

Software Surgery

THE COLUMN THAT TAKES A LOOK INSIDE THE LATEST RELEASES

Addcomm
Vine Micros

You get a better bit of Basic with Addcomm

AS the name indicates this is a utility program which adds commands to your Electron, giving you, in effect, an even more extended Basic.

It is stored on a ROM chip and to be able to use it you must have a ROM card or box such as the Slogger ROM Box.

The added commands fall into four categories:

- Graphics.
- Logo graphics.
- Toolkit commands.
- Miscellaneous.

Electron graphics are already superb, but the Addcomm commands make them even better. There are, for example, commands for creating circles and ellipses. They are really easy to use.

For instance:

```
10 MODE5
20 CIRCLE640,512,400
```

will draw a circle of radius 400 screen units with its centre at the centre of the screen. If you don't like your screen being 1280 units across and 1024 units up, you can scale it.

```
SCALE0,10,0,10
```

will make the screen 10 units across and 10 units down. To get a similar circle you need:

```
CIRCLES,5,4.
```

On the scaled screen you cannot use MOVE, DRAW or

PLOT but you can use SMOVE, SDRAW and SPLOT.

Any line in a program can mix Basic and Addcomm statements or variables. So:

```
CIRCLEX,Y,RND(500)
```

is quite acceptable. Extra parameters give arcs.

If you've ever envied the colour fill routines you see in commercial programs it's easy with Addcomm. The statement FILL with X and Y coordinates will fill from the point specified to the edge of the screen or to a non-background colour.

This will fill any shape, however complex, even through text. If you fancy patterned filling, CFILL allows you to specify a mix of colours for individual pixels.

It's even possible to have a non-upright screen by using ROTATE, or to shift the whole thing up or to the side with the TRANS command.

Logo graphics give you the

chance to create displays with minimal use of coordinates. It's rather more like using a pen.

First you decide where to start on the screen (LMOVE). This sets the position of the Logo cursor.

You choose your PEN (dots, lines or fills) and then the ANGLE to draw and the distance to ADVANCE.

If you prefer relative rather than absolute angles, TURN can be used to rotate a specified number of degrees. LCIRCLE and LELLIPSE draw circles/ellipses or arcs around the Logo cursor.

The Toolkit commands offer a mix of useful and informative commands.

MEM displays in decimal and hex how much memory your program takes and what is spare. This is based on your current mode, so put yourself into the correct mode when using it.

CHAR is a superb com-

mand enabling you to design/edit VDU23 characters. There are dozens of programs which do this, but this one can be used while you're actually writing your program.

GOODPROG attempts to mend a Bad Program, while FKEYS displays on screen the contents of your function keys. Both work well, and, because Addcomm becomes a part of your computer they sit there waiting to be used in crisis situations.

The same applies to LVAR, which lists all the variable names you've used. This can really help to avoid a muddle.

VERIFY is a facility that was missed out on the Electron. It checks that what has been saved on tape exactly matches the same program in memory.

LLIST is a line listing command. Unlike ordinary LIST, LLIST can be included in a program. Silly? Not at all. My error routine, when debugging programs is now:

```
ON ERROR
NODE6:REPORT:PRINT"at
line";ERL:LLIST ERL:END
```

This tells me the error and displays the problem line.

Using FIND it is possible to list the whole program, including the possibility of different list formats such as multi-statement lines broken up.

Whenever I program I like to use meaningful variable names. It makes life easier for me, but the computer can handle the resident integer

```
10REM POOLS SELECTOR
20REM By Rog Frost
30REM
40REM
50MODE1
60rnd=RND(-TIME)
70VDU19;4;0;
80PRINTTAB(1,10)"Auto selection of twenty four numbers"TAB(10,13)"for football pools"TAB(4,20)"Press space bar to get numbers"
90REPEATUNTILGET=32:CLS
100DIM choiceX(55)
110FORselectionX=1TO24
120numberX=RND(55)
130IFchoiceX(numberX)<>0
THEN120
140choiceX(numberX)=numberX
150NEXT
160PRINT
170FORprintoutX=1TO55
180IF choiceX(printoutX)<>0
PRINT printoutX
190NEXT
```

Program I: Long pools selector

```
10MODE1:RX=RND(-TIME):VDU19;4;0;:PRINTTAB(1,10)"Auto selection of twenty four numbers"TAB(10,13)"for football pools"TAB(4,20)"Press space bar to get numbers":REPEATUNTILGET=32:CLS
20DIM CX(55):FORSX=1TO24
30NX=RND(55):IFCX(NX)<>0
THEN30
40CX(NX)=NX:NEXT:PRINT:FORPX=1TO55:IF CX(PX)<>0
PRINT PX
50NEXT
```

Program II: Compacted

From Page 25

variables (A%-Z%) more quickly and efficiently.

Using the Addcomm command GREPL I can ask for a variable such as *pos_of_frog* and change it right through the program to *F%*.

SREPL finds each variable you might want to change and then gives you the option to change it or not.

I can now use variables that I understand and when the program is fully de-bugged change to short fast variables.

Not only that, I can use KILLREM to remove all REM lines and then the superb COMPACT to combine lines, thus saving more memory and processing time.

Programs of mine that I have put through the processes save about one third of the original memory - and run more quickly in to the bargain.

Programs I and II are identical in what they do (selecting football pool numbers) but Program II has had its variables changed and its REMs removed.

It has then been compacted and renumbered. Table I shows the differences, which are quite dramatic. Imagine the effects on long programs.

So far all has been very good, but three of the miscellaneous commands are dreadful. POPGOS, POPREP and POPFOR allow you to jump out of subroutines, REPEAT/UNTIL loops and FOR/NEXT loops.

All of these make for very bad programming and should be avoided.

Another command, LGOTO, is like GOTO but instead of jumping to a specified line number you jump to a label.

It works fine, but so far I've had no particular use for it. SETWIN allows you to predefine seven text windows which can then be called with

the WIN command. It's easier than VDU 28.

The last miscellaneous command is SORT. This can be used to sort string arrays into alphabetical order.

In truth, it actually produces Ascii order with upper case letters before lower case. It's a very fast sort taking just 0.75 seconds to sort 100 words into order.

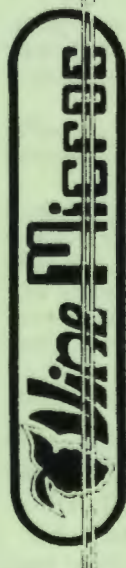
To sum up, Addcomm is brilliant, particularly the graphics and the toolkit, but there is a small price to pay in terms of memory.

It requires 256 bytes of RAM for its own use and so it pushes page up to &F00. If you are going to load a long commercial program it's as well to turn Addcomm off. To do this type ADDCOMM and then hit the Break key. *FX163 will turn it on again.

Firmware like Addcomm requires good documentation, and Vine Micros has achieved a winner here. The 72 page manual is clear, concise and simple giving full syntax and examples for each of the forty commands.

Addcomm is the most valuable piece of software I've got. Every Electron owner who does some programming should consider getting it. It's easily worth the £28.

Rog Frost



Available from:

VINE MICROS, MARSHBOROUGH,
NR. SANDWICH, KENT,
CT13 0PG.

(Or your local Computer Dealer)

	Program I	Program II
Time taken	2.1 seconds	1.9 seconds
Program length	394 bytes	246 bytes
Variable storage	291 bytes	232 bytes
Spare memory	7763 bytes	7970 bytes

Table I: Addcomm advantages

Reprinted from "The Electron User" June '85