

Computing *with the* AMSTRAD

MAY, 1987

\$4.50

A Database Publication

The independent magazine for Amstrad computer users
Registered by Australia Post - Publication No. VB097411



MORE!

On Basic
On Graphics
On Sound
On Machine Code

MORE!

Software Reviews
Listings
News
Book Reviews

MORE!

On Desktop Publishing
On Mini Office II
On CPM
On Public Domain

Computing With The Amstrad

May 1987

Published monthly by Strategy
Software under license from
Database Publications Ltd.

Telephone: [002] 29 4377
Telex: AA 58134
Attn. HT163

Mail: P.O. Box 11
Blackmans Bay
Tasmania 7152

Annual Subscription (12 issues)

Australia: \$ 45
South Pacific (inc. NZ) \$A65

'Computing With The Amstrad'
welcomes program listings and
articles for publication. Material
should be typed or computer
printed, and preferably double
spaced. Program listings should be
accompanied by cassette tape or
disk. Please enclose a stamped
addressed envelope or the return of
material cannot be guaranteed.
Contributions accepted for
publication by Database
Publications or its licensee will be
on an all-rights basis.

©Database Publications and
Strategy Software. No material may
be reproduced in whole or part
without written permission. While
every care is taken, the publishers
cannot be held legally responsible
for any errors in articles, listings or
advertisements.

'Computing With The Amstrad' is
an independent publication and
neither Amstrad Consumer
Electronics plc or Amsoft or their
distributors are responsible for any
of the articles in this issue or for
any of the opinions expressed.

Contents

May 1987

6 FIRST STEPS
PETE BIBBY LOOKS AT
HOW TO GIVE VALUES TO
VARIABLES AND TELLS YOU
ALL YOU WANTED TO KNOW
ABOUT READ AND DATA
COMMANDS.

10 CPM
THE SERIES CONTINUES
WITH A LOOK AT THE DYNAMIC
DEBUGGING TOOL.

14 U.K. NEWS
ALL THE LATEST FROM
THE U.K. INCLUDING HIGH
FINANCE, A NEW PRINTER AND
RECENT RELEASES.

17 GRAPHICS
NOW IT'S TIME TO GET A
LITTLE MORE AMBITIOUS AND
EXPLORE THE GRAPHICS
SCREEN. THE COMMANDS
MOVE, DRAW, MOVE AND
DRAW ALL COME UNDER
SCRUTINY THIS MONTH.

23 MACHINE CODE
THE FIFTH PART OF THIS
HIGHLY PRAISED SERIES
UNCOVERS THE REMAINING
Z80 REGISTERS AND SHOWS
HOW THEY CAN BE USED AS
SIMPLE MACHINE CODE
VARIABLES.

28 LITTLE PEOPLE
EVALUATING THE LATEST
RELEASE FOR THE AMSTRAD
FROM ACTIVISION, WE REVEAL
THE WONDERFUL WORLD OF
THE FOLK WHO LIVE INSIDE
YOUR MICRO.

31 SOFTWARE SURVEY
CONTINUING ON FROM
OUR LOOK AT THE LITTLE
PEOPLE WE HAVE FIVE MORE
PAGES OF REVIEWS WITH
SOMETHING TO PLEASE
EVERYONE WHETHER YOU
OWN A CPC, PCW OR EVEN ONE
OF AMSTRAD'S NEW PCS!

37 UTILITY
DISCOVER WHERE THOSE
ELUSIVE VARIABLES ARE
LURKING WITH THIS UTILITY
FOR BASIC PROGRAMMERS.

41 BUSINESS SECTION
CHECK THE INDEX ON
PAGE 41 TO SEE WHAT'S IN
STORE FOR BUSINESS USERS.

57 SOUND
AS WE GO FURTHER
DOWN MELODY LANE, YOU'LL
FIND THAT MAKING AND
BREAKING A RENDEZVOUS IS
CHILD'S PLAY FOR YOUR
AMSTRAD.

60 GAME OF THE MONTH
PIT YOUR WITS AGAINST
THE MICRO IN THIS THOUGHT
PROVOKING RECREATION OF
THE TRADITIONAL BOARD
GAME OF BACKGAMMON.

66 PUBLIC DOMAIN
JOIN SHANE KELLY IN
ORGANIZING YOUR DISKS
WITH THESE USEFUL CPM
UTILITIES.

68 POSTBAG
THE PART OF THE
MAGAZINE YOU WRITE
YOURSELVES.

72 ORDER FORM
THE BIT WHERE WE
CHASE YOUR MONEY!

76 BOOKMARK
A LOOK AT FIVE BOOKS
ON LOCOSCRIP.

78 MY SAY
SOME IMPORTANT NEWS
PERHAPS?

**SEPERATE
BUSINESS INDEX -
PAGE 41.**



THIS month we're leaving strings behind us and looking at how we can use READ and DATA to give values to variables. This may not sound too exciting, but by the end of the article you should be able to see how useful it can be.

To start with, have a look at Program 1. All this does is to give, or assign, values to the numeric variables x, y, and z, add them and print the answer. It's easy as 1,2,3.

```
10 REM Program 1
20 LET x=1
30 LET y=2
40 LET z=3
50 LET sum=x+y+z
60 PRINT sum
```

Program 1

All right, so it's hardly going to win the Program of the Year award, but what it lacks in complexity it makes up in efficiency. It works and it makes a point.

What I want you to notice is the program's rigidity. If I now

wanted to add 6, 7 and 8 using the same structure I'd have to rewrite lines 20 to 40. As you can see from this, assigning values to variables using simple LET statements can be

PETE BIBBY looks at how to give values to variables

fairly inflexible.

Of course there are other ways of assigning values to variables. We've come across two of them already. Take a look at Program 11, which uses the less obvious of the two methods.

```
10 REM Program 11
20 sum=0
30 FOR loop=1 TO 3
40 sum=sum+loop
50 NEXT loop
60 PRINT sum
```

Program 11

Here we're using the (I hope) familiar FOR...NEXT loop to add 1, 2 and 3. The loop cycles three

times.

The first time round, *loop* is equal to 1 while *sum* is initially 0. The result is that line 40 stores 1 (1+0) in *sum*. Next time round *loop* is 2, while *sum* starts off as 1 and ends up as 3. I leave it to you to figure out what happens to *sum* during the third cycle when *loop* is 3.

So, we've used *sum* to hold the running total of successive values of *loop* and hence added 1, 2 and 3.

Incidentally, you don't really need line 20, the Amstrad automatically assumes that numeric variables are 0 until it's told otherwise. Not all micros do this, however, and some come crashing to a halt if you haven't already given a variable a value, even if it's only zero.

So, even if line 20 is redundant in this case, it's a good habit to get into. And your Amstrad's got enough memory to cope with a little bit of redundant code.

While all this may seem a bit longwinded when just adding 1, 2 and 3, try adapting Program 11 to add up all the integers (whole numbers) from 1 to 1000. You'll see that it beats the first program's method hands down.

The trouble is that while using a loop control variable to give values to a running total is both efficient and adaptable, it is a bit limited. If you think about it you'll see why.

Since the loop control variable increases by the same amount each time round, the values of *loop* are in a regular pattern. It's easy to add, say 1, 3 and 5 or 4, 8, 12 and 16 using this method. All you do is vary the STEP parameter of the loop to allow the loop control variable to take the required values. That's easier to do than to read about! In the case of Program 11 you'd use:

30 FOR loop=1 TO 5 STEP 2

in the first case and:

30 FOR loop=4 TO 16 STEP 4

for the second.

The problem arises when you try to use the method to add 1,7 and 23. So while Program II may be better than Program I, it still has its drawbacks.

Program III shows a much more flexible method of getting information into a program. It has you actually typing it in at the keyboard when the Amstrad requests it.

```
10 REM Program III
20 sum=0
30 FOR loop=1 TO 3
40 INPUT "Number? ",number
50 sum=sum+number
60 NEXT loop
70 PRINT sum
```

Program III

When you run Program III you'll see how it can handle adding 1, 7 and 23. In fact it's so flexible that it can add any three numbers that you think of, provided that they're in the Amstrad's range.

It's the INPUT of line 40 that gives the program this marvelous adaptability. However nothing in life is all good, and this use of INPUT does have its drawbacks, adaptable though it may be.

The major one is that it holds up the program until you respond to the request for keyboard input. And imagine trying to add a thousand numbers using this method...

Also you have to input all the numbers every time you run the program. One error typing in your responses and you have to go right back to the beginning again, or indulge in some quick mental arithmetic.

So each of the methods used in our first three programs seems to have a drawback. Wouldn't it be nice if there was a way to give values to variables that was flexible, wouldn't take any numbers, and wouldn't involve keying things in while the

program is running?

Have a look at Program IV, which meets all these criteria.

```
10 REM Program IV
20 sum=0
30 FOR loop=1 TO 3
40 READ number
50 sum=sum+number
60 NEXT loop
70 PRINT sum
80 DATA 1,2,3
```

Program IV

As you can see, it has added up the numbers in the last line, our old favorites 1,2 and 3, and printed out the result. What's interesting is the way in which it's done it.

Before we get to that, let's have a look at the familiar bits of Program IV. As before, line 20 isn't strictly needed but it is good programming practice. Lines 30 and 60 are identical to those of Program III, forming a FOR ... NEXT loop that cycles three times.

Even line 50 is the same, using the variables *sum* and *number* to hold a running total. And line 70 just prints out the value of *sum* when the program drops out of the loop. So what's the big difference between Programs III and IV?

The answer is line 40. Here, instead of the old familiar INPUT, we've got a READ. And we've also got a new line, line 80, with its DATA statement and 1, 2 and 3.

However it's not all that different. In Program III, the INPUT of line 40 was there to get a value for *number*. You had to type it in at the keyboard.

In Program IV line 40's READ does exactly the same thing, getting a value and storing it in *number*. Only now the Amstrad doesn't look to the keyboard for the next value of *number*, it looks to line 80 where the data is held.

In other words, where INPUT has the micro looking outside the program for values for a variable,

READ has it looking to another part of the program itself where the required values are stored.

The first time round the loop *sum* is 0. Line 40 tells the Amstrad to READ a value from the data line and put it in the variable *number*.

Obediently the micro searches through the program for the first occurrence of a line beginning with DATA and takes the first value it finds there. In this case it is 1. Then the program goes onto line 50 which works in the usual way, storing the running total, 1, in *sum*.

The second cycle of the loop sees *sum* starting off with the value 1. Line 40 again tells the computer to READ a value for *number*. As before, it looks to the line beginning with DATA, but this time it takes the second item it finds, in this case 2. So *number* becomes 2 and, after line 50, *sum* becomes 3.

The third time round the loop the READ of line 40 has the Amstrad looking for the third item in line 80's data list, the figure 3. This is stored in *number* and line 50 sees *sum* becoming 6. The loop is now ended and the PRINT of line 70 gives the answer.

Notice that line 80 - the one beginning with DATA - does nothing but hold the data. Without the READ somewhere to make use of it, it's barren.

To sum up, READ causes the program to look to a line beginning with DATA and to store the values that it finds there in the variable that follows the READ command. The micro keeps track of where it is up to in the data list and every time it obeys a READ command it takes the next unused value.

In other words, it moves sequentially along the list of data, never using the same one twice.

As we said before, the READ command works exactly like the INPUT command except that instead of looking at the keyboard the computer looks in the program

itself for the value to be assigned to the variable.

In a way it's a combination of the best features of all three of the above methods, but without sacrificing any flexibility.

This adaptability comes from the fact that if we want to give the program different values all we have to do is to change the numbers after the DATA of line 80. So to add up 56, 78 and 345 we would alter it to read:

```
80 DATA 56,78,345
```

while:

```
80 DATA 11,22,33
```

gives us the sum of 11,22 and 33.

We don't have to have all of our data in one line. Program V uses READ and DATA to add up 10 numbers.

```
10 REM Program V
20 total=0
30 FOR cycle=1 TO 10
40 READ figures
50 total=total+figures
60 NEXT cycle
70 PRINT total
80 DATA 1,2,5,34,6,7
90 DATA 3,67,54,1,2
```

Program V

You should have no problem seeing that the method used is the same as in Program IV. However as we are now adding up 10 numbers the loop cycles 10 times, each time **READING** in more data.

Notice that now there are two lines beginning with DATA, line 80 and 90. This is easier for us to follow, two lines of five numbers looking better than one of 10.

To the Amstrad it makes no difference. When it comes across a **READ** it starts at the first item of data after the first DATA it finds and then carries on doggedly in sequence. In this case it **READs** 12, then 5,34,6 and 7 in order. The sixth time round the loop line 60's **READ** demands another value for figures.

Undaunted by having reached the end of line 80, the Amstrad carries on, finds the next line beginning with DATA, and takes the first value following that. In this case the line is line 90 and the first value is 3. The next four cycles of the loop have the micro **READING** in 67,54,1 and 2 in turn.

You'll have noticed from the last two programs that the commas in the lists of data act as separators. They come between the numbers, telling the micro where one item of data ends and another one starts. Obviously they're very important and if one is left out or misplaced the consequences can be dire.

To see what I mean, try changing line 80 of Program V to:

```
80 DATA 125,34,6,7
```

This is what would result if we missed out the comma between the 12 and 5 when typing in the program. Now when you run it you get the message:

```
DATA exhausted in 40
```

and the program stops.

What has happened is that the **FOR...NEXT** loop has tried to cycle 10 times as directed. The first nine times there's no problem as there are, after our mis-typing, nine items in the data list.

The tenth time round the loop, line 40 tells the micro to **READ** another value from the data list but there are only nine there. The result is a nervous breakdown and the program grinds to a halt with the aforementioned:

```
DATA exhausted in 40
```

message as some form of explanation.

But it's not a very good explanation. It points to the wrong line. After all, we made the mistake in line 80 and the program ran out of data items at the end of line 90, yet the message says the problem is at line 40.

In a way it's true, because the

problem occurred when the micro tried to obey the **READ** of line 40 for the tenth time. So it can be justified.

However it's still a nuisance. Half the problems people have with typing in programs from Computing with the Amstrad occur when they mis-type something in DATA statements. The program crashes and the error message points to a **READ** that's miles away from the offending data.

The poor typist keeps looking at the **READ** and knows that it's correct even though the message tells him the fault lies at line 40. Hence they give up in despair, assuming the program doesn't work.

The moral is, if you come across an error message that points to a line with **READ** in it, check out the DATA statements carefully.

And if that potentially misleading error message wasn't enough, you can make another typing mistake in a DATA line that doesn't give a message at all. To see what I mean try changing line 80 to:

```
80 DATA 1,2,5,34,6,7
```

Here, instead of leaving out a comma and "glueing" two numbers together as before, our errant typist has added a comma between the 1 and the 2 of what was supposed to have been 12. Now if you count the number of data items in lines 80 and 90 you'll find there are 11. But the loop is only going to cycle 10 times so what's going to happen? Run the program and see.

