choose to split the window then the split will occur at the column or row containing the cursor. You should therefore position the cursor at the point where you want the split to occur before making the split. Whole columns will always be displayed. Each window in a vertical split will never be less than ten characters wide. You then are given a further choice as to whether the two windows should move together (T) or separately (S). If you specify the T option, this means that any change in the position of one window - in the direction parallel to the split - will cause a corresponding change in the position of the other. Movements at right angles to the split are not related in this way. The S option allows the two windows to move around the surface of the grid independently.

6.11.18 XECUTE Forces a recalculation of all formulae appearing in the grid. A recalculation is normally performed automatically when you make any new entry in the grid. You will need to use this command if you have switched off the automatic recalculation option by using the design command or if you want to activate any askn() or askt() functions stored in the cells of the grid.

6.11.19 ZAP Clears the entire contents of the grid and returns you to the beginning of ABACUS for a fresh start. Since this command is irreversible, you will be asked to confirm your request. If you press [Esc] you will return to the command menu without any deletion taking place. You should press [Enter] to confirm your wish to clear the grid.

6.12 FUNCTIONS

6.12 FUNCTIONS Think of a function as a kind of recipe which converts a number of values, known as the function's arguments into a different value, which is said to be the value that is returned by the function. In ABACUS this is the value which would be shown in a cell containing the function. The functions provided by ABACUS may take three, two, one or no arguments. The arguments for a function are placed in brackets after its name. You must not leave a space between the name and the opening bracket, but spaces are allowed between items within the brackets. If a function takes more than one argument, the arguments are separated by commas. All functions must be followed by the brackets, even if they take no arguments. The presence of the brackets is a useful reminder that you are referring to a function. They allow you to distinguish between a label and a function, even if they have the same name. In the descriptions of the functions: **n** - is either a numeric expression or a reference to a cell displaying a numeric value **text** - is either a text expression or a reference to a cell displaying a text value **range** - is a grid range reference.

A numeric expression is either a number or an expression which gives a numeric result. A text expression is either a text string (enclosed in quotes) or an expression which gives a text result. The following functions are provided:

ABS(n) - Returns the absolute value (that is, the value ignoring any minus sign) of the argument. For example, abs(3) returns 3 and abs(-7) returns 7.

ASKN(text) - Allows the input of numeric data. It displays the given text (which may be up to 40 characters in length) as a prompt in the input line, followed by a '?', and waits for a reply to be typed in. The reply is shown in the cell containing the function. Input will only be requested when you first put the function into a cell, and when you recalculate the grid by use of the xecute command. It is not asked for during an auto-calculate after each grid entry.

ASKT(text) - Allows the input of text strings. It displays the given text (which may be up to 40 characters in length) as a prompt in the input line, followed by a '?', and waits for a reply to be typed in. The reply is shown in the cell containing the function. Input will only be requested when you first put the function into a cell, and when you recalculate the grid by use of the xecute command. It is not asked for during an auto-calculate after each grid entry.

ATN(n) - Returns the angle, in radians, whose tangent is n.

AVE(range) - Returns the average of the numeric values contained in all the cells in the specified range. Empty cells and cells containing text are ignored in the calculation of the average. If there are no numeric cells in the range it will return a value of zero.

CHR(n) - Returns the ASCII character whose code is n. A character with an ASCII code less than 32 has no effect on the screen, but is sent to the printer (when you

print the portion of the grid containing it) if preceded by an ASCII null. For example, chr(0)+chr(13) passes the ASCII character for a carriage return to a printer, when the cell containing it is printed. You can show an 'A' on the screen by putting chr(65) in a grid cell.

CODE(text) - Returns the ASCII value of the first character found in the specified text.

COL() - Returns the number of the current column.

COS(n) - Returns the cosine of the given (radian) angle.

COUNT(range) - Returns the number of non-empty cells in the specified range. Only numeric cells are included in the count.

DATE(n) - Returns today's date as a text string in one of three forms:

n	date string	
0	"YYYY/MM/DD"	
1	"DD/MM/YYYY"	
2	"MM/DD/YYYY"	You must first have set the system clock.

DAYS(text) - Returns the number of days, from the first of January 1583, to a date given by a text expression of the form "YYYY/MM/DD". The conversion assumes that the Gregorian (modern) calendar - which was first introduced in 1582 - is being used and is therefore only valid for dates after 1582.

DEG(n) - Takes an angle, measured in radians, and converts it to the same angle in degrees.

EXP(n) - Returns the value of the constant e (approximately 2.718) raised to the power n. The returned value will be in error if n is greater than +88, since the result will then exceed the numeric range of ABACUS.

IF(expression,true,false) - expression:= n, true:= n or text, false:= n or text Calculates the value of the expression and uses it to determine which of the following two arguments should be returned. If the expression evaluates to 0 it is considered to be false and the "false" argument is returned. Any non-zero value for the expression is interpreted as being true and causes the "true" argument to be returned. The "true" and "false" arguments may be either text or numeric in nature. Thus both the following examples are valid uses of the function.

if(A1=B1,"equal","not equal")
if(A1,1,0) You can also mix a text and a numeric argument as in the following example. Try this one out if you are not sure how if() works.