The result is that the program works but it gives the "wrong" result. Actually the result isn't really wrong, you'll find that 180 is the result of adding together the first 10 items of data. The trouble is that the first 10 items of data weren't the 10 items of data we wanted to add! Our slipshod typist has messed it up.

This kind of mistake can be a real problem. Where there are too few items of data in a list then the

Amstrad will tell you as it did before. The error message may be a bit misleading, but it will be there.

However if there are more data items than expected, all the micro does is take the number it wants and ignores the rest. There's no error message.

In our case the program returned the value 180 from the faulty data, instead of the 191 we'd have got if the typist hadn't been daydreaming. The computer has done its sums correctly, but it's done them with the wrong numbers. And without any error message it's easy for the mistake to slip through.

There's another problem that doesn't show up in this short program. The Amstrad keeps track of where it is up to in the data list by means of what is known as a data pointer. All this does is to "point" to the next item of the data list to be used. Now when our typist has made line 80:

```
80 DATA 1,2,5,34,6,7
```

Program V appears to work correctly and stops. The data pointer is pointing at the next item in the data list, 2, but as the program has ended, it is never used.

In longer programs, however, there can be more than one READ command. If Program V did have another READ command tucked away somewhere then the first data item it would take would be the one that the data pointer is indicating, that is, 2.

Because of the inadvertent comma all the succeeding READs will be one data item out. This can cause havoc in a long program and can be very hard to track down.

The moral of the above is type in DATA lines carefully.

So far we've had our data lists at the end of the program. However they don't have to be there, as you can see from Program VI in the next column.

Here the DATA lines are all over the place. This makes no difference to the Amstrad as its data point keeps track of things. However it's good programming practice to keep your data lists together. This is usually at the end of short programs which might have several different data lists, in logically appropriate places.

```
10 REM Program VI
20 accumulator=0
30 DATA 12,23
40 FOR loop=1 TO 7
50 DATA 34,45,56
60 READ fromsomewhere
70 accumulator = accumulator +
  fromsomewhere
80 NEXT loop
90 DATA 67,78
100 PRINT accumulator
```

Program VI

Hence you may find all the data on salaries with the subroutine that calculates pay and all the data on employees' names and addresses near the subroutine that prints out the payslips.

While you've got Program VI in your micro, change line 50 so that it reads:

```
50 DATA 34,45,56,
```

that is, add a comma after the 56. Now run the program and see what happens. Instead of our previous total of 315 the result is 237. Yet we're still using the same data. Or are we? The difference between 315 and 237 is 78, which is the last item of data in line 90. Is it more than coincidence?

The answer, of course, lies with the commas we added to the end of line 50. As we saw before, commas are used as separators between data items. When the Amstrad comes across a comma it expects some data immediately after it. And if there's none there it takes it to be the value 0.

This is what's happened when we added the comma to the end of line 50. Even though there's no data

there, the Amstrad has assumed that there is and given *fromsomewhere* the value 0. And, of course, this effectively adds an item to the middle of the data list so there are now eight instead of the previous seven. The loop only cycles seven times so poor old 78 gets left out.

If you find that hard to understand add:

```
65 PRINT fromsomewhere
```

and you'll see the 0 tucked between 56 and 67. Once again the moral is take care with commas in DATA lines.

And that's all we're going to cover for this month. I hope that you'll have had a glimpse of how powerful and versatile the READ and DATA commands can be. And also how careful you have to be with data lists.

We'll be dealing with them again next time, but for the present I'll leave you with Program VII.

```
10 REM Program VII
20 sumone=0:sumtwo=0
30 FOR loop=1 TO 5
40 READ first,second
50 sumone=sumone+first
60 sumtwo=sumtwo+second
70 NEXT loop
80 PRINT sumone,sumtwo
90 DATA 1,2,3,4,5
100 DATA 6,7,8,9,10
```

Program VII

Here the READ is followed by two variables, first and second. How does it work? Does first take its values from the data in line 90 while second uses line 100? Or is it done another way? We'll find out next time.

In next month's First Steps, Pete Bibby digs deeper into our investigation of READ and DATA, makes you aware of some of the pitfalls, and looks at their associated command - RESTORE.

END

Dynamic debugging

ONE 8080 utility supplied on Amstrad's CP/M disk which is useful, if only in the absence of anything better, is DDT.COM which stands for Dynamic Debugging Tool.

This is a CP/M debugger which allows us to load files from disk and control program execution as well as examine and change memory. Load and run DDT by typing its name and after it has signed-on issue the command:

-d0

to display C0h bytes of memory starting at 0000h. The first number in a line is the address of the first byte displayed - 16 bytes from that address onwards are then displayed in hex.

At the end of the line are the Ascii characters which the values in memory might represent - if the value of a byte is not a legal Ascii character then a dot "." is printed instead.

This Ascii field makes it easier to spot and read blocks of text. You can examine a disk file by loading it into memory either from within DDT by using the I and R commands as follows:

-istat.com This initializes an FCB.

-r This reads the file specified in the FCB.

or by typing its name as a parameter on the command line when you invoke DDT:

A>ddt stat.com

or:

A>ddt test.hex

The file is then loaded into the TPA as if it were a program about to execute and DDT displays some status information of the form:

NEXTPC
1580 0100

NEXT tells us the first memory location free above the program which has been loaded and PC tells us the address pointed to by the user program counter and is used as the default if you enter a command such as D without specifying an address. Note the value of NEXT - you will need to use it to work out the number of pages the program takes up if you intend to save any changes you might have to make.

To exit DDT just type Ctrl+C and you can then use the CCP save command to write your file in memory to disk if you have made any changes you want to keep.

If you ask DDT to load a .HEX file, such as the output from assemblers like ASM and ZSM, DDT will convert it to binary first and then load it into the appropriate place.

The G command will execute code - you can specify the entry address and up to two breakpoints. For example, typing:

-g

will execute code starting at whatever address the current value of the user program counter is pointing at while:

-g100,124,3c00

will execute code starting at

address 100h, but will first insert a breakpoint instruction (RST 6) at addresses 124h and 3C00h.

DDT keeps track of the contents of the processor registers as used by the program being debugged - you can examine and change any of these with the X command. For example, typing:

-x

will display the current contents of all the registers. Typing:

-xp

will display the current value of the user program counter. If you then just press Enter it will not be changed but if you type a hex number then the user PC will be changed to this new value.

Similarly xa will let you alter the accumulator, xb, xd and xh will let you alter the double register BC, DE and HL respectively and xs lets you alter the Stack Pointer.

The T command will trace through the program, displaying the contents of the processor registers after each instruction has been executed. Typing:

-t

will trace a single instruction. Typing:

-t20

will trace 20 h (32 decimal) instructions - note that all numbers which you give DDT are taken to be hexadecimal. The associated command u or Untrace will execute the specified number of instructions but will only display the register states once, at the end.

This can be useful if, for example, you simply want to check the register contents on entry to and exit from a subroutine and don't want the screen cluttered up with all the intermediate results.

Unfortunately there is an important limitation on the

usefulness of the trace commands - DDT is an 8080 utility and so only understands the subset of the Z80's instruction set which the old 8080 was able to execute (this was explained in more detail in last month's article).

This means that if we try to trace a program written specifically for a Z80, DDT will almost certainly meet an instruction which it does not understand. The consequences of this are unpredictable, but tend to be brief, spectacular and fatal.

So remember if you intend to debug a program using DDT you must either write in 8080 assembler and use ASM, or if you have a Z80 assembler, restrict yourself to using instructions for which there are 8080 equivalents - see the table in last month's article for a list of these.

The same restriction applies to another useful facility, DDT's line assembler. Typing:

-a200

will put you into assembler mode and redisplay the address - 200h in this example - at which you wish to ORG your code. You can then type in a sequence of assembler instructions such as:

```
0200 mov a,b
0201 out 24
0203 ret
0204
```

and DDT will insert the appropriate machine code into memory. Just pressing Enter on its own gets you out of assembler mode. As you may have noticed the line assembler must be given 8080 mnemonics, not Z80.

The I command does the opposite of this - typing:

-I100

will display a disassembly of a section of code starting at the specified address. Again this will be displayed in 8080 assembler mnemonics and extra Z80 instructions will not be recognized - they will be printed as ??-to show

DDT's confusion.

The s or Substitute command allows you to step through memory, altering bytes if you wish. Typing:

-s100

will redisplay the address - 100h in this case - and will also show the value of the byte at that address. Just pressing Enter will step on to the next byte without changing the first one and typing a hex number will substitute it for the old value before stepping on to the next location. This continues until you enter a full stop.

To move a block of memory type a command such as:

-m100,2ff,6d00

This copies the block starting at 100h and ending at 2FFh inclusive to an area starting at 6D00h. The original memory area is not altered. To fill an area of memory with a value, for example 0FFh, type:

-f1000,1fff,ff

which will fill the area starting at 1000h and ending at 1FFFh inclusive with the value 0FFh.

DDT has the ability to perform elementary hexadecimal arithmetic - typing a command in the form:

-hx.y

where x and y are hex numbers will make DDT display the sum and the difference of the two numbers.

Use DDT to examine the SPA before and after loading a file and identify the information it contains. Look at any programs you have written and practice tracing and executing the code using breakpoints. You'll be making a lot of use of DDT or something very similar if you start writing programs to run under CP/M, so it is well worth learning how to use it properly.

There are many other debuggers

around- CP/M Plus, for example, comes with a more powerful one called SID - but all work in very similar ways to DDT. Often even the commands are the same - some can just do more than others.

Several are available in the public domain and Rick Surwilo's powerful Z80 debugger Z8E (SIG/M library volume number 239) stands out here. But beware the inevitable trade-off in a small system such as ours - the more powerful a debugger is, the bigger it is.

When you run a debugger the first thing it does is to relocate itself up into high memory just below the BDOS. When you then load other programs it fools them into thinking that it is part of the BDOS and so is not overwritten by them. However this trick means that the TPA available to other programs is reduced in size by the length of the debugger.

If you are only debugging small routines this should not be a problem, but if you wish to write and test large programs on an Amstrad CPC 464 with a normal TPA of under 40k bytes you will probably want the smallest debugger you can find.

Having described the structure of CP/M and some of the facilities and utility software available to us we'll use the knowledge to help us write routines and programs which will run under CP/M on our Amstrads, and exploit its capabilities.

Last month I introduced the concept of BDOS function calls, which are routines inside the BDOS we can use to carry out particular functions for us by calling them in a standard way. These functions fall naturally into two groups - those which are concerned with disk operations and those which are not.

We'll start with the latter group as they're simpler. Figure 1 lists the 13 non-disk function calls with any parameters which they expect to receive as well as those which they will return. We'll examine

TOOLBOX

Available now for Amstrad CPC464
ONLY!

Toolbox consists of four utilities:

Ansprit & Ansprite form two outstanding tools for creating screen pictures and animated sprites with the option for easily re-creating and using these in your own Basic programs. Using these two, you can easily add graphics to your adventure games, or write an arcade standard action game with smooth pixel motion of your characters.

Ansmon is a very useful utility which allows you to type machine-code programs into memory and examine the contents of either RAM or ROM. Experienced users will know that addresses 0 to 3FFF hex are used for both RAM and ROM. Normally, reading from these addresses will access RAM (e.g. when using PEEK), so special action must be taken to read from ROM. This is catered for by **AMSMON**.

FCOPY can be used to make backup copies of any cassette which conforms to the Amstrad file format - this includes programs saved using any of the usual SAVE commands, both Basic and machine-code. **FCOPY** is NOT to be used to copy programs which you have not purchased as this is illegal under Australian copyright legislation and you could be subject to fines of up to \$250,000 or five years imprisonment.

**Toolbox is available
now from Strategy
Software on TAPE
ONLY and priced at
\$19.95.**

**Please use the order
form on Page 72.**

the functions individually next month, but for the moment we'll look at the way these powers are invoked.

As an example let's use function number 2 - Console Output, often abbreviated to CONOUT. We would use this function to print a character on the screen.

From Figure 1 we see that the BDOS expects us to put the character we want to print into register E, and will return nothing - in other words we won't get any error messages back if things go wrong as we will with more complex functions.

If we remember from last month that we also need to put the number of the function we want into register C and execute a CALL instruction to address 0005 of the SPA we can write the short assembler program on the right to print an asterisk on the screen.

Notice how we finished the program - the:

```
jp warm
```

instruction executes a jump to address 0000h in the SPA, which contains another jump into the BIOS to perform a Warm Boot and brings us back to the CCP prompt. End

No.	Function	Input parameters	Output Parameters
0	System reset	None	None
1	Console input	None	A-char
2	Console output	E-char	None
3	Reader input	None	A-char
4	Punch output	E-char	None
5	List output	E-char	None
6	Direct console I/O	E=0FFh(input) E-char(output)	A=00h(no char) A-char(char read) A=(IOBYTE)
7	GetIOBYTE	None	None
8	SetIOBYTE	E=(IOBYTE)	None
9	Print string	DE=addr.of buffer	None
10	Read console buffer	DE=addr.of buffer	(buffer)=chars
11	Get console status	None	A=00h(no char) A=0FFh(char read)
12	Return version number	None	HL=Version No.

Figure 1: Non-disk BDOS function calls

```
; Program to demonstrate use of BDOS
; function 2 - CONOUT - by printing
; a # on the screen.
;
;          org 100h          ; Start at base of TPA
;
; conout   equ 2             ; Function number for CONOUT
; warm     equ 0000h         ; Warm boot address at start of SPA
; bdos     equ 0005h         ; BDOS entry point in SPA
;
; start    1d c,conout       ; Get function number into C
;          1d e,'#           ; Character into E
;          call bdos         ; Call BDOS routine to print character
;
;          jp warm           ; Finish properly when we come back
;                               ; from BDOS - execute a warm boot
;                               ; and restart CCP
;
;          end
```


MTBASIC

Multitasking BASIC Compiler

Softaid's MTBASIC is the Basic Compiler for MS-DOS, PC-DOS and CP/M-80 Systems

Features

MULTITASKING - Multitasking is an important new feature for microcomputers. Before MTBASIC, multitasking was only available on mini's and mainframes. With MTBASIC, up to 10 Basic programs (tasks) can run concurrently, giving minicomputer performance to the micro user. Using MTBASIC's multitasking features, you can create games that continue to compute even while awaiting input from the user, create your own print spoolers, and write nearly any application requiring multiple simultaneous operations. MTBASIC performs all of the context switching; no special hardware or software is needed. Although MTBASIC does support all hardware interrupt modes, it will automatically generate software interrupts if no source of hardware interrupts exists. A complete set of Basic statements to control tasks and multitasking is provided.