[A1] 1 [B1] 0 [C1] if(A1 or B1,"either",0)

You should see the word 'either' appearing in cell C1 since the first parameter of if() returns a non-zero (true) value if either cell A1 or cell B1 contains a non-zero value. If you change the contents of cell A1 to be zero then you will see a zero displayed in cell C1. Note that both parts of the expression are evaluated and if an error occurs in either part you will be told.

INDEX(column,row) column:= n, row:= n Returns the contents of the cell at the intersection of the specified column and row.

INSTR(main,sub) main:= text, sub:= text Finds the first occurrence of 'sub' within 'main' and returns the position of the first character of 'sub' in 'main'. It will return a value of zero if no match is found. The match is case-dependent.
instr("January","Jan") returns 1 instr("January","an") returns 2
instr("January","AN") returns 0

INT(n) Returns the integer value of the number, by truncating at the decimal point. The truncation is always towards zero. For example: int(3.7) returns 3
int(-4.8) returns -4

IRR(range,period) period:= n Calculates the Internal Rate of Return for the numeric data in the specified range, which may be either a row or a column. The data in the range represents a cash flow for each of a series of periods, separated by n months. Negative values represent cash outlays and positive values represent cash returns. The function returns the rate of interest necessary so that investment of your outlay would match the proposed returns. The interest is calculated on an annual flat rate basis. Because of the nature of this function, the calculated rate of interest is only accurate to one decimal place. Note that the first item in the range is counted as period zero, the next is period one, and so on. The function assumes that each amount is payable in full at the end of the relevant period.

LEN(text) Returns the number of characters in the specified text.

LN(n) Returns the natural, or base e, logarithm of n. An error results if n is negative or zero, since logarithms are not defined in this range.

LOOKUP(range,offset,value) offset:= n, value:= n Implements a look-up table in the grid. Two tables of values are assumed to be present. The first table occupies the specified range (which can be in a row or a column). The second table runs parallel to the first, in a following row or column, separated from the first by "offset" cells. For example, if the first table is in column G, from G10 to G25, and "offset" is 3, the second table will be assumed to be from J10 to J25. Every entry in the first table should have a corresponding entry in the second. The first table is searched for the largest value that is less than or equal to the specified "value". The function returns the corresponding entry from the second table. Note that it is assumed, for the correct operation of this function, that both tables

contain numeric values, and that those in the first table are arranged in ascending order. The first value in the first table is a dummy. It must be less than the second value, which is the lower limit for the table lookup process. It is otherwise ignored. The first value in the second table is the value that is returned if lookup() is called with any number less than the lower limit.

MAX(range) Returns the largest numeric value found in the cells in the specified range. If there are no numeric cells in the range the function will return zero.
MIN(range) Returns the smallest numeric value found in the cells within the specified range. If there are no numeric cells in the range the function will return zero.

MONTH(n) Returns, as text, the name of month n. For example, month(3) returns the text "March". If an argument larger than 12 is used, it is replaced by the remainder after division by 12 so that, for example, month(13) and month(1) will both give the result "January".

NPV(range,period,percent) percent:= n, period:= n Calculates the Net Present Value for the cash flow data in the specified range. "Percent" is the annual interest rate (14 represents a 14% rate). The data is assumed to refer to a series of periods, separated by equal intervals of "period" months. The net present value is the amount of money required now to produce a given future cash flow, assuming a percentage interest rate. The interest is quoted as an annual flat rate (ie a 12% interest rate with periods between cash flows of 3 months is interpreted as a rate of 3% per quarter).

PI() Returns the value of the mathematical constant pi.

RAD(n) Takes an angle, measured in degrees, and converts it to the same angle in radians.

REPT(text,n) Returns a string consisting of n copies of the first character of the given text. For example, rept("_*",5) returns "*****" , rept("abc",3) returns "aaa"

ROW() Returns the number of the current row.

SGN(n) Returns +1, -1, or 0, depending on whether the argument is positive, negative or zero.

SIN(n) Returns the value of the sine of the specified (radian) angle.

SQR(n) Returns the square root of the number n, which must not be negative.

STR(n,type,dp) num:= n, type:= n, dp:= n Converts a number, num, to the equivalent text string. Type indicates the form of the converted string as follows; 0 decimal (floating point), 1 exponential, or scientific, notation, 2 integer, 3 general format The third parameter, dp, specifies the number of figures after the decimal point in the converted string. It should always be included, although its value is ignored for integer, general and monetary formats.

SUM(range) Returns the sum of all the numeric values within the specified range. Empty cells and cells containing text are ignored.

TAN(n) Returns the tangent of the specified (radian) angle.

TIME() Returns, as text, the time of day in the format "HH:MM:SS". You must first have set the system clock.

VAL(text) Converts the text to its equivalent numeric value. It will only convert text composed of valid numeric characters and the conversion will stop at the first character that can not be interpreted as a digit. For example, val("1.1ABC") will return the value l.l, and val("ABC") will return 0.0.

WIDTH() Returns the width, in character spaces, of the current column. Note that there is one space separating adjacent columns.

7.0 ERRORS

7.1 Grid Errors Any syntax error in a formula - such as supplying the wrong number of arguments for a function, or mis-matched brackets - will be reported at the time you type in the formula. You are told the nature of the error and the formula is left in the input line. You can then examine it, and then correct it with the line editor. The possible error messages are listed below.

- Missing closing quotes in a string eg 'abc' + 'def
- Badly formed numeric constant eg 1.5e (missing number after 'e')
- Number out of range eg 1.5e99 Illegal character eg 12 __ 5 (underscore instead of minus)
- All names must refer to columns
- All names must refer to rows
- Name references may only be relative (see section 6.8 on Cell References)
- Badly formed range reference eg a1:
- Badly formed name reference eg c3.
- Name is not a row or column
- First name reference undefined
- Second name reference undefined (the text does not appear in the grid, above and to the left of this cell)
- Function requires a range reference eg irr(1,2,3) - see description of irr()
- Illegal range
- Syntax error
- Mismatched brackets
- Type mismatch eg 1 + 'abc'

- Wrong number of function arguments eg sqr(1,2)
- String bigger than 255 characters eg rept("*",256)
- Division by zero
- Illegal function arguments eg sqr(-1)
- Function requires a range reference eg max()
- String subscript out of range (either subscript less than zero or greater than 255, or first subscript greater than length of text)
- Reference out of range (to a cell outside the grid)
- Reference to an error cell (the formula refers to a cell containing a formula which produces one of the errors described below)
- Out of memory, use RUBOUT to make more room (this will only occur on very large grids where you have a very large number of separate formulae - about 8000. The rubout command should be used to reduce the size of the grid.)

Other errors, such as attempting to add a text value to a numeric value, will not be detected until the result of the formula is calculated - after the formula has been placed in the grid. If a formula contains a reference to an empty cell, ABACUS will assume that the cell has a numeric value of zero. This may well cause an error - eg division by zero - when the formula is calculated. If ABACUS detects an error when a formula is calculated it gives a brief error message in the relevant cell. You can then move the cursor to the cell to examine the formula and find out what is wrong. The possible errors are:

##TYPE	the formula contains a reference to a cell containing information of the wrong type, ie numeric instead of text, or vice versa.
##LONG	the formula contains a reference to a text string that is more than 255 characters long.
##ZERO	The formula is attempting to divide something by zero.
##NUM	The formula contains a function called with the wrong number of arguments, eg sin(1,2).
##ARG	The formula contains a function called with a non-valid value for one or more of its arguments, eg ln(-5).
##SUB	The formula uses a string slicing operator with an error in one or more of its subscripts.
##REF	The formula contains a reference to a cell which is outside the grid. The formula in such a cell will show the word "ERR" for each cell reference that is not valid.
##ERR	The formula contains a reference to to a cell which contains an error. You can ignore these messages since they will disappear when the original error, in the cell to which the formula refers, is corrected.

7.2 File errors

The following error messages will only appear if an error occurs while you are using a file-related command eg load or files.

- does not exist - The file name you gave was not found in the specified directory.
- Cannot open file - xxxxxxxx.xxx - The file xxxxxxxx.xxx was not found on the disk in the specified drive.
- File I/O incomplete - The loading or saving of a file has started successfully but has failed at a later stage - this may mean that the data in the file has been corrupted, or that the disk or the disk drive mechanism has been damaged.
- Unable to open file - The file can not be opened - for one of the reasons given for the previous error.
- Wrong file type - The file name extension is not the one that ABACUS was expecting - eg attempting to use the load command on an export file, instead of the Import option of the files command.
- Illegal file name - eg "3test" File names must start with an alphabetic character and may not be more than 8 characters long.
- Illegal import file format - This is only likely to occur if you attempt to import a file not created with an export command.

10.7 THE START-UP PARAMETERS

When you first load ABACUS or after a Zap command, it is in the state described by the following list. You can change each of the properties by the method indicated in the right hand column.

Feature	***Initially***	***Change By***
Auto-calculate	YES	[F3] Design A
Blank if Zero	NO	[F3] Design B
Calculation Order	Row	[F3] Design C
Form Feed	Yes	[F3] Design F
Gaps between lines	0	[F3] Design G (number)
Lines per page	66	[F3] Design L (number)
Monetary symbol	£	[F3] Design M (symbol)
Printer paper width	80	[F3] Design P (number)
Grid Column Width	10	[F3] Grid Width (no.) from to
Password	off	[F3] Grid Security (password)
Protection	off	[F3] Grid Protect (range)
Titles	not set	[F3] Grid Titles
Justify	Text Left Numbers Right	[F3] Justify
Units	General	[F3] Units

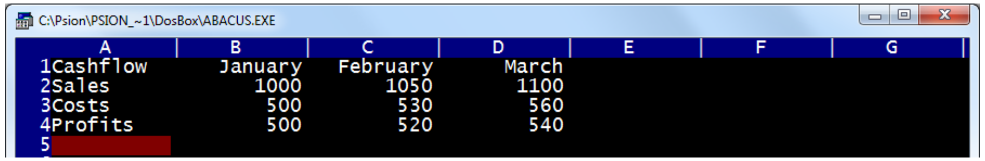
Appendix A

Import, Export and Transfer

You can transfer information between the programs of the PC-FOUR family with a comprehensive set of import and export commands. The information used by, for example, ABACUS, ARCHIVE and EASEL is very similar in nature, in that it can always be represented in the form of a table. Transferring information between them is therefore straightforward. QUILL handles formatted text which cannot, in general, be represented in tabular form. It must be treated in a different way from the other members of PC-FOUR. Each member PC-FOUR has its own Export and Import commands, usually as an option within the files command. ARCHIVE, however has separate Export and Import commands within its programming language.

First consider the direct use of the Export command of one task and the Import command of another. These commands are described in the commands section. Export creates a named file which is automatically saved on disk. This file can then be imported to another task, or to several tasks. The export file remains available until you decide to delete it.

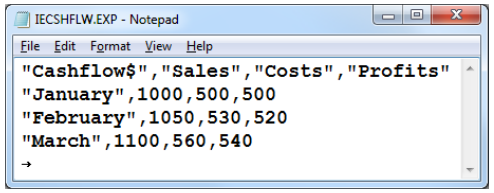
Let us first consider export and import between ABACUS, ARCHIVE and EASEL. The export files produced by all three programs are identical in structure and can be imported to any of them, regardless of the program of origin.   Suppose we have an ABACUS grid containing the following information ready for Exporting:



	A	B	C	D	E	F	G
1	Cashflow	January	February	March			
2	Sales	1000	1050	1100			
3	Costs	500	530	560			
4	Profits	500	520	540			
5							

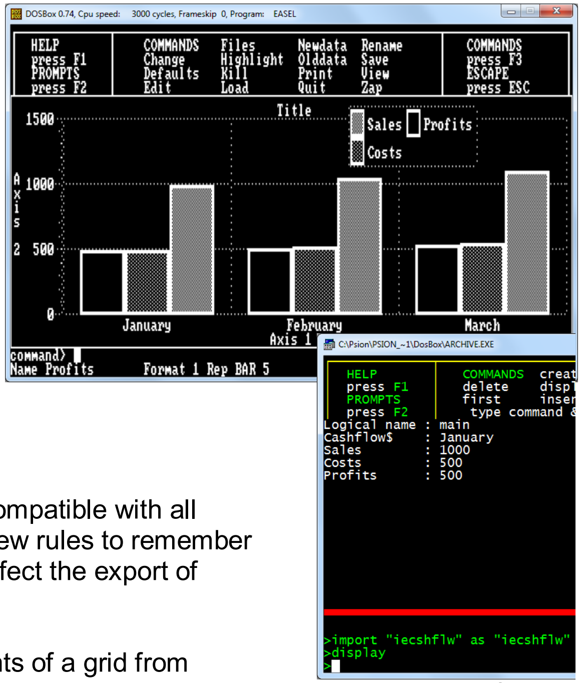
If we exported this data and then imported it to EASEL, it would be interpreted as three sets of figures, named 'sales', 'costs' and 'profits'. EASEL interprets the first set of text items that it finds - in this case the month names - as the cell labels for the graph.

The scheme is:
cell labels>January,February,March
sales graph>1000,1050,1100
costs graph>500,530,560
profits graph>500,520,540



```
"Cashflow$","Sales","Costs","Profits"
"January",1000,500,500
"February",1050,530,520
"March",1100,560,540
→
```

EASEL does not use the first piece of text 'cashflow'. When you export a set of figures from EASEL, it automatically inserts the text 'label' in this position to maintain compatibility with ABACUS and ARCHIVE. If we were to import this same export file to ARCHIVE, the result would be a data file containing three records, each of which would have four field names cashflow$ (a text field) costs sales and profits (numeric fields).


DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: EASEL
HELP press F1 PROMPTS press F2
COMMANDS Change Defaults Edit
Files Highlight Kill Load
Newdata Olddata Print Quit
Rename Save View Zap
COMMANDS press F3 ESCAPE press ESC
Title
Sales
Profits
Costs
1500
1000
500
0
Axis 2
January
February
March
Axis 1
C:\Psion\PSION_~1\DosBox\ARCHIVE.EXE
HELP press F1 PROMPTS press F2
COMMANDS delete first
creat displ inser
type command &

```
command>
Name Profits          Format 1 Rep BAR 5
```

```
Logical name : main
Cashflow$    : January
Sales        : 1000
Costs        : 500
Profits      : 500

>import "iecshflw" as "iecshflw"
>display
>
```

Rules

To make sure that export files are compatible with all members of PC-FOUR there are a few rules to remember about exported data. They mainly affect the export of information from ABACUS.

(1) When you export the contents of a grid from ABACUS, the section of the grid being exported must have text in the first cell of each row (or of each column if you export it in column order). The text must not include spaces but an underscore, for example net_profit is acceptable.

(2) If the first cell of any row (or column) of an ABACUS grid is empty, that row (or column) is not exported. There must be data in the cell immediately following the text cell in each exported row (or column). The type of this data (numeric or text) determines the data type used for all the data in the rest of that row (or column). Each row (or column) must therefore contain all numeric or all text data.

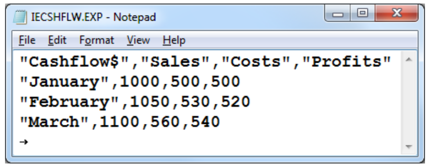
(3) You can export files from ABACUS or ARCHIVE which contain several sets of textual data. EASEL can only export a file containing one set of textual data - the cell labels.

(4) If you import a file containing several sets of data to EASEL, it uses the first set for its cell labels and ignores all following sets of text.

The Export File Structure

The export file consists of a series of records. Each record ends with the two characters <CR> (ASCII code 13) and <LF> (ASCII code 10). The import commands will, however, accept either of these characters - or the two together, in either order - as an end of record marker. The end of the file is marked by a control-Z (^Z) character (ASCII code 26).Each record consists of a series of values, separated by commas. The values are either text (which is enclosed in quotes) or numbers. The first item in each record must be text. If its name ends with a dollar sign, the following values must be all text. Otherwise the following values must be all numeric. The export file produced by exporting the data of from ABACUS.

```
"cashflow$","sales","costs","profits"<CR><LF>
"January",1000,500,500<CR><LF>
"February",1050,530,520<CR><LF>
"March",1100,560,540<CR><LF>
^Z
```



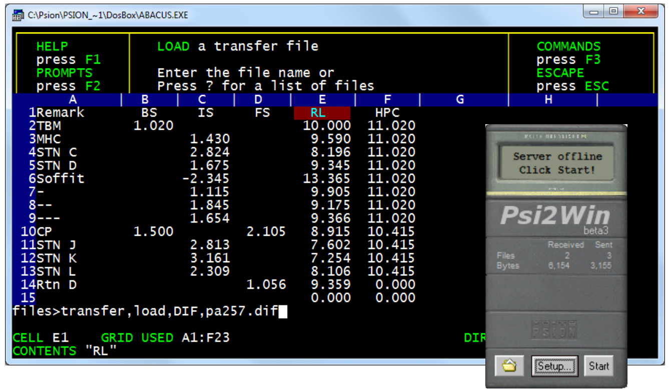
```
"Cashflow$","Sales","Costs","Profits"
"January",1000,500,500
"February",1050,530,520
"March",1100,560,540
→
```

Export and Import for QUILL

Since QUILL works with formatted text ,a file imported to QUILL must be a plain text file, rather than the normal tabular export file structure. QUILL will accept any ASCII text containing form feeds and line feeds (ASCII codes 12 and 10 respectively) in addition to the printable ASCII characters. Any other characters in the file are simply ignored. QUILL interprets a line feed as the end of a paragraph and a form feed as a page break. The Export commands of ABACUS and ARCHIVE can produce text files suitable for import by QUILL. ARCHIVE can also export a formatted report to QUILL. You do this by printing (with **lprint**) the report to a file, using the export option of the spoolon command. A file exported from QUILL contains only plain text and line feed characters, marking the end of each paragraph. In general, a file exported from QUILL is not suitable for import to the other members of PC-FOUR. The main purpose is to be able to produce text files that are readable by a wide range of other programs. (You can, however, write text - containing the necessary quotation marks, dollar signs and commas - that will result in an export file that can be imported to ABACUS, ARCHIVE or EASEL.) One obvious use of export from QUILL is to allow you to write or edit programs. You can, for example, write ARCHIVE programs in QUILL. Once you have exported them, they are in a suitable form for immediate loading and running.

Import from PSION Organiser II Pocket Spreadsheet

Connect the Organiser to a PC with *Psi2Win* select [Plan] and [Load] the pocket spreadsheet file. Use [Mode][File][export][Dif] filename. In abacus use [F3] File Transfer Load Diff filename. This will transfer the 'data' but all the formulae and formatting will be lost.



```
 HELP           LOAD a transfer file                 COMMANDS
 press F1                                            press F3
 PROMPTS        Enter the file name or               ESCAPE
 press F2       Press ? for a list of files          press ESC
        A       |  B  |  C  |  D  |  E  |  F  |   G    |   H   |
  1Remark         BS     IS     FS     RL     HPC
  2TBM          1.020               10.000  11.020
  3MHC                 1.430         9.590  11.020
  4STN C               2.824         8.196  11.020
  5STN D               1.675         9.345  11.020
  6Soffit             -2.345        13.365  11.020
  7-                   1.115         9.905  11.020
  8--                  1.845         9.175  11.020
  9---                 1.654         9.366  11.020
 10CP           1.500         2.105  8.915  10.415
 11STN J               2.813         7.602  10.415
 12STN K               3.161         7.254  10.415
 13STN L               2.309         8.106  10.415
 14Rtn D                      1.056  9.359   0.000
 15                                  0.000   0.000
files>transfer,load,DIF,pa257.dif

CELL E1    GRID USED A1:F23                          DIR
CONTENTS "RL"
```

Appendix B

ABACUS transfer document format

The format of transferable ABACUS spreadsheet files Written by Peter - 10/09/85 Purpose of transfer files To allow a spreadsheet to be passed between machines with different implementations of ABACUS, without information loss. This is different from export files where the formulae and attributes of a spreadsheet is lost. To allow a spreadsheet file to be sent down a seven bit link without information loss. A virtual (IBM PC in reality) character set is used to allow correct handling of foreign characters. Characters with bit 7 set are represented by a hex code preceded by a ^, e.g. a acute accented 'e' is represented by ^82. This scheme is used to represent the monitory value symbol and strings.

Restrictions and information loss using transfer files. Any password the spreadsheet may have is lost in a transfer file. When moving a large model spreadsheet with protected cells into ABACUS that does not support these, this information about the cells will be lost. The characters used for monitory values is not mapped so there may be hash/pound type problems. A large spreadsheet may fail when moved into a small grid version of ABACUS or a system with little memory available. In this case ABACUS will reinitialise. If a formulae contains a function that is not known by the receiving ABACUS, the spreadsheet being loaded will be rejected and ABACUS will reinitialise. This will also be the case when formulas containing external file references are loaded into a small model ABACUS. The values of formulated numeric and text cells are stored even though they could be recalculated because: - this is the way the regular load operates - a spreadsheet full of askns etc can be loaded with original values - its easier to program, less extensive renumbering of formulae required.

Spreadsheet transfer filenames and user interface The default extension of an ABACUS transfer file is ".abt". Transfer files are saves / loaded under the transfer option of the files command. The overall structure of an ABACUS transfer file A transfer file is a line structured ASCII file, each line being terminated by a CR (ASCII 13) character. The first line contains four characters that denotes the file type. These are "TAB0" (for transfer ABACUS file version 0). Then follow sections for:
- Global attributes of spreadsheet.
- Column width information.
- Public grid section declarations.
- Spreadsheet grid structure and content.
- Master formulae and text table.

Cell references, numeric values and string in a transfer file Integers in the transfer file represents 16 bit values and are stored as signed integers, e.g. "-1" is -1 or 65535. Column references is a transfer file consisting of two alphabetic characters, the first of which is a space if the column denoted is less than 27, otherwise these references are exactly as seen on the grid. eg. " A" is column A "AA" is column AA Row references consist of an integer that gives the row no.

The first row number is always 1. Cell references consists of a column reference followed by a row reference. eg. " A1" is cell A1 "BL23" is cell BL23 Floating point no. are stored in the same form that ABACUS uses for exporting them, i.e. dtob(double,3,0,buffer,20). They are always delimited by a end of line. String text is always stored on a line by itself and is delimited by the end of line. e.g. "hello" is stored as hello<CR>

Global attributes of spreadsheet These attributes are stored on two lines of the transfer file. The first line contains 11 characters which are as follows:

<L|R|C>	text justification type (left/right/centre)
<L|R|C>	numeric justification type (as above)
<(|_>	brackets for negative values or not
<I|D|E|M|%|G>	integer, decimal, exponent, monatory, percent, general
<Y|N>	auto calculate On/Off
<Y|N>	blank if zero On/Off
<R|C>	calculation order, row or column
<4|6|8>	display setting: 40, 64 or 80 columns
<Y|N>	form feeds On/Off
<n>	gap size, single ASCII digit
<c>	monetary symbol char

e.g. RL_IYNR8Y2$

The second line consists of three integers separated by commas, which denote the default no. of decimal places, the no. of lines per page, and the printer line width, e.g. 2,66,80

Column width information The first line in this section contains an integer giving the no. of following column widths. The column widths then follow as integers one per line, for the consecutive columns. e.g. 255 11 this is the width of column A ... another 254 lines of column widths

Public grid section declarations The first line contains an integer which gives the number of declarations given in this spreadsheet. If the spreadsheet saved does not contain publically accessible grid sections, this section consists of a line with an integer value of 0. If the ABACUS implementation loading this spreadsheet file cannot support public spreadsheet sections, it uses the integer in the first line as a count of the number of entries to skip before the spreadsheet grid occurs. It should skip 2 lines per entry. Each entry consists of two lines which are:
The name of the grid section
The cell ref. of the top left cell, row or col., no. of cells e.g. 1
1 public grid section bignose name of grid section BL230 C10 a column of 10 cells starting at BL230
Spreadsheet grid structure and content The spreadsheet grid is saved by rows with the value and attributes of each used cell in the row being recorded. The

section starts with two integers on a line separated by a comma which give the maximum column and the maximum row of the grid being used. From this the loading programs can determine whether the grid to be transferred will fit in the particular version of ABACUS being used. Next comes a single integer on a line which gives the no. of rows that contain non empty cells. Then for each row of the spreadsheet comes: two integers on a line giving the row number of this row and the no. of used cells in this row. now for each cell in the row comes: the column reference is a standard two character column id, e.g. " A"for column A, "BL" for column BL. the attribute are four characters followed by a decimal integer as follows:

<F|G|N|S>	cell type, formula/litt, num/str
	F = formulated num
	G = formulated str
	N = literal num
	S = literal str

<L|C|R>	numeric justification
<L|C|R>	text justification
<Y|N>	cell protected, Y = Yes
<(|_>	negative value in brackets flag
<I|D|E|M|%|G>	numeric format n integer, no. of decimal places e.g. BLFLRY(M10

a line containing the current value of the cell, if the above cell type was a literal or formulated string then this line contains a string reference no. otherwise it contains a floating point no. in ASCII form. if the cell type indicated a formulated value then the next line contains a formulae reference no., otherwise this line is absent.
e.g. 2 Thus an example spreadsheet grid in transfer file form might look as follows:

2,1	range of grid is A1:B2 2 - 2 rows coming up
1,3	here is row 1 with 3 cells in it AFR_G0 col A, f(n), right justify, general form, neg id '-', zero d.p.s
5	current value of cell is 5 20 uses formula no. 20 to calculate value of this cell BFR_G0 col B, f(n), e.t.c.
1	current value 1 22 uses formula no. 21 to calculate value of this cell 2,1 here is row 2 with 1 cell in it AFR_G0 col A, f(n), right justify, general form, neg id '-', zero d.p.s -0.95892427466314 current value of cell is ...
23	uses formula no. 23 to calculate value of this cell Master formulae and text table The master formulae and text are stored in this section. The first entry in this table is described as formulae/text no. 20 by reference within the grid, the second as no. 21, e.t.c.

The table starts with a single integer on a line giving the no. of formulae/text entries in the table. Then for each formulae/text there follows: a line of the form <F|T>n,n where <F|T> indicates F = formula, T = text n, the no. of references in the grid for formula or text n the no. of bytes required to store formula or text then if the entry was text ('T') the next line contains the characters of the text delimited by a <CR>, or if the entry was a formula ('F') the following lines describe the formula in a way described in the next section.

The representation of formulae in the transfer file A formula consists of a number of piece identifiers that indicate what the type of the following data is, or in the case of operators, what the operator itself is. A list of the possible piece identifiers are: OPERATORS: '<' '<=' '>' '>=' '<>' '=' '+' '-' '*' '/' '(' ')' '~' unary minus '&' logical AND '|' logical OR '!' logical NOT 'Ÿ' "to" for use in string slicing ',' comma LITERALS: 'N' floating point literal, to be followed by f.p.num. on next line 'S' string literal, to be followed by string text on next line

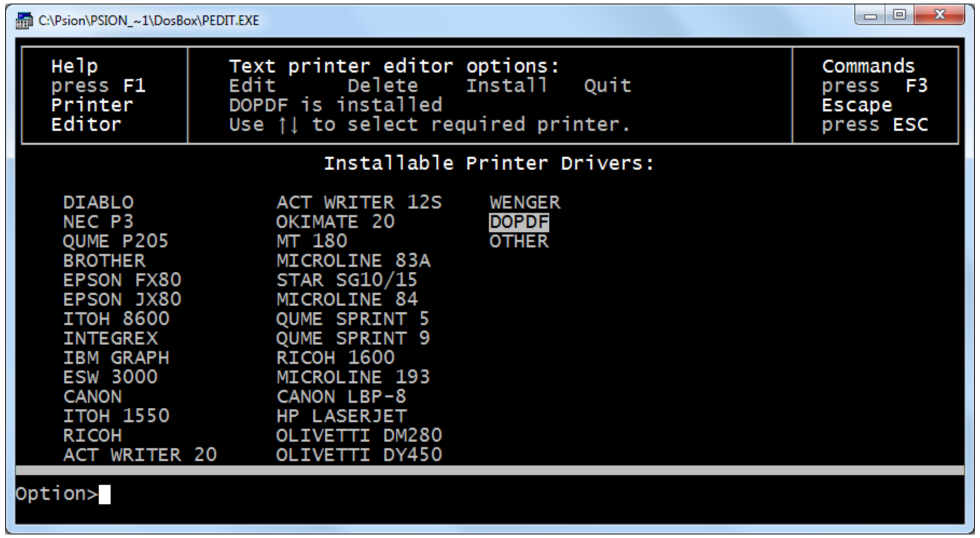
CELL REFERENCES: 'C' relative cell reference, followed by n,m where n is the signed column offset, and m is the signed row offset. Delimited by spaces. '$' absolute cell reference, followed by a cell reference. Delimited by space. ':' relative range reference, followed by n,m:p,o where n,m and p,o are relative cell references for the top left and bottom right of the range. Delimited by a space. '[' external range reference, followed by x,y giving the position where the external range is mapped on the current grid, and then on a separate line "file;name" delimited by the end of line giving the file the external range is to be found in and the name it is declared under. OTHERS: 'F' function application, text of function delimited by '(' follows. '.' end of formula. Here is an example master formula/text table:

4 4 formulas/text in table F1,17 formula 20, referenced onced, requires 17 bytes of storage N numeric value 1 floating point no. 1 + C0,0 plus, relative cell ref. C[+0]R[+0] . Formula is "1+C[+0]R[+0]" F1,17 formula 21, referenced onced, requires 17 bytes of storage + C-1,1 plus, relative cell ref. C[-1]R[+1] . Formula is "1+C[-1]R[+1]" F1,17 formula 22 N numeric value 1 floating point no. 1 + C-2,2 formula is "1+C[-2]R[+2]" F1,8 formula 23, referenced once, requires 8 bytes of storage. Fsin(C0,-1) . function "sin",C[+0]R[-1] . Formula is "sin(C[+0]R[-1])" .

Appendix C

Printer Drivers

Use PEDIT to create an appropriate printer driver. For any PC FOUR application to print it requires a printer configuration file: TPRINT.RES for text based applications and GPRINT.RES for the Easel Application.



```
C:\Psion\PSION_~1\DosBox\PEDIT.EXE

Help            Text printer editor options:              Commands
press F1        Edit      Delete    Install   Quit        press  F3
Printer         DOPDF is installed                        Escape
Editor          Use ↑↓ to select required printer.        press ESC

                     Installable Printer Drivers:

    DIABLO              ACT WRITER 12S    WENGER
    NEC P3              OKIMATE 20        DOPDF
    QUME P205           MT 180            OTHER
    BROTHER             MICROLINE 83A
    EPSON FX80          STAR SG10/15
    EPSON JX80          MICROLINE 84
    ITOH 8600           QUME SPRINT 5
    INTEGREX            QUME SPRINT 9
    IBM GRAPH           RICOH 1600
    ESW 3000            MICROLINE 193
    CANON               CANON LBP-8
    ITOH 1550           HP LASERJET
    RICOH               OLIVETTI DM280
    ACT WRITER 20       OLIVETTI DY450

Option>
```

To run PEDIT ensure PEDIT.EXE and PRINTER.RES are in the same directory. Use [Edit] to adapt or create a new printer driver. Using [Install] will place (or overwrite) the TPRINT.RES in the same directory.

Appendix D

Zip file contents. PCFOUR.ZIP and Examples.ZIP

PCFOUR.ZIP		**PCFOUR.ZIP**	
Group.aba	Example Spreadsheet	*Unnamed.TMP*	*Temporary file*
GemEasel.app	GEM OS Application	*ABACUS.TSL*	*xchange Task Sequencing Language tutorial files*
ABACUS.EXE	PC-FOUR Applications, including the printer driver editor (PEDIT)	*ARCHIVE.TSL*	
ARCHIVE.EXE		*EASEL.TSL*	
EASEL.EXE		*QUILL.TSL*	
PEDIT.EXE			
QUILL.EXE			
Company.dbf	PCFOUR.ZIP Supplied example database files	**EXAMPLES.ZIP**	
Expenses.dbf		Birds.doc	Birds list document
Payrol.dbf		QuillMan.doc	Quill Manual
Persons.dbf		Template.doc	Quill 9.1
Salaries.dbf		**Examples.aba**	Abacus 3.3
Blunders.exp	Export file for Quill	**CashFlow.aba**	Abacus 5.1
ABBA.HOB	Application Help Files	**BarChart.aba**	Abacus 5.2
ARCHV.HOB		**Cheques.aba**	Abacus 5.4
GRAF.HOB		**AutoBar.aba**	Abacus 5.6
QUILL.HOB		**Mortable.aba**	Abacus 5.7.1
Expenses.IX1	Example database index files	**MortCalc.aba**	Abacus 5.7.2
Payrol.IX1		**IRR.aba**	Abacus 5.8
Persons.IX1		**HPC02.aba**	Abacus 5.9
Salaries.IX1		**IECshFlw.aba**	Abacus Appendix A
Persons.IX2		MailList.prg	Archive 9.0
*.OVL	*xchange overlay files*	QPSION.csv	Archive 14.0
Demo.prg	Example database programme files	QPSION.exp	
Persons.prg		QuizPack.scn	
Printer.res	Printer driver files	QuizPack.prg	
TPrint.res		Qarchive.exp	
Gazet.scn	Screen layout files for Archive	Qquill.csv	
Helpp.scr		CashFlow.exp	Easel 3.4
Persons.scn		CashFlow.grf	

Manual display suggestions

Print the manual pages - 2 up on A4 landscape single sided. Centre cut to A5 sheets and place in A5 Poly Pocket Ring Binder or A5 Display book presentation folder.