WINDOWING - MTBASIC can create up to ten windows on the screen at the same time. Each separate task, or multiple windows can be assigned to each task. One window can be overlaid by another, and overlaid windows can later be restored, making pop up and pop

down menus a breeze to implement. A complete set of MTBASIC window control statements are provided. Windowing works equally well with all operating systems.

FAST CODE - MTBASIC generates highly optimized object code. Byte's September, 1981 Sieve runs in 26 seconds on a 4 Mhz Z-80 (10 iterations). Interpreters take over 1400 seconds. MTBASIC typically compiles over 100 Basic statements per second, so most programs compile in a second or two and then run 50 times faster than interpreted Basic code.

INTERACTIVE - Because MTBASIC runs interactively, like an interpreter, debugging is quick and easy. To the programmer, MTBASIC appears to be a tremendously fast interpreter. MTBASIC programs are entered with line numbers and each line's syntax is checked as it is entered. Unlike interpreted Basics, however, when RUN is typed, MTBASIC compiles the program to object code and executes it. The code generated by MTBASIC can be stored to disk files and executed stand alone. No runtime license is required to distribute object code.

MTBASIC produces ROMable code. Stand alone code, stored to disk file, can be downloaded to ROM to run in target systems. No runtime license is required to distribute the ROMed code.

FUNCTIONS: Multi-line, user defined functions, ACOS, ADR, ASC, ASIN, ATAN, BAND, BOR, BXOR, (binary AND, OR & XOR), CHR\$, CONCAT\$, COS, CVI, CVS, ERR AND ERR\$ (disk error processors), EXP, INP, KEY, LEN, LOG, MID\$, MKI\$, MKSS\$, PEEK, RND, SIN, SQR, STR\$, TAN, VAL.

STATEMENTS: CALL (Call an assembly routine), CANCEL (stop a task), CHAIN, CLOSE (file), CLS, CODE (in line machine code), CURSOR, DATA, DEF (define function), DEFSEG (8088 only), DELETE (file), ERASE, EXIT (leave a task), FEND (end function definition), FIELD, FILE, FOR/NEXT, PRINT (formatted printing - more powerful than PRINT USING), GOSUB, GOTO, IF-THEN, INPUT, INPUT\$, INTEGER, INTMODE (set Z-80 interrupt mode), INTON & INTOFF (interrupts on/off), JVECTOR & VECTOR (hardware interrupt vector definitions), OFF ERROR, ON ERROR, OPEN, OUT, POKE,

PRINT, PUT, RANDOMIZE, READ, REAL, REM, RETURN, RGET, RUN (start a task), SEEK (random files), STOP, STRING, TASK, TRACEON, TRACEOFF, WAIT (suspend a task for a specified amount of time), WFRAME (draw an outline around a window), WINDOW (create a window), WSAVE (save a window), WSELECT (select a window), WUPDATE (update a window).

MTBASIC supports single precision floating point with 6 decimal digits of accuracy, long variable names, sequential and random disk files, strings and string arrays, and automatic mixed mode arithmetic.

FULLY RECURSIVE-MTBASIC is fully recursive, permitting development of sophisticated algorithms which previously could only be coded in PASCAL.

Specially configured for your Amstrad and available by mail-order only from Strategy Software - Make Your Amstrad Fly!

The MTBASIC package includes all the necessary software to run in interpreter or compiler mode, an installation program (so any system can use windowing), three demonstration programs, and a comprehensive manual.

\$125

Strategy Software

P.O. Box 11

Blackmans Bay

Tasmania 7152

[002] 29 4377

\$125

Aiming for \$325 million

Amstrad heading for record profits

AMSTRAD is riding a boom that looks like leading to ever-escalating profits in the months to come.

Market analysts are predicting that when the last balance sheet of the current financial year is revealed this summer it will show pre-tax profits of at least \$325 million.

The just-published half yearly report, showing profits up 159 per cent to more than \$177 million, immediately added 50¢ to Amstrad shares.

Now analysts are predicting a

Trademark crackdown

A CRACKDOWN by Amstrad on the illegal use of its name has claimed its first victim. Lawyers acting for the company have successfully applied for an injunction against a Midlands dealership calling itself Amstrad Computers Limited.

Issued by the High Court in London, it ordered the defendants, Michael John Hancox and his company, to cease use of the name Amstrad and also not to infringe the trademark.

In addition the court ruled that the defendants must pay all Amstrad's legal costs and that damages be assessed.

Now representatives of Herbert Smith, the highpowered London firm of solicitors representing Amstrad, are reported to be investigating 20 other cases involving abuse of the company's name or trademark.

One of these involves Crown Computer products of Latham, Lancs, which is marketing an anti-glare screen under the name of Amsfilter.

The company has been informed that Amstrad believes this infringes its rights because of the

similar increase over the next few months which will take Amstrad into the GEC-Plessey-Racal league as a multi-million pound electronics company.

One expert told Computing with the Amstrad: "These shares are trading at between \$4 and \$4.25. Even so, they are very much undervalued."

"I expect them to reach their proper level of around \$5 without much difficulty, and in doing so put Amstrad's value at over two and a half billion dollars."

"Current predictions for pre-tax

use of the 'Ams' prefix.

"We are taking action where the name Amstrad - or part of that name - is being used improperly", said a spokesman for Herbert Smith. "After all, our clients have built up considerable goodwill around their name and trademark and this must be protected."

In most cases the Herbert Smith team will simply write a letter pointing out an infringement to an offending company and asking them to desist.

But where the breach is considered to be flagrant a writ will be issued.

Then if the infringement is proved the company served in this manner will find itself faced with considerable costs.

Should a defence be entered it would become necessary to engage a leading counsel to appear in the High Court. His fee alone could be in the region of \$10,000 a day (No - it's not a misprint!).

"Unless you have substantial capital it would take a very brave man with a very good case to fight Amstrad in the courts", said the director of one firm whose knuckles have been rapped.

profits of at least \$325 million by the end of the current financial year could be too low by \$25 to \$50 million.

"Amstrad is showing an extremely vigorous growth rate at the moment, and there is no reason I can see for it to let up."

"The company may be struggling to establish itself in the US, but the boom situation in Europe and elsewhere will more than compensate."

"And Alan Sugar's promise of a continuing supply of new products, backed by Amstrad's proven marketing skills, means the trend should continue well beyond the end of the current financial year".

New printer launched

Amstrad selected the Which Computer? Show in Birmingham to launch a modem for its market leading PC and a new dot-matrix printer.

Developed by Pace Micro Technology, the multi-speed modem comes in the form of a card containing both the hardware and software.

Priced at \$375, it will enable any IBM-compatible machine to be used for communications.

Bundled with the modem is free membership to MicroLink, the UK's fastest growing electronic mail service, which also provides access to telex.

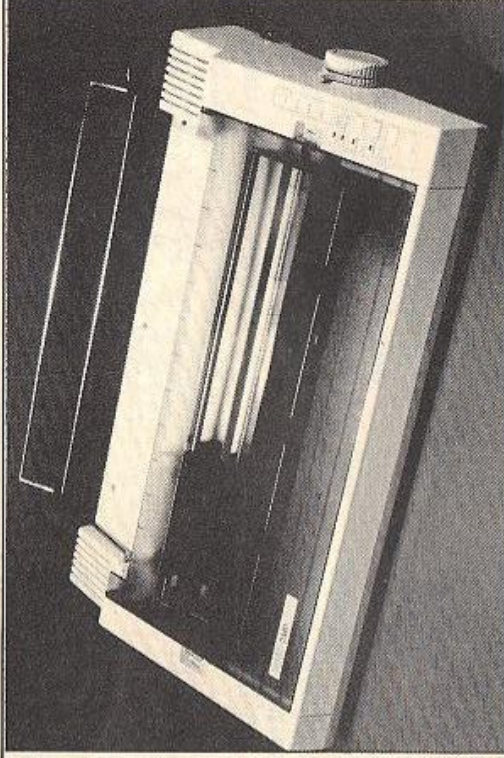
The modem card is simply inserted into one of the slots available in the PC1512.

Amstrad's latest printer - the DMP4000 - will carry a price tag of \$875. Though aimed at the PC1512 range, it is also compatible with the Amstrad CPC series, which have standard parallel printer output.

With a 15 inch wide carriage and a printing speed of 200 character, it offers more than 100 typeface combinations.

These include near letter quality coupled with a complete Ascii and IBM graphics character set.

*Amstrad's
DMP4000 printer*



Keep out the cuppas

A CUP of tea accidentally spilled over his micro keyboard inspired lawyer Dorian Young's creation of a unique Amstrad product.

His Seal'n Type protective cover for PCW and PC machines is now being marketed by Kado Enterprises, the firm he has set up with his wife Kate.

"I devised the flexible plastic cover after spilling a cup of tea all over my keyboard while working on my PhD", he told Computing With The Amstrad.

"The response so far has been very encouraging. One customer, an archaeologist, said it would be ideal for protecting his keyboard from dust while he was on a dig.

Court saves CPC

AMSTRAD enthusiast and Prestel contributor Paul Needs' favorite CPC6128 has narrowly survived an attempted zapping by British Telecom.

He was found guilty on four charges at Port Talbot magistrates court and fined \$1000 plus \$320 compensation for "dishonestly abstracting electricity" from BT.

But an application by BT that Needs' CPC6128 be destroyed was

denied by the court.

Needs had admitted breaking into the PSS system using an unauthorized account number - one used for demonstration purposes by BT's PSS marketing department.

His CPC was confiscated at the time of his arrest and he had to buy another to maintain his Amstrad user news service on Micronet.

"Now I have my original micro back I'll have to sell one of them to help pay my legal costs".

MUSIC COMPOSER - Superb fun for all the family. Suitable for the beginner or experienced musician. Clear graphics displays the music as it is composed. Full editing facilities. Compositions may be joined together and/or saved on tape/disk.

TAPE \$17.95
DISK \$29.95

See Page 72 for
ordering
information

EASIDATA III is a completely revised and vastly improved version of our machine-code database. This new version includes: sorting on up to 9 levels at any one time; searching on any combination of fields; a powerful report generator allowing you to produce reports in the layout you require; facilities to restructure your database at any time.

TAPE \$41.45
DISK \$53.45

EASIWORD is an incredibly fast and powerful professional Word Processor. It is easy to use and has all the facilities you would expect of the modern office Word Processor. There is a print module which will allow you to set up 52 of your own print commands, customizing **EASIWORD** for use with your own printer.

TAPE \$37.95

DISK \$49.95

DATABASE/MAILING LIST

Easy to use and flexible electronic card index. Up to 50 items per record. Up to 360 records. Fast sort and search facilities. Simple label printing routine with multiple copy facility. Large files may be split into smaller files.

TAPE \$29.45
DISK \$41.45

EASIDATA II - The ultimate database program for your CPC464. Up to 32 items per record. Up to 1,000 records. Full search facilities, including wild card. Records automatically inserted in correct order. Built-in report generator allows simple word processing with mail merge and label printing.

TAPE ONLY \$29.95

Strategy
Software

New on the games front

NOW CPC and PCW owners can take part in their own Hollywood B-movies thanks to the latest Infocom release from Activision.

Uncle Buddy Burbank and Aunt Hildegard are the stars of Hollywood Hijinx with the player inheriting their estate.

However, this can only be claimed if 10 treasures hidden in the Malibu home are discovered in one night. Price \$62.50 (can this be right?).

Also new from Infocom is a compendium for the PC1512 based on the award winning Zork games.

The trilogy is set in an underground labyrinth with players searching for the Treasures of Zork.

Many enemies are encountered in the process including the Wizard of Frobozz, Dimwit Flathead and the Dungeon Master.

In a different vein, you can take to the water against 15 other nations to win the greatest of all ocean races.

Sailing, a yachting simulation for the CPC range, is based on the Americas Cup race with players choosing their yacht and the country they wish to represent.

A FOLLOW-UP to its very popular Secret Diary of Adrian Mole has been released for the PCW and CPC range by Virgin Games.

The Growing Pains of Adrian Mole is based on the bestseller by Sue Townsend and users must try to make Adrian as popular as possible with everyone.

Price \$25 on tape, \$37.50 on disk.

TWO new games for the CPC range have been announced by Martech.

Pulsator is based on a maze and incorporates five levels, each of 49 rooms.

On each level a Pulsy must be rescued by finding the key which unlocks the door to its prison.

Aliens are encountered and these become progressively more intelligent and will deplete your power if they touch you. Price \$22.50 on tape, \$37.50 on disk.

Nemesis the Warlock is based on the popular comicstrip character and follows his fight against Torquemada, Grand Master of the Terminators.

LATEST addition to the MiniBus range of budget titles from Bubble Bus Software is Hi Rise for the CPC series.

In a scaffolding network, players must paint all the areas while warding off the attacking Nasties. Price \$10 tape, \$25 on disk.

NEW from Alligata is a two-game title for the PCW range.

Blagger incorporates 20 screens with users playing master burglar. Five keys must be collected per screen in order to open the safe and proceed to the next level.

In Guardian players must shoot down waves of baters, landers, swimmers and mutants in order to rescue the humanoids. Price \$37.50.

A SEQUEL to its bestselling range of games featuring Monty the Mole has been released for the CPC range by Gremlin Graphics.

Auf Wiedersehen Monty follows our hero as he flees for his life across Europe attempting to raise enough

money to buy the Greek island of Montos.

There he can escape extradition as no-one else knows of the island's existence.

Travelers cheques and other valuables must be acquired along the way and finally Monty must travel to Zurich to set up a Swiss bank account. Price \$25 on tape, \$37.50 on disk.

A NEW multi-level game for the CPC range is available from Hewson.

RanaRama is set in a labyrinth of rooms which are viewed from above to maximize the effect.

Users control Mervyn, a sorcerer's apprentice apprentice, as he tackles 12 guarded wizards through eight levels to obtain valuable missiles, power and shields. Price \$22.50 on tape, \$37.50 on disk.

THREE new games for the CPC, PCW and PC have been announced by Ariolasoft.

The Fourth Protocol for the PCW and PC is based on the film starring Michael Caine and is a story of intrigue and counter-espionage. Price \$40 for PCW, \$50 for Amstrad PC.

Their Finest Hour for the CPC is a wargame challenging users to control RAF Fighter Command and prevent the Luftwaffe from gaining air superiority.

Bride of Frankenstein for the CPC challenges the user to revive husband-to-be Frankenstein before the wedding day.

Various human parts must be obtained by robbing graves and crypts while avoiding the wandering lost souls. Price \$22.50 on tape, \$37.50 on disk.

Plotting imaginary points with an invisible cursor

Having thoroughly explored the text screen and its associated windows in the last couple of articles, we can now turn our attention to the graphics screen.

As with the text screen at switch-on the graphics screen covers the whole screen area (save, of course, the border area). It can be used with any of the three screen modes.

In Part V of our series GEOFF TURNER and MICHAEL NOELS move away from the windows and concentrate on the graphics screen

The first thing we need to become familiar with is the graphics screen coordinate system, as shown in Figure 1. We've already met one system of coordinates, those of the text screen. However these are designed primarily to cope with the positioning of characters on the text screen, a task the system handles admirably.

We need more precision when specifying points on the graphics screen. So for graphics purposes we divide it into a 640 by 400 grid. We specify a point by giving its horizontal position first, followed by its vertical position - much as you learned at school when doing graphics.

Note that the graphics origin is situated at the bottom left hand corner of the screen and is identified as (0,0). The X axis (horizontal) ranges from 0 to 639 and the Y axis (vertical) ranges from 0 to 399.

This is completely different from how things work on the text screen, where the origin is at the top left of the screen and the numbering starts at 1.

The figures for the graphics screen apply whichever mode we happen to be working in, and the physical location of any graphics point will therefore be the same in every

mode. That is (0,0) will always be at the bottom left of the screen and (639,399) will always be at its top right.

Let's try drawing some lines on the screen. To do this we imagine something called the graphics cursor, which is invisible yet occupies a specific point on the screen.

MOVE is used just like the DRAW command, by specifying X and Y coordinates. So the command:

MOVE 639,399

will move the cursor across the screen to the top right hand corner, but this time no line will drawn.

Time to put theory into practice with some simple programs. Program I draws our diagonal line again, followed by a second line down the right hand edge of the screen, and finally a third line back to the starting position to form a triangle.

By the way, we have included in this program a dummy last line to prevent the text cursor and "Ready" message reappearing on the screen after the drawing is completed. This helps to keep a tidy graphics screen for the purpose of these demonstration programs. Of course you will need to press Escape twice to exit the program.

Now to illustrate how the MOVE command is used, change line 40 to:

40 MOVE 639,0

When you now run the program you will see that the triangle is incomplete, as we have only moved the cursor down the right hand side of the triangle, instead of drawing a line.

Let's try a more complex shape

Initially when we switch on or reset the computer the graphics cursor will always reside at the origin point (0,0). We can use the DRAW command to draw a line from the cursor's present position to any other point on the screen. For instance if we enter the command:

DRAW 639,399

a line will be drawn diagonally across the screen from position (0,0) to (639,399).

We can also position the graphics cursor at any point on the screen, without actually drawing a line. The command to do this is MOVE, and it is used with the same coordinate system as the DRAW command. So we may either draw from the cursor's current position to any other, or alternatively we can simply move the cursor between the two points.

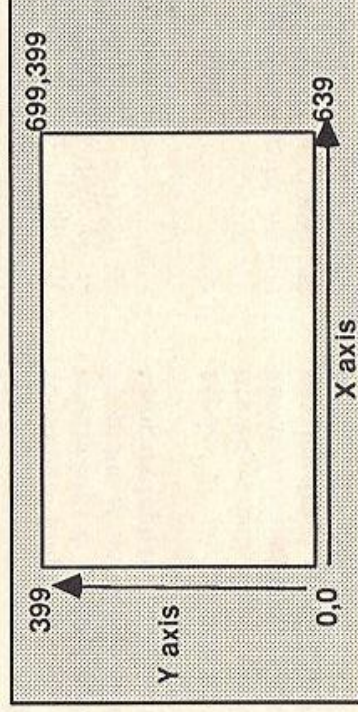


Figure 1: Graphic Screen Co-ordinates

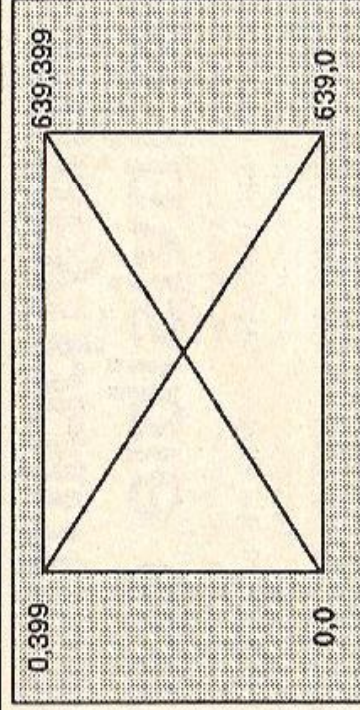


Figure II: Output of Program II

now. Program II forms a rectangle, with intersecting corner to corner diagonals as shown in Figure II. See how we have used a combination of MOVE and DRAW commands to produce this shape.

You may have noticed, however, that the drawn lines appear slightly thicker than when we used Mode 1. The diagonal lines appear to have a stepped effect instead of appearing as a smoother straight line.

Of course this shape cannot be produced using DRAW alone (without drawing over some line twice), so we need to throw in the occasional MOVE command to complete the shape.

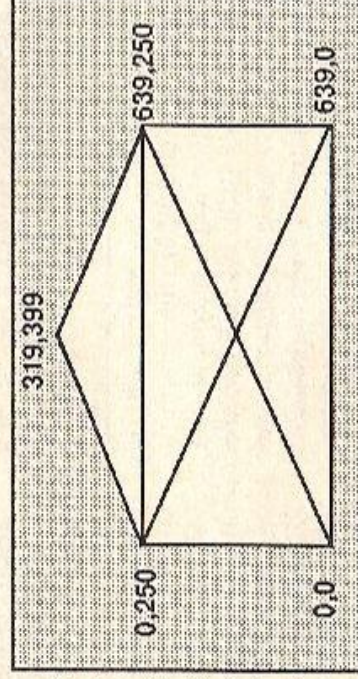


Figure III: Output from Program IV

By way of an exercise, see if you can reproduce the envelope shape shown in Figure III using only DRAW commands and no MOVE commands. It's something that we have all tried to do at some time with pen and paper, but now we can attempt it with a computer. We promise you it can be done!

Programs I and II have used Mode 1 to draw the shapes, but we can also use the other two Modes, 0 and 2. You'll remember that when we were dealing with the text screen, the mode controlled the number of characters we could print across the screen. The mode command also has an effect on the graphics screen, but in a slightly different way. Change line 20 in Program II to:

20 Mode 0

and run the program again. As you will see, the shape is drawn in exactly the same way, and it occupies the same position and area on the screen.

restricted our use of color on the text screen? Unfortunately the same rules apply to using color on the graphics screen.

The high resolution graphics of Mode 2 are only available in two colors. Therefore if we consider one color for the background we can only really draw lines in one other color. Of course we can select various colors from the 27 available, just like we did with the INK command in previous chapters.

Mode 0 can be used to produce multicolored drawings in 16 colors, but we lose the fine detail of the higher resolution screens.

So it's really a case of choosing the best mode for your particular application.

We measure screen resolution in pixels, each being a separate dot of color on the screen. The more of these dots you can cram onto the screen the finer the lines we can draw, the better the resolution.

Table I shows the relationship between the screen resolutions. You can see that the vertical resolution is always 200 points or pixels, while the horizontal resolution varies according to the mode.

MODE NUMBER	HORIZONTAL PIXELS	VERTICAL PIXELS
0	160	200
1	320	200
2	640	200

Table I: Mode - pixel relationship

screen mode the individual points are much smaller and the lines appear thinner.

If we want to use really high resolution graphics then we can select Mode 2. Try using Mode 2 in Program II and see the effect of using the high resolution screen.

Mode 2 allows us to create drawings with much finer detail, but as you might expect we have to pay the price for this in terms of color.

Remember how the mode

Notice that one pixel doesn't necessarily correspond to one screen coordinate point. Sometimes neighboring screen coordinate points have to share a pixel. The more points sharing a particular pixel the less the resolution will be.

However the important point to remember is that the coordinates used in DRAW and MOVE commands are always 0 to 399 vertically, and 0 to 639 horizontally, regardless of the mode selected. This may seem a little confusing at first, but you

will soon find that it works out much easier just using one set of coordinates instead of the three sets shown in Table I.

Just to make sure that you have grasped the effects of different modes, enter Program III. This steps through each mode and draws a line to the centre of the screen. Points to note are that the same coordinates are used for each mode, and that the thickness of the line varies with each mode. Line 60 simply introduces a short delay between each change.

So far all our lines have been drawn in bright yellow on a blue background. These are the normal default colors whenever we switch on the computer. Let's try introducing a change of color into our line drawing.

```
10 REM PROGRAM III
20 FOR resolution = 0 TO 2
30 MODE resolution
40 PRINT "MODE"; resolution;
50 DRAW 320,200
60 FOR delay = 1 TO 2000:NEXT
70 NEXT resolution
80 GOTO 20
```

Program III

In the previous examples we didn't actually specify any colors, and so the computer assumed that we just wanted to draw in the default color. If we wish to change the drawing color we tell the computer by adding a third parameter after the DRAW command.

Reset the computer, and enter the following commands:

```
CLG
DRAW 639,399,2
```

Now our diagonal line will appear in bright cyan. You see, the third number after the DRAW command tells the computer which graphics pen to use. After issuing this command any further lines will still be drawn in bright cyan until we specify a new pen with the DRAW command.

It's very similar to selecting a text pen. Once selected, it remains in

use until a further PEN command is used. You'll see if you enter;

```
DRAW 639,0
```

the second line is still drawn in bright cyan, even though we didn't specify a color.

Now enter the command:

```
DRAW 0,0,1
```

and the color will be restored to bright yellow to complete the triangle.

Program IV draws that envelope shape that you may have been struggling with earlier. This time we're using Mode 0 so that we can introduce a good selection of color into the proceedings. You will see that each line is drawn in a different color because we have appended a color parameter to the end of each DRAW command.

```
10 REM PROGRAM IV
20 MODE 0
30 DRAW 639,0,2
40 DRAW 639,250,3
50 DRAW 0,0,4
60 DRAW 0,250,5
70 DRAW 639,250,6
80 DRAW 319,399,7
90 DRAW 0,250,8
100 DRAW 639,0,9
110 GOTO 110
```

Program IV

Try omitting some of the color parameters in this program to see how the overall color of the drawing is affected.

While we are on the subject of color, let's consider how we can change the color of the background for our graphics screen.

The CLG command is used to clear the graphics screen. It works in much the same way as the CLS command, which clears the text screen.

However, if you try CLG in direct mode (that is, just type it into the Amstrad and press Enter) you'll see that the Ready prompt appears on a line corresponding to

the last position of the text cursor - not at the top of the screen. We may also append a pen number to this command in order to change the background color.

```
10 REM PROGRAM V
20 MODE 0
30 FOR color=0 TO 15
40 CLG color
50 PRINT "CLG";color
60 FOR delay=1 TO 1000:NEXT
70 NEXT
```

Program V

Reset the computer, and enter:

```
CLG 3
```

Now as the graphics screen is cleared it changes color to red, the pen number specified. Other colors can be selected as long as they are within the range allowed by the particular mode in use. This principle is very similar to changing the paper color on the text screen, but of course the entire area of the graphics screen must be changed at once, as you have to clear it to change color.

Program V demonstrates how the background color may be changed with the CLG command. Note that the text paper color is unaffected by this command, and that the text cursor is not returned to its home position as it is when a CLS command is issued.

You might like to try inserting some different background colors in some of the other programs. A word of warning though. Take care to avoid drawing lines over a background color with the same foreground color. The lines will still be drawn, but you just won't see them!

Some very interesting effects may be obtained by introducing random elements into our drawings. We could select random colors and also draw or move to random coordinates.

In Program VI we are drawing lines in random colors to random locations. After each line is drawn we move the cursor back to the

origin point (0,0) and the end result is a series of lines radiating from the bottom left hand corner.

```

10 REM PROGRAM VI
20 MODE 0
30 WHILE INKEYS=""
40 color=INT(15*RND(1))+1
50 x=INT(639*RND(1))+1
60 y=INT(399*RND(1))+1
70 MOVE 0,0
80 DRAW x,y,color
90 WEND
    
```

Program VI

Suppose we now leave out line 80 so that the cursor is not moved back to the origin. What effect do you think this will have? Try it and see.

Alternatively we could move the cursor to another position instead of the origin. Try using these various options at line 80, and run the program again:

```

80 MOVE 320,200
80 MOVE x,0
80 MOVE x,y
80 MOVE 0,y
80 MOVE y,y
80 MOVE x,x
    
```

We can also produce some interesting patterns by leaving out the random functions and taking a more logical approach. Program VII again makes use of the DRAW and MOVE commands, but this time our lines are drawn in a more orderly fashion.

We start off with two fixed points (x,y) and (a,b) at extreme corners of the screen and draw a line between them. Then the values of the four variables used are changed in steps of 8 units before the next line is drawn.

There are many variations on the same theme. Program VIII is another example. Perhaps you may like to try writing some of your own routines similar to these. Try and predict what the finished pattern will look like before you run each program.

There are many effects to be obtained by creative use of the

MOVE and DRAW commands. To use the computer cliché, you are limited only by your own imagination.

You should by now be familiar with the graphics screen coordinates as used with the MOVE and DRAW commands. There are, however, two related commands which use a slight variation on the graphics screen coordinate system.

These two new commands are MOVE and DRAW. Notice that we have simply added an R to the end of MOVE and DRAW. The R stands for RELATIVE, and it simply means that the moving or drawing should be relative to the present position of the graphics cursor.

```

10 REM PROGRAM VII
20 MODE 0
30 FOR color=1 TO 15
40 x=0
50 y=0
60 a=639
70 b=399
80 WHILE y<400 AND x<640
90 MOVE x,y
100 DRAW a,b,color
110 x=x+8
120 y=y+8
130 a=a-8
140 b=b-8
150 WEND
160 NEXT color
    
```

Program VII

Let's start off with a clear graphics screen, then we know that the cursor will be situated at (0,0). If we now issue the commands:

```

DRAW 200,200
DRAW 400,0
    
```

the first line will be drawn to a position 200 points across the screen and 200 points up from the base line. The second line is drawn from the end of the first line, to a position 400 points across the screen, and to vertical point 0 - that is, back to the base line. We end up with a sort of inverted V shape.

Notice that all the coordinates

used in the above commands are with reference to the origin point (0,0).

10 REM PROGRAM VIII

```

20 MODE 0
30 x=0:y=200
40 a=639:b=200
50 WHILE a>320
60 MOVE x,y
70 DRAW a,b,2
80 a=a-8:b=b-4
90 WEND
100 WHILE x<320
110 MOVE x,y
120 DRAW a,b,3
130 x=x+8:y=y+4
140 WEND
150 WHILE a>0
160 MOVE x,y
170 DRAW a,b,4
180 a=a-8:b=b-4
190 WEND
200 WHILE x<639
210 MOVE x,y
220 DRAW a,b,5
230 x=x+8:y=y-4
240 WEND
250 WHILE a<320
260 MOVE x,y
270 DRAW a,b,6
280 a=a+8:b=b+4
290 WEND
300 WHILE x<320
310 MOVE x,y
320 DRAW a,b,7
330 x=x-8:y=y-4
340 WEND
350 WHILE a<640
360 MOVE x,y
370 DRAW a,b,8
380 a=a+8:b=b-4
390 WEND
400 WHILE x>0
410 MOVE x,y
420 DRAW a,b,9
430 x=x-8:y=y+4
440 WEND
450 GOTO 30
    
```

Program VIII

They are known as ABSOLUTE coordinates.

Let's try the same thing again using our two new commands. First of all enter a CLG command to restore the cursor back to the origin, then enter these commands:

```

DRAWR 200,200
DRAWR 400,0
    
```


This time the lines have been drawn relative to the graphic cursor's last position. The first line is, in fact, identical to the one in the previous example, because the cursor was initially situated at (0,0) - our CLG saw to that.

However the second line is completely different. It's been drawn horizontally from the end of the first line and ended up at a point with absolute coordinates (600,200).

What has happened is that the second line has been drawn relative to the last position of the cursor instead of the origin point. In fact the two sets of coordinates have been added together like this:

$$\begin{aligned} 200 + 400 &= 600 \\ 200 + 0 &= 200 \end{aligned}$$

and so we have ended up at point (600,200). The same rules apply when using the MOVER command. If we entered the commands:

CLG

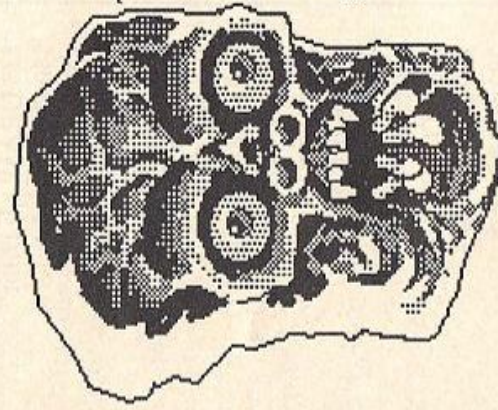
MOVER 150,200
MOVER 100,150

then if our maths are correct the cursor would end up at point (250,350).

So you see it is just as easy to use relative coordinates as it is to use absolute ones. The only problem is that you can easily wander off the edge of the screen when using relative coordinates, so you need to keep track of exactly where you are moving the cursor to.

You might like to try substituting the MOVE and DRAW commands in some of the programs in this article with MOVER and DRAWR. If you use the same coordinates, the shapes produced will differ greatly from the originals, but you should be able to adjust the values in order to restore the correct shapes.

In theory any shape can be drawn using either the absolute or the relative coordinate system. There are two points to remember. First of all, negative coordinates are allowed, so the commands:



There's something to look forward to in EVERY issue of Computing With The Amstrad!

Next month in CWT A....

All the regular series on Graphics, Sound, Machine Code, CP/M, Public Domain PLUS Software Survey, Your letters, 16 page Business section and more!

New series on PCW graphic handling, dBase II v Condor, Nucleus review, Search utility for CPC machines and Aleatoire looks skyward to find the South pole! There's a terrific new game to play and lots of surprises - don't miss it!

There's much, much more in EVERY issue of Computing With The Amstrad - order yours NOW!

CLG
MOVER 200,200
MOVER -200,-200

would move the cursor out to (200,200) and back again to (0,0). In fact negative numbers are almost a necessity when drawing with relative coordinates.

Program IX demonstrates this point by drawing a square using relative coordinates.

the second point to bear in mind is that it is possible to move the cursor off the screen by using coordinates greater than those normally available.

We have so far only used coordinates within the range (639,399), but it is quite legal, and sometimes necessary, to specify numbers greater than these.

Enter the commands:

CLG
DRAW 1000,1000

and the computer will quite happily draw a line to an imaginary point (1000,1000). Of course the line will disappear off the screen. To prove that the cursor has in fact gone to the specified point enter another command:

DRAW 500,0

and you will see that the line reappears as though it had in fact come from point (1000,1000).

ZOOM TECHNIQUE IS VERY HANDY

Program X demonstrates how off-screen coordinates may be used. The program begins by drawing a triangle small enough to fit on the screen. Now if you press the + key the triangle is crased and redrawn slightly larger. Keep on pressing the + key and eventually the triangle will be drawn outside the limits of the screen perimeter. It is in fact being drawn using off screen coordinates.

Use the - key to reduce the triangle down to its original size. This zoom technique is often used to vary the size of images onto the screen in order to show more detail in complex drawings.

```
10 REM PROGRAM IX
20 MODE 1
30 MOVE 300,300
40 DRAWR 200,0
50 DRAWR 0,200
60 DRAWR -200,0
70 DRAWR 0,200
```

Program IX

```
10 REM PROGRAM X
20 MODE 1
30 color=1
40 x=100:y=100
50 a=400:b=300
60 c=400:d=100
70 GOSUB 110
80 IF NOT INKEY(28) THEN
  GOSUB 170
90 IF NOT INKEY(25) THEN
  GOSUB 250
100 GOTO 80
110 REM draw triangle
120 MOVE x,y
130 DRAW a,b,color
140 DRAW c,d
150 DRAW x,y
160 RETURN
170 color=0
180 GOSUB 110
190 x=x+8:y=y+8
200 a=a+8:b=b+8
210 c=c+8:d=d+8
220 color=1
230 GOSUB 110
240 RETURN
250 color=0
260 GOSUB 110
270 x=x+8:y=y+8
280 a=a+8:b=b+8
290 c=c+8:d=d+8
300 color=1
310 GOSUB 110
320 RETURN
```

Program X

ZOOM TECHNIQUE IS VERY HANDY

Anyway that's enough for this month. Try plotting some shapes of your own with DRAW, MOVE, DRAWR and MOVER. Next month we'll move onto plotting individual points.

Now for some more registers

A C B E H L

SO far in this series we've concentrated on the A register, a single byte internal memory location within the Z80 processor itself. However, I have hinted darkly that there are other registers available, and we'll be looking at some of them this month.

The registers we're interested in are the B, C, D, E, H and L registers. Each of these, like the A register, can store a single byte.

There are other registers, such as the F register, but for the moment we'll just concentrate on the half dozen I've listed.

(Incidentally, there are no G, I, J or K registers.)

Since they are single byte, or eight bit registers, we can load them directly with a number in the range 0 to 255, just as we did register A.

That is, commands such as:

LD B,&2A

and

LD H,&0C

exist.

We can represent this type of operation, where we load a register with a byte, symbolically as:

LD r,n

where: LD stands for Load, r stands for one of the registers A, B, C, D, E, H, L and n stands for a number in the range 0 to 255. We refer to such numbers as constants.

Table I shows the opcodes for each of these register loads, including LD A,n which we're already familiar with.

From Table I you can see that:

LD B,&2A

would translate into:

06 2A

whereas:

LD H,&0C

would give:

26 0C

Right, so we now know how we can load the various registers with numbers - but why would we want to do so?

In Part V of his introduction to Machine Code MIKE BIBBY reveals the secrets of the remaining Z80 registers

Well, you'll know from your programming in Basic how valuable variables are for keeping track of things. In machine code we haven't got that luxury. However, we can use the registers as rudimentary variables.

LD B,n	06 n
LD C,n	0E n
LD D,n	16 n
LD E,n	1E n
LD H,n	26 n
LD L,n	2E n
LD A,n	3E n

Table I: LD r,n opcodes

Even though each single register can only hold a number in the range 0 to 255, you can still achieve some powerful results. After all, even Basic was written

in machine code!

As an example of the sort of thing I mean, let's locate the text cursor at a given column and row on the screen and then print an asterisk there.

Fortunately there's a firmware routine that allows us to do just that - it's located at &BB75. I'm going to call it PosTCur, short for Position Text Cursor.

To use it, we load the H register with the column we want and the L register with the row we want. We then call PosTCur.

Having positioned our cursor, we still have to print the asterisk. To do this we simply load the A register with the Ascii code for asterisk (&2A) then call our CharOut routine at &BB5A to print it out.

While we're christening routines, let's call the routine at &BB6C - which clears the text screen - clrText. We use it in the program below before positioning our cursor, printing the asterisk and then returning to Basic.

address	hex code	mnemonics
3000	CD 6C BB	CALL clrText
3003	26 14	LD H,&14
3005	2E 0C	LD L,&0C
3007	CD 75 BB	CALL PosTCur
300A	3E 2A	LD A,&2A
300C	CD 5A BB	CALL CharOut
300F	C9	RET

In this program the H and L registers have clearly been used as variables to specify where we want our text cursor positioning. Try altering the values you give H and L to place the cursor in different positions on the screen.

Of course, as I've said, we can likewise load the other registers with numbers in the range 0 to 255.

The simplest way to demonstrate this would be to load the register in question with a number and then to poke it into Hexer's "workspace" (&2FF8 - &2FFF), where we could examine it at leisure.

As you'll recall from last month, this works well enough for the A register:

```
LD A,n
LD (&2FF8),A
RET
```

Here you load A with the number you require (shown here as n - an eight bit number) and then load memory location &2FF8 with the contents of A. In effect, you're poking the A register's contents into &2FF8.

You'd then look at &2FF8 to see that it really does contain what you'd loaded A with.

Unfortunately, this doesn't work with the other registers, because you can't poke into memory with a register other than the A register. So:

```
LD B,n
LD (&2FF8),B
RET
```

fails because there is no such instruction as:

```
LD (&2FF8),B
```

More formally,

```
LD (pq),r
```

does not exist except where r is the A register. Here pq refers to a two byte, 16 bit number. Remember, we're specifying a memory address inside those brackets, so we need two bytes.

Similarly, you can't peek with registers other than A, so:

```
LD B,&2FF8)
```

is also illegal. More formally,

```
LD A,(pq)
```

is the only peek allowed.

TASWORD 464 THE WORD PROCESSOR

WE HAVE A LIMITED NUMBER LEFT AND ARE SELLING THESE AT ONLY \$36.95 ON TAPE AND \$48.95 ON DISK. MANUAL IS COMPLETE WITH ADDENDUM TO ENABLE TRANSFER TO DISK. HURRY! THESE WON'T LAST. WORKS ON 664/6128

TASWORD 464

The Word Processor

A
Tasman Software Program
for the
Amstrad CPC 464

Strategy
Software

ONLY 15 LEFT!

BOX 11, BLACKMANS BAY, TASMANIA 7152

See order Form on Page 72 for further details

obvious reason, check to see if you're wrongly assuming that some of your CALLs to other subroutines are preserving the registers.

In practice you'll tend to find some registers preserved and some not. The two routines we've relied on most heavily, CharOut and CharIn, preserve all the registers, fortunately.

Even PostCur leaves some registers unchanged - the B, C, D, E registers. We can take advantage of this by transferring our "sensitive" register values (A, H, L) into these unchanged registers for safe keeping, transferring them back when we need them.

By using these ideas, we can successfully recode our previous routine to allow us to place two asterisks on the screen, one directly below the other.

We've used B to store H, C to store L and D to store A. Later we'll discover easier ways of getting round the problems of corrupt registers. As it is, the above example certainly illustrates operations with out eight bit registers.

address	hex code	mnemonics
3000	CD 6C BB	CALL CInText
3003	26 12	LD H, &12
3005	44	LD B, H
3006	2E 0C	LD L, &0C
3008	4D	LD C, L
3009	CD 75 BB	CALL PostCur
300C	3E 2A	LD A, &2A
300E	57	LD D, A
300F	CD 5A BB	CALL CharOut
3012	60	LD H, B
3013	69	LD L, C
3014	2C	INCL
3015	CD 75 BB	CALL PostCur
3018	7A	LD A, D
3019	CD 5A BB	CALL CharOut
301C	C9	RET

By working out what's happening you'll definitely master the ideas. Why not try printing out a column of three asterisks? B, C, D will still be preserved, remember.

Last month we saw that we could add and subtract numbers from the A register. You might be tempted to think that since we could INC and DEC the other registers we could ADD and SUB them also. Unfortunately not.

ADD B,8

and

SUB C,4

do not exist!

You can, however, ADD the contents of a register to the A register.

In both cases, the answer is left in the A register.

Formally,

ADD A,r

and

SUB r

exist, where r is any register of B, C, D, E, H, L, A.

Notice, by the way, that when you're adding, you have to specify A followed by the register you're adding to it. When you're subtracting you only need to specify the register you're subtracting - the micro automatically knows you're taking it from the A register.

r	ADD A,r	SUB r
B	80	90
C	81	91
D	82	92
E	83	93
H	84	94
L	85	95
A	87	97

Table IV: ADD A,r and SUB r opcodes

Incidentally, the way the answers end up, or accumulate, in the A register explains its alternative name; the accumulator. And, of course, since the answers do end up in the A register, they're handy

for poking into memory (Remember we do our eight bit pokes into memory via the register.)

The opcodes for ADD and SUB are given in Table IV. You'll need them to do the following sums Problems 1 to III. Don't try them before you've read the whole of the article, though, as in a moment I'll give you what my math teacher used to call a worked example.

The answers to each of the sums is sorted in &2FF8, so you can examine them easily with option 2 of Hxer. You should be able to follow what's happening without too much trouble.

Remember, if you add 1 to &FF the answer is 0 when you're dealing with eight bit numbers. Similarly, if you take 1 from 0, the answer is &FF. As we saw last month, the numbers cycle round.

Notice that I haven't spelt out the actual hex opcodes for you - you'll have to do that yourself. This is called hand assembling the code. Tables 1 to V give all the opcodes

QWERTY

The typing trainer

\$14.95 Tape

\$27.95 Disk

Please note tapes
are only suitable
for 464

What we can do, though, is to adopt a roundabout approach, loading the number into the register under examination, then loading the A register with the contents of that register and poking the A register into memory.

The code would be something like:

```
LD B,n
LD A,B
LD (&2FF8),A
RET
```

It relies on a new instruction:

LD A,B

This allows us to load register A with the contents of register B. In fact, this is just one of a whole set of opcodes that allow us to transfer the contents of one register to another. We describe this class of opcodes as:

LD r,r'

where both r and r' are any of the register A, B, C, D, E, H, L;

Table II shows the opcodes - there are quite a few!

When you're using Table II, remember that, as it's set out, the first register (r) specifies the row and the second register (r') gives the column it's in. Hence the opcode for LD B,C is 41.

You'll probably have noticed a pattern. If not, it might help you to see it if I tell you that there's both a row and a column missing between the L and A registers. We'll see what that means later.

You'll see such a pattern in many of our opcodes - it helps the Z80 to work out what the opcode's telling it to do.

By the way, LD r,r' copies the contents of r' into r. In doing so, it doesn't alter the value in r', so when the micro's finished the opcode, both r and r' will have the same value.

To demonstrate some of these opcodes in action, we can play a game of "Chinese Whispers" with

our registers. Let's load H with &2A, then pass it on from register to register until we reach the A register, when we'll print it out with CharOut.

address	hex code	mnemonics
3000	26 2A	LD H,&2A
3002	6C	LD L,H
3003	5D	LD E,L
3004	53	LD D,E
3005	4A	LD C,D
3006	41	LD B,C
3007	78	LD A,B
3008	CD 5A BB	CALL CharOut
300B	C9	RET

	r'						
	B	C	D	E	H	L	A
B	40	41	42	43	44	45	47
C	48	49	4A	4B	4C	4D	4F
D	50	51	52	53	54	55	57
E	58	59	5A	5B	5C	5D	5F
H	60	61	62	63	64	65	67
L	68	69	6A	6B	6C	6D	6F
A	78	79	7A	7B	7C	7D	7F

Table II: Opcodes for LD r,r'

So far we've learned to do quite a bit with our registers: we can load them directly with numbers, and we can copy numbers from one register to another.

There are other operations we can perform on the registers. For instance, we can INC and DEC them. INC A as you'll recall from last month, increases the value of the A register by one. DEC A decreases its value by one.

INC r

and

DEC r

where r is any of registers B, C, D, E, H, L, have similar effects on the specified registers.

Table III contains the opcodes you need for INCREMENTING and DECREMENTING our eight bit registers. As we found out last month, none of them affect the carry flag.

To illustrate their use, we can incorporate them into our previous

code, to place not one but two asterisks on the screen, the second directly below the first.

Since L contains the row you might think that the way to do it would be to run our code as normal up to printing the first asterisk, then INC L, to give us the row below (think!), call

r	INC r	DEC r
B	04	05
C	0C	0D
D	14	15
E	1C	1D
H	24	25
L	2C	2D
A	3C	3D

Table III: INC r and DEC r opcodes

PostCur again, then print another asterisk, as in:

```
CALL ClrText
LD H,&12
LD L,&0C
CALL PostCur
LD A,&2A
CALL CharOut
INCL
CALL PostCur
CALL CharOut
RET
```

Unfortunately this won't work, as calling PostCur actually changes the values that the H, L, A registers contain. This isn't surprising since PostCur is just another machine code routine and will need "variables" itself.

It's inconvenient for us, though. It would be nice if L still had the same value so we could just INC it to get the cursor on the next row when we call PostCur.

Nor can we guarantee that H still contains the correct column or that the second call to CharOut will print an asterisk - the value in A will have changed.

When a machine code routine changes registers, we say it corrupts the registers. When a routine leaves the registers intact we say it preserves the registers.

If you find your machine code programs going wrong for no

you need, Table V being a resume of the codes we met last month.

A few hints on hand assembling might be useful here:

. Write out your code on paper before entering it into Hexer.

. Make sure you're dealing with hex numbers- any decimal numbers will

have to be translated.

. Check that the code ends with the opcode for RET, &C9.

. Be careful with your writing - it's all too easy to confuse 8 with B, 2 with 7, and E with F.

. Make certain that you write addresses in lo byte, hi byte form. That is,

&2FF8 becomes F8 followed by 2F in a machine code program.

. Double check the code before you run it!

Now for the worked example.

mnemonic	opcode
CALL pq	CD q p
RET pq	C9
LD A,(pq)	3A q p
LD (pq),A	32 q p

Table V: Additional opcodes

When a machine code routine changes registers, we say it corrupts the registers. When a routine leaves them intact we say it preserves the registers.

Suppose you had to hand assemble:

LD A,8
SUB A
LD (&2FF8),A
RET

Firstly you'd decide where you were putting it in memory. Since we're using Hexer, we tend to put everything at &3000.

Look at the first instruction:

LD A,8

we can see that it's of the LD A,n type. As Table I shows us, the opcode for this is 3E, followed by the number we want loading in. In this case it's 8, and there are no translation problems since it's the same in hex and decimal. So the first line translates as:

3E 08

The next line for us to code is:

SUB A

This is a rather interesting piece of code, since it subtracts the A register from the A register! I think you can guess the result...

From Table IV, you can see that the code for this is 97, so we can add this to our string of bytes in memory to give:

3E 08 97

The next instruction,

LD (&2FF8),A

is the poke type instruction we saw last month. Its opcode is 32. The trick is to make sure we follow it with the address we want to poke into split into lo byte, hi byte order. Given this, our code becomes:

3E 08 97 32 F8 2F

All that remains now is to translate the last instruction,

RET

The opcode for RET is C9 - you should really know this by heart now. Our code becomes:

3E 08 97 32 F8 2F C9

which, once we've done all our checks, we can enter via Hexer.

I tend to use a slight variation on this technique, writing the code for each instruction on a separate line. I also label each line with the memory location that the first byte (the opcode) of that line is stored in. Doing things this way makes it easier to compare the

code with the original mnemonics.

Also, knowing the starting address of each instruction will be handy later on when we do more with jumps and so on.

Doing things this way, the above code would be:

3000 3E 08
3002 97
3003 32 F8 2F
3006 C9

That's all for this month. You may have noticed that, apart from specifying addresses, the numbers we've been dealing with have all been one byte long.

Next month we'll have a look at pairing up the registers to allow us to handle two byte numbers.

A final tip: You might find it useful to copy the tables of opcodes onto card for easy reference - that's what I do.

Problem I

LD B,&01
LD A,&01
ADD A,B
LD (&2FF8),A
RET

Problem II

LD C,&1
LD A,&F
SUB C
LD (&2FF8),A
RET

Problem III

LD D,&FF
LD A,&2
ADD A,D
LD (&2FF8),A
RET

MY daughter has a new friend called Lionel. He lives in a nice little house with his dog, and she visits him as often as she can.

Come to that, most of the family have dropped in on him from time to time. In fact, they're spending so much time with him that nothing's being done in our house.

Lionel is my daughter's Little Computer Person, and he lives inside our Amstrad CPC464. Most computers have LCPs living inside them, but they are far too shy to come outside when anyone's around.

But who hid your shoes when you knew exactly where you put them? Who left the fridge door open last night? Chances are it was the work of

All he needs to be happy is his own little house, and that's what those lovely people at Activision have given him. Not a pokey little two-up-two-down terrace, but a magnificent three-storey semi with all mod cons.

As a child I was never particularly fond of playing with dolls and doll's houses, and the now-famous Action Man arrived too late on the scene for me to make a bosom buddy out of one.

I accepted that this new idea in home entertainment might have something special to offer, so with genuine interest I watched my daughter at her first encounter with her LCP, and before long I was really

Once she had entered her name and the date and time, there was a short pause before the house was displayed on a Mode 0 screen in a multitude of colours, with the front wall removed.

Sarah waited patiently staring at the screen in bewilderment wondering what was going to happen next. She couldn't tell her – I didn't know.

After what seemed like ages, but was in retrospect only a couple of minutes, the front door suddenly opened and in walked Lionel.

After a cautious look round, with contented smile on his face he started to amble around the premises gliding smoothly up and down stairs, examining every room and all the items of furniture.

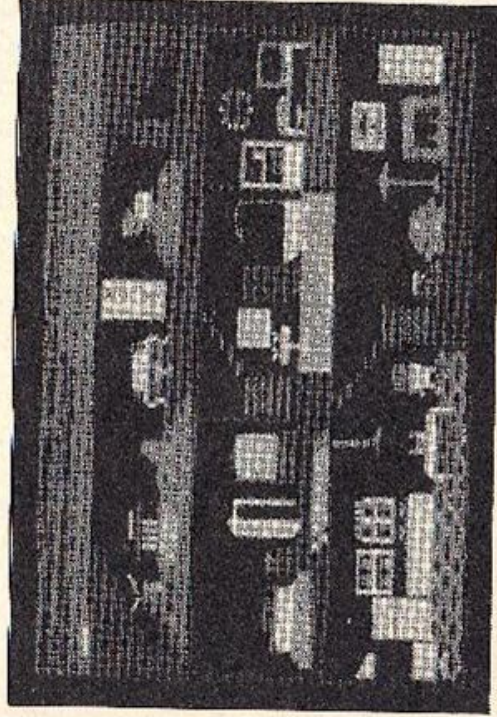
Attention to detail showed here in the change of footstep noises as Lionel moved from tile floor to carpet or wooden staircase.

I watched with interest as he rummaged through the filing cabinet and dressing table drawers. He looked in the fridge and inspected cupboard — just like anyone would who was looking for a potential home.

Then, without warning, he was gone — out through the front door. At this stage in the game there was no facility for input although we tried tapping the keys. All we could do was watch and wait.

After a short time he was back followed by his pet dog. He knew

*'Dear friend . . .
In his first floor study
the LCP types a letter*



where he was going now and headed straight for the computer room where he played a game on his micro.

The dog knew where it was going as well, making a beeline for its food bowl in the kitchen.

Lionel in the meantime had abandoned his computer, switched on the TV and settled in front of it in a comfortable armchair.

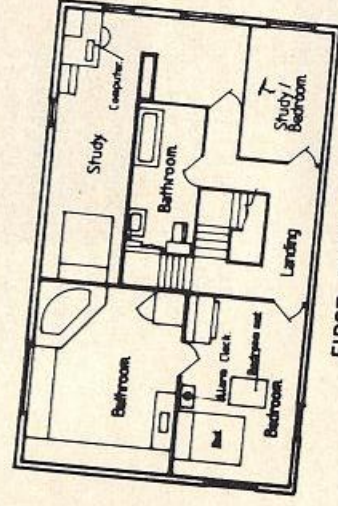
Then he sprang up and trotted to his bedroom, where he engaged in some exercises before going back up to the study, switching off the TV and putting a record on his stereo system.

To the sound of the music he did his LCP version of body-popping, jiggling about and clapping his hands like some demented fool.

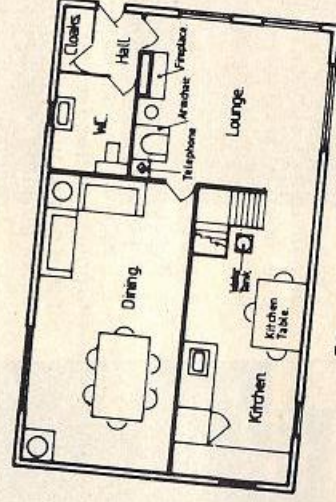
We could have watched him doing his own thing for hours, but the instructions gave hints on how to interact with our new-found friend, so we decided to try out a few. You can see the commands available in Table I.

At the press of a key we delivered food to the front door for both himself and his dog. On hearing the door bell he stopped what he was doing and promptly collected each item and deposited it in the fridge for future use.

We also sent him a record, which he put on the turntable but preferred not to play at that moment; and a book that he tucked away in the cupboard in

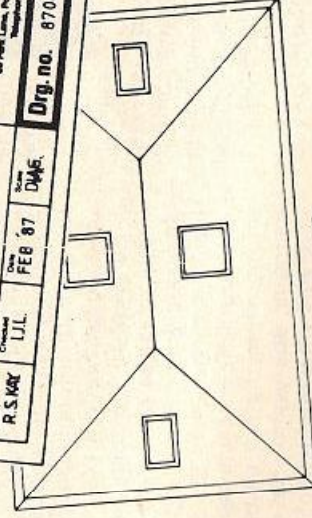


FIRST FLOOR

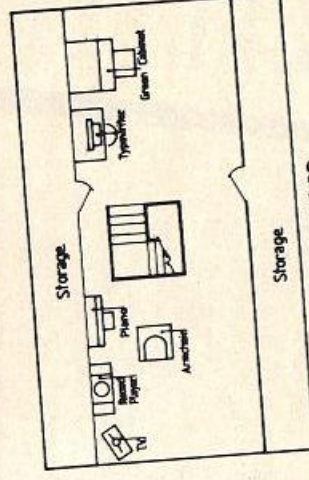


GROUND FLOOR

Lydon Reece Partnership Chartered Civil & Structural Engineers 30 Park Lane, Piccadilly, London W1K 1ND Telephone: 01-224 8772/3			
Drg. no. 8706 LCP 1			
Client	Contract	Date	Rev
R.S.KAY	L/L	FEB 87	1
HOUSE FOR LITTLE COMPUTER PEOPLE GROUND FLOOR AND FIRST FLOOR			



ROOF



SECOND FLOOR

Lydon Reece Partnership Chartered Civil & Structural Engineers 30 Park Lane, Piccadilly, London W1K 1ND Telephone: 01-224 8772/3			
Drg. no. 8706 LCP 2			
Client	Contract	Date	Rev
R.S.KAY	L/L	FEB 87	1
HOUSE FOR LITTLE COMPUTER PEOPLE SECOND FLOOR AND ROOF PLANS			

his study – also for future entertainment. When we made his phone ring he adjourned to his armchair in the lounge to answer it in his own LCP untranslatable language.

Stroking his hair seemed to make him very happy indeed, as he relaxed at full length in his favourite armchair.

We could always tell how Lionel was feeling by the look on his face. It ranged from a broad smile when he was happy, through a frown when he needed cheering up, to a grimace when he was short of nourishment.

There were no hard and fast rules on how best to cheer him up – guidelines and suggestions, yes, but the rest was all trial and error.

A minor emergency occurred when my daughter Sarah went to bed and I watched TV for a couple of hours leaving Lionel to his own devices.

When I returned he was tucked up in bed with a green face sporting a painful expression and looking decidedly ill.

The cassette inlay suggested this was due to lack of food or water, and after several attempts I managed to get him, sad-faced, out of bed to take a food delivery.

He was now a completely different Lionel, moving slowly across the floor looking down in the dumps – but at least he had food.

Delighted that I had apparently saved his life, I was horrified when he dumped all that lovely scoff into the fridge and shot back to bed.

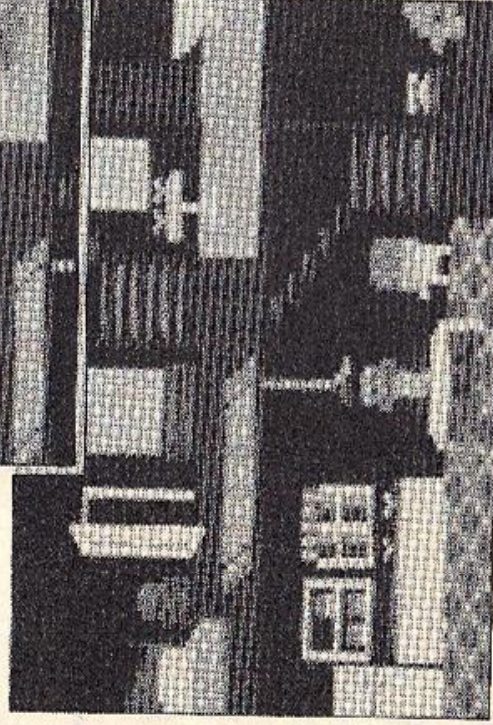
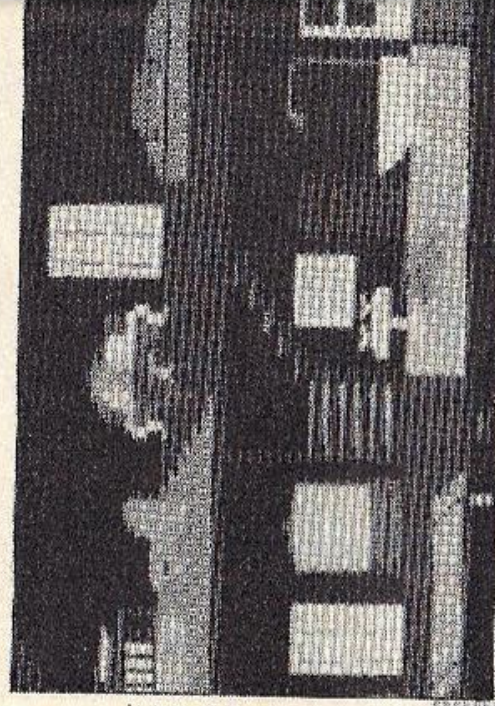
I had to beg him to eat and drink before he recovered – and that not before he had given me several grumpy mouthfuls from the comfort of his armchair. It took me some time to realise that the solution to getting him back to normal was to stroke his hair.

Our little friend did all the things you would expect of a new lodger. When

F Food is delivered to front door.
W Fills water tank in kitchen.
A Rings alarm clock in bedroom.
D Leaves dog food at front door.
C Makes a phone call.
P Strokes LCP's hair.
R Leaves a record at front door.
B Leaves a book at front door.

Table 1: Control keys

Making music:
Piano-playing
is one of
Lionel's
favourite
diversions



Time for tea:
Even LCPs
have to eat

he was tired he would change into his pyjamas and go to bed.

And naturally he had to be woken up (with the in-built alarm clock) in order to get him out.

He went to the toilet, washed his hands, cleaned his teeth, then went downstairs and made his breakfast, not forgetting to feed the dog.

He spent his time wandering around the house trying out all his gadgets, including the piano. He didn't play very well – his rendition of Mozart's C Major Piano Sonata was appalling – but we managed to get him off it by delivering a new record to add to his collection.

Any of the things he did under his own steam we could get him to do on request by typing in at the keyboard – it was by asking him to type us a letter that we managed to find out his name.

At intervals he would go to a box in his study, try to attract our attention by tapping on the screen, and if he succeeded would offer a choice between a simple card game, anagrams, or Poker.

If we pretended to ignore him he impatiently tapped his feet and checked his watch.

I tried my luck at a game of poker, which contained all the usual options to bet, draw and fold. After being in front for a short while I blew my win-

nings on two pairs against Lionel's three aces. Suddenly this was no friend of mine.

There's a lot more to this game than meets the eye – and much more than I've revealed. I can't fault its smooth colourful graphics, its user friendliness and charm.

It should carry a government health warning for first-time users. It might even have potential in an educational environment where it could provide ideas for study and discussion among young children.

In the early stages it was the centre of attraction in my lounge for long periods when other things should have been done, such as homework and housework.

My main concern now is whether, after the first week, when Lionel has probably done everything he is likely to do, the kids will still be going back for more.

The answer to that can only be found in the mind of a child. And if you can read that you are fortunate indeed. ■

Little Computer People

Tap: \$25

Disk: \$37.50

XEVIOUS

TAPE ONLY

\$22.50

XEVIOUS, one of the best Atari arcade games of the last five years, has now arrived on the Amstrad, courtesy of US Gold. And just how well does it stand up?

Well, it should be pointed out that this conversion does not strictly conform to the original screen layout of the arcade version.

Only half the screen area

is actually used for the action. That apart, the gameplay is pretty similar.

Piloting your Solvalu super spacecraft, you cruise over the vertically scrolling landscape, bombing Xevious ground installations and zapping air targets, while dodging flying mirrors. Your ultimate target is the Andor Genesis mothership.

Graphically the game has a very clean style to it, with good use of colour in both landscapes and sprites.

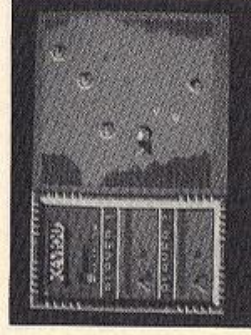
But I feel it lacks the

Thin on thrills

excitement of the arcade original, due to the slowness of the action – so it's not as addictive as you might expect.

The problem seems to be the program's inability to have too many fast-moving objects on the screen at once.

One solution would have been to restrict the action to a black background, but then it would not have matched the graphic style of the superb arcade game on which it is based.



Overall then, Xevious was a bit of a disappointment.

Victor Laszlo

Sound.....	7
Graphics.....	8
Playability.....	5
Value for money.....	5
Overall.....	6

ZOMBI

DISK ONLY

\$37.50

IT is rare that I receive a game which has me jumping up and down with excitement. At last it's arrived so put down your magazine now and go out and buy a copy of Zombi.

A terrible plague is sweeping the land and the dead victims are rising up as zombies.

Four people escape to the countryside in a stolen helicopter, but lack of fuel forces them to land on top of a hypermarket.

Using the four characters your task is to lock the three main doors of the hypermarket, kill all the zombies still inside and find fuel for the helicopter.

As if this were not difficult enough the helicopter on the roof attracts the attention of a roving gang of hell's angels.

Anyone lucky enough to have seen the classic film Zombi – Dawn of the Dead will be at a distinct advantage as the game follows the plot very closely.

The view, as seen by the currently selected character, is displayed in a large graphic window.

At the bottom of the screen are the game control icons. Selectable using joy-

stick or keyboard there are 18 in all with nine displayed at any one time.

Selecting the movement icon you send the first member of the team cautiously down into the building.

As each new scene scrolls into view you hold your breath waiting to catch sight of your first zombie. As you inch your way along the corridor on level three the zombie icon flashes.

Now you've eight seconds grace before the beast attacks and in this time you must decide whether to run, shoot it (if you've got a gun), bash it with a heavy object, or pulp it with your bare hands – yeuch!

Bashing and pulping rely on a good joystick waggling technique. The rate of waggle is displayed on a pulp-o-meter, and as you overpower your rotting adversary his icon fades away.

If you are carrying a gun you must quickly select the use icon. A line will move repeatedly across the pulp-o-meter, and you press fire when the line is dead centre to ensure a clean kill.

It is not advisable to panic when holding a gun – if you fire when there's no zombie around the program assumes that you wish to commit suicide and

High on skills

promptly blows your head off.

All the views are drawn in high resolution black and white and objects that can be carried or manipulated are highlighted in blue and red.

The shops on either side of the shopping malls can be entered and searched for useful items – a good place to begin is the gun shop on level two.

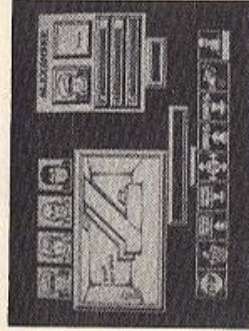
In the computer shop on the same level interrogating a micro reveals the words Electricity and Night.

The significance of these cryptic comments soon becomes apparent as the sun sets and you realise that the power is off.

The logic used is impeccable. For instance when shooting a second zombie immediately after killing the first, the rate at which the little bar in the pulp-o-meter travels is much faster due to the increased speed of your heartbeat.

If you are moving around in the dark and you know exactly where you left a particular object you can retrieve it by placing the Pick Up cursor in the same place on the blank screen – it will still be there.

Moving about near the exits on the ground floor is not to be recommended – there are too many corpses



wandering around.

On one occasion, feeling particularly brave, I ventured out into the car park – the zombies were coming thick and fast but I was holding my own.

Stepping a little too far away from the hypermarket resulted in my requiring an unexpected change of underpants as I came face to face with an army of decomposing deadheads – I soon joined them.

This is the best arcade adventure game I have ever seen. The scenario is complex, the atmosphere spine-tinglingly tense and the action explosive.

Jon Revis

Sound.....	8
Graphics.....	10
Playability.....	10
Value for money.....	10
Overall.....	10

EXPLORER

TAPE: \$25
DISK: \$37.50

WHILE zipping through the galaxy in your newly acquired, second-hand star cruiser the inertial stabilizers fall off.

Thirty seconds later you are spiralling down towards certain death on a desolate and nameless planet.

Fortunately you survive. Unfortunately your cruiser broke up on re-entry and its nine parts are now scattered over the planet's surface.

Once you've regained

your senses you switch on your object sonar. This will give you a compass bearing and, if you're very lucky, the distance to the object.

The planet's surface is covered with trees, water and termite mounds, all artistically displayed as detailed full screen drawings.

As you walk forwards through the jungle the section directly in front of you becomes greatly enlarged before being replaced with the next piece.

Exploration of the planet is made a little easier by the thoughtful provision of tel-

ports at regular intervals.

On entering one of these you are prompted for the name of your destination and according to the inlay card there are so many locations provided that the one you enter will exist.

The locations aren't chosen at random because typing the same name into different teleports will take you to the same destination.

Reassembling the ship - if you ever find any of it - is made easier by the provision of nine beacons and anit grav drones.

No matter how hard I tried to appreciate the complexity

HIVE

TAPE: \$25
DISK: \$37.50

DUE to a quirk of evolution bees have evolved far beyond mere humble stingers.

You have been chosen to pilot the Grasshopper through the twisting and branching passageways which make up the hive.

Stretching out in front of you is a series of concentric

circles, the inner ones often offset in one direction to create the illusion of curves.

Below this is the control panel, which provides information regarding energy levels, attack warning lights, what you are carrying, and the Save Game password.

While searching the tunnels you will be attacked by numerous flying objects which you must destroy before they ram you.

Shooting them is no easy task as the Grasshopper handles as though she were

flying through honey.

The alien threat is made worse by large numbers of high spikes and ground stingers which are attached to the tunnel walls.

The Grasshopper can carry up to 24 objects, including keys, special lasers, bombs and energy pylons, these last being essential if you are to last more than a few minutes.

As such a great deal of effort has been put into the quality of the graphics it is a pity that the game is spoilt

Pretty pictures



of Explorer I couldn't help feeling that it was just a lot of pretty pictures and very little else.

Steve Brook

Sound.....	7
Graphics.....	8
Playability.....	5
Value for money.....	5
Overall.....	5

Baleful bees

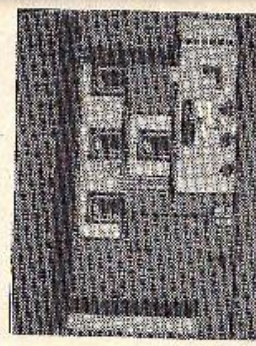


by its sluggish response to the controls.

Nev Astly

Sound.....	5
Graphics.....	8
Playability.....	8
Value for money.....	8
Overall.....	8

Feeble four



seen such crude sprites grace an Amstrad screen.

The backgrounds are a little better but nothing to write home about. The game's only redeeming factor is the number of secret locations that can be explored.

Another stick-man hero - Bruce Lee - stars in the second game. The graphics are roughly of Zorro standard, but it is slightly more fun to play.

In this one or two-player game you take the role of Bruce Lee in his struggle against the evil wizard.

In game number three the

stick men begin to put on a little weight as Bounty Bob Strikes Back!

As Bounty Bob you have returned to reclaim your mine from the mutant blobs.

You do this by stepping on every last section of an incredibly crude ladders and levels background.

The final game, Dam Busters, is the famous flight simulator which never really got off the ground.

The program lets you join 617 squadron on its epic mission to destroy the German dams.

You are expected to flit between the roles of pilot,

gunner, navigator, engineer and bomb aimer.

The Amstrad Academy is a collection of four third rate games - you have been warned.

Carol Barrow

Sound.....	5
Graphics.....	5
Playability.....	5
Value for money.....	5
Overall.....	5

AMSTRAD ACADEMY

TAPE ONLY
\$25

THIS is a compilation of four US Gold programs - three ladders and levels games and a simulation.

The first game follows the exploits of a minute stick man by the name of Zorro.

The evil Sergeant Garcia has captured a beautiful senorita and imprisoned her in the fort.

You, as Zorro, must run, jump, climb, and fence your way past the guards on your way to the fort.

It is a long time since I've

IMPOSSABALL

CPC TAPE ONLY
\$22.50

AT LONG last, an original game, featuring original gameplay and graphics.

You've heard that before? Well possibly, but Impossible is different and like the best games relies on simple gameplay to get you really hooked.

This is a race against the clock to bounce a ball between two mosaic patterned floors, to squash

cylinders protruding from the ceilings and the floors, while avoiding spikes, firebolts and plasma fields on the way.

The real beauty of the game is the way the ball can be shot away into the distance and pulled right up to the front of the screen, using the checker board pattern to accurately judge the distances involved.

From a technical viewpoint, the game features fast variable-speed split-level horizontal scrolling and true

Something new

perspective with precise curved-surface collision detection.

And there are eight different skill levels.

This program, the first for Hewson by John Phillips, introduces new ideas in a deceptively simple format.

And while it may not appeal to hardened arcade players, it will definitely appeal to those seeking a stimulating game.

The sound effects do not overly intrude into the quietly relaxing atmosphere

**THE SACRED
ARMOUR OF
ANTIRIAD**

CPC TAPE ONLY
\$22.50

IN 2086 Earth was plunged into a nuclear war as a result of the development of the awesome and indestructible Anti-Radiation Combat Suit.

As the centuries passed mankind built a new and peaceful civilisation, but this peace was shattered by the arrival of an alien invasion force.

Secretly the elders trained young men to become warriors and the greatest of these was Tal.

When the time was right Tal was sent on a mission to find the sacred armour of Antiriad.

The terrain varies between steamy jungle and the broken remains of former civilisations, yet the dangers are always the same.

All regions of the planet are under constant surveillance by alien machines and contact with any of these

will severely deplete your strength.

Tal's quest is made more difficult by the fact that four of the suit's components are missing. A further twist is the fact that you cannot pick up any of them until you have located and activated the suit.

Until the armour is complete Tal must rely upon more primitive weapons to combat the aliens.

An excellent combination of first-rate graphics and sound puts The Sacred

**500cc GRAND
PRIX**

CPC ONLY
TAPE: \$25.00
DISK: \$37.50

BEING a keen biker I was eager to try my hand at 500cc Grand Prix as soon as it arrived at the office. Would it live up to the real thing?

After loading up the game I wasn't disappointed. This is an exciting motorcycle race simulation with superb graphics, though the sound could be improved.

It's a one or two-player game. There are six motorcycles in all with the

micro controlling the remaining four or five bikes.

After setting the difficulty level and number of players you select either a full world championship or training at any of 12 circuits around the world.

Unfortunately, after making your selection you can't return to the initial menu which is very frustrating.

The screen display is split down the middle with two views of the track and riders.

One follows your bike and the other the leader - rather like two cameras just behind the bikes.

Below these are your

On your bike

speedometer, rev counter and gear indicator.

You can't run off the road on a bend as the track is fully enclosed by crash barriers, but if you rub against them for more than a few seconds your leathers will catch fire and you'll have to stop to put them out.

The starting grid is rather cramped and you need to take care when dashing for the first bend.

The computer-controlled riders weave and dive about jockeying for the best position and it's all too easy to collide with another bike and end up in a heap on the

tarmac.

I thoroughly enjoyed this grand prix and if you like race simulations you'll love this version. It's almost as good as the real thing.

Roland Waddilove

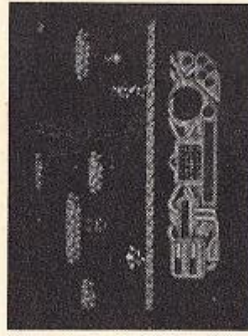
Sound..... 5
Graphics..... 9
Playability..... 9
Value for money..... 9
Overall..... 9



and I think I will be playing this unique game for many months to come.

Victor Laszlo

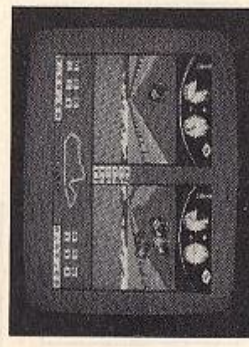
Sound..... 7
Graphics..... 8
Playability..... 8
Value for money..... 8
Overall..... 8



Armour of Antiriad right up there with the very best arcade adventure games.

James Riddell

Sound..... 9
Graphics..... 9
Playability..... 8
Value for money..... 9
Overall..... 9



Highest tech

STARGLIDER

CPC

TAPE: \$37.50

DISK: \$49.95

has succeeded with flying colours.

Starglider puts you at the controls of the AGAV (Airborne Ground Attack Vehicle) in a mission to defend your planet against the nasty Egrons.

James Follett's 64-page novella, included with the game, sets the scene for what is basically the highest-tech shoot-'em-up game yet produced for a home computer.

Comparisons will be made with games like Mercenary, but the animation and smooth action of

Starglider puts the game in a class of its own.

Your cockpit display provides all the updates you need on energy levels, force-shield status, laser weapon strength, altitude plus a myriad of other features.

Docking in the Alliance silos to replenish the shields, plasma power units and laser cells provides the tactical element of the game.

Overall, the speed of frame animation does not match that of the Atari ST version, but that reservation

HANDS up those of you who thought Starglider would never make it over to the Amstrad CPC!

The rave reviews given to the amazing Atari ST version, which features superb sampled sounds obviously set a hard challenge for those converting it to the 8 bit Amstrad computers.

But Realtime Software

apart, Starglider is an essential purchase for Amstrad gamers of all tastes and persuasions!

Victor Laszlo

Sound..... 8
Graphics..... 9
Playability..... 9
Value for money..... 8
Overall..... 9

THE GREAT ESCAPE

CPC

TAPE ONLY: \$22.50

ACHTUNG! achtung! der Britisher prisoner is escaping! Breaking out of a POW camp may look easy on TV but just try it for yourself - you'll soon find out that Colditz is no holiday camp.

The Great Escape is an arcade/adventure game in which you formulate an escape plan, gather together the equipment needed, then

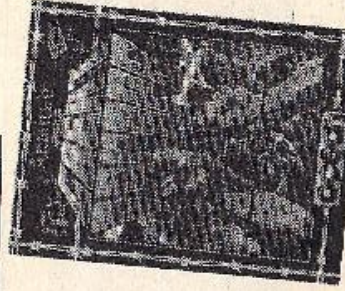
have to find a safe place to store them.

Capture results in the confiscation of anything you are carrying and a spell in solitary confinement.

The 3D graphics are drawn in black and white on a small scale using a great deal of detail.

These would be very good were it not for the ripple which occurs every time the computer is asked to redraw the full display. After about an hour this effect was driving me mad.

The Great Escape is a



good idea but the action is slow and the graphics irritating.
Nev Astly

Sound..... 4
Graphics..... 7
Playability..... 7
Value for money..... 7
Overall..... 7

GAUNTLET

CPC

TAPE: \$25

DISK: \$37.50

THE wait is over! Gauntlet, one of the first multi-player arcade games, has finally arrived on the Amstrad.

US Gold has produced the two-player variant and has stuck very closely to the arcade original, right down to glossy title and menu selection screens.

The game is best played on the two-player option where the level of interaction makes it one of the most exciting games on the

home market.

For those of you who've not seen it, Gauntlet is a multi-directional medieval scrolling maze game.

Survival's the object, in the face of monsters' attacks while you search for food, treasure and magic potions.

But clever use of tactics in the collection of special potions will give you extra powers, including extra armour, shot speed and fight power.

The choice of the hero you control will determine your own fighting, shooting and magic powers, giving each new game a character and

Medieval mazes

variety of its own.

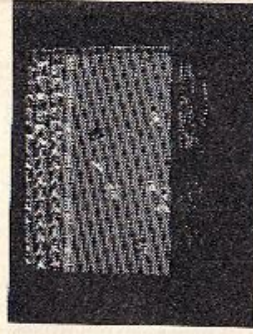
The graphics of the arcade version have been faithfully reproduced.

Plenty of attention has been given to the fine detail of the animated characters, treasures and amulets in particular.

The nice bit about Gauntlet is the fact that the villains all have their own modes of attack, and some need three direct hits before they die.

Therefore you must attack each one in a different way if you wish to keep your health intact.

Not only does Gauntlet already have masses and



masses of different levels of play, but US Gold plans to release an expansion cassette and disc containing hundreds of new levels and treasure rooms.

This game will run and run!
Victor Laszlo

Sound..... 7
Graphics..... 9
Playability..... 8
Value for money..... 9
Overall..... 9

To boldly go...

TAU CETI

PCW 8256/8512

DISK ONLY:

\$49.95

GAL-CORP needed to make the planet safe for recolonisation – and guess who volunteered for the task?

So there I was, about to launch into the unknown, when I got a last message from Gal-Corp Central.

It was all there – the low-down on procedures for navigation, docking with reactors and supply centres, defence estimates and silhouettes of known robot defence installations – now they tell me!

But even that didn't prepare me for the sheer enormity of the task ahead.

The Skimmer had been fully fuelled and armed. Checking the 31 other cities on the map I decided to make a Grand Tour and

launched into the bright Tau Ceti sunlight.

I flew in low towards the centre of Reema, the first city, slowing down to conserve fuel – I didn't know where the next supply station was going to turn up.

Then the warning lights flashed on. A quick check on the radar confirmed my fears, I was under attack.

They'd caught me by surprise and my shields were wilting under their laser fire.

I dodged as best I could then turned on them and with my attack computer tracking the renegade robots, reduced them to molten scrap.

And so it went on as I battled my way to the jump pad, my passport to the next city.

I made it, relieved, just as the sun was going down. My relief was shortlived –

for as I teleported through, the robots attacked with even greater ferocity.

The computer warned of enemy missiles and I immediately launched the AMMs to counteract.

Tracking a fast-moving Hunter Droid I fired two heat-seeking missiles.

It destroyed the first but, through the cockpit window, I just saw the second connect as I rounded on an enemy control tower, taking it out with my laser cannon.

It was now difficult to see in the darkness. Luckily I had a supply of Starlight flares with me, but were eight going to be enough?

Things were getting pretty hot as I made out the bulk of a reactor substation through the infra-red night sights.

As I approached, the airlock opened and I was safe. Here was the first of the reactor rods I needed,

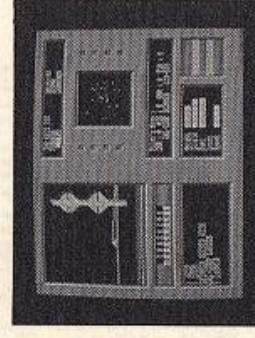
but I still had to find the rest, assemble them and replace them in the main reactor.

But my most immediate concern was to refuel and re-arm, there just had to be a supply depot out there somewhere.

It was going to be a long and dangerous mission, but I was in no hurry...

Niels Reynolds

Sound.....	3
Graphics.....	8
Playability.....	9
Value for money.....	9
Overall.....	9



BOUNDER

PCW 8256/8512

DISK ONLY:

\$37.50

IF you used to enjoy dodging the cracks in paving stones, you're going to love Bounder. Well, perhaps the analogy is not quite right, but it was an association I couldn't help making.

Imagine a wide, hexagonally-paved path, seen directly from above.

On this path you control a bouncing tennis ball, the Bounder, which becomes larger and smaller as it bounces towards and away from you.

Bounder must be moved around the screen, hopefully surviving until the bonus level at the end of each stage is reached – most definitely easier said than done!

You'll have to keep to the hexagonal slabs – landing

on anything else usually means a life lost.

Collisions with walls, mountains and various aliens also cost lives. However, you also meet useful objects in your travels, although these are not as numerous as the nasties.

Other features include slabs with question marks and arrows.

The former are usually bad news, deducting lives or a number of jumps from those available in the bonus stage, but they can sometimes be helpful too.

The arrowed slabs allow you to make extra long jumps – handy at times, if not essential.

The pathway scrolls downwards at a fixed speed and consequently you don't get much time to ponder particular hazards.

In this game you learn by your mistakes and hope to remember them next time

around.

Movement is controlled by four keys, which are unfortunately not redefinable, although Cascade's joystick interface is supported.

Most points are scored in the bonus stages, which involve bouncing Bounder on to question marked slabs within a set number of jumps.

Points are scored for each slab and for any unused jumps from your initial allocation.

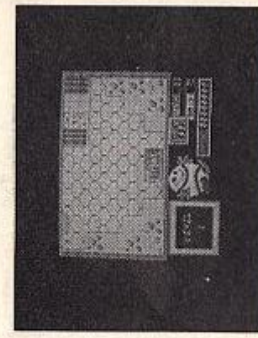
One niggle was the lack of a high score facility.

Having seen the Spectrum version of Bounder, I was impressed with the graphics on the little machine.

However, I must admit some disappointment that Gremlin hasn't improved on this for the PCW, although the standard is nonetheless good enough.

Bounder is novel in its

Jump starts



approach and presentation although it perhaps lacks interest in depth.

Still, I did enjoy the game in its own right and after a shaky start began to find it very playable and increasingly addictive.

For those who like a single minded, infuriating game requiring great dexterity and coordination, Bounder should prove great fun.

Niels Reynolds

Sound.....	3
Graphics.....	8
Playability.....	9
Value for money.....	8
Overall.....	8

THE PAWN

PCW/CPC 6128

DISK ONLY:

\$49.95

I HAVE to admit that, after all the attention this graphic adventure has received, I couldn't wait to load it and find out if it was as good as they said. Believe me it is.

The game is set in the land of Korovnia at a time of great social unrest.

For many years there has been dissatisfaction with the rule of King Erik. The Roobikyoub dwarves, under the leadership of Gringo Baconburger, are agitating for the abolition of the monarchy.

The situation reached a

head when Queen Jendah II was assassinated. In the heat of the moment the dwarves were blamed and banished from the land.

They maintain that the likely perpetrator of the evil deed is the court magician Kronos.

After all, his sister Zita is known to have had an unwholesome influence over the recently vanished Princess Lacey.

The populace also feels that some sort of investigation should be made and the longer the king ignores the situation, the more unpopular he becomes.

It is into this situation that you arrive. You awake on a path after being rendered

unconscious by an unknown assailant.

Your main task is to get out of the country before it explodes into violence, but don't expect to do so without getting involved with the inhabitants.

You type in your commands from the keyboard, but unlike other adventures, the parser seems to handle virtually any input you give it, and the degree of interaction with the other characters is very impressive.

I still have a fair way to go in the adventure, but the problems I have solved have been logical, if only in retrospect.

The Pawn comes with a



44-page novella which serves as the background to the section. A small book gives advice on tackling adventures in general and the Pawn in particular.

Overall, the atmosphere is beautifully evocative and the gameplay excellent. **Gand**

Presentation	10
Atmosphere	10
Frustration Factor	4
Value for Money	10
Overall	10

STARGLIDER

PC 1512

DISK ONLY:

\$49.95

ASK anyone what their opinion is of the PC1512 and they'll tell you that it's a good, solid business machine with literally hundreds of excellent software packages available, from word processors to computer-aided design and communications.

However, Rainbird has set that idea on its head with this PC version of the top-selling game Starglider.

It features some of the fastest, smoothest graphics around and is pure arcade action at its best.

The battle is set on Novenia, a distant planet in a far-off galaxy that has been invaded by the evil Ergon empire.

You take the part of a pilot in control of a modestly armed AGAV - Airborne Ground Attack Vehicle - and your task is to annihilate the enemy forces.

The main playing screen shows your view from the cockpit of the AGAV with your instruments occupying

the lower quarter of the display.

These include radar scanner, energy level, shields, laser status, altitude, velocity and so on.

Through the windshield you see the planet surface stretching away beneath you, towers reaching up into the sky, and enemy craft, both land and air based.

These are all drawn using state-of-the-art vector graphics, with full hidden-line removal.

The speed and animation has to be seen to be believed and only the Atari ST version (the micro that Starglider was originally developed on) is better.

Many of the craft are animated and include the Ergon's flagship, the dreaded Starglider One, complete with flapping wings.

At first sight Starglider appears to be a shoot-'em-up and your first instinct will be to stick to the tried and trusted rule of "if it moves blast it, if it doesn't, blast it anyway".

However, this rather simplistic view will not get

Tops for PC

you very far.

The game does involve a fair degree of arcade action, but in addition the flying and combat sequences make it not unlike a flight simulator in some respects.

And you'll need to develop your own strategy if you are to survive.

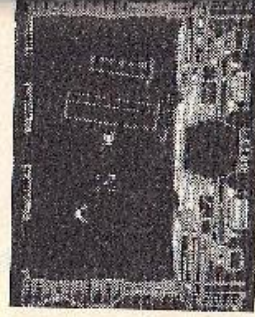
There is also plenty to discover and investigate. Your energy banks, weaponry and shields are soon depleted and need replenishing. The documentation does tell you how, but in practice it isn't so easy.

The AGAV can be topped up with energy by performing a particularly dangerous manoeuvre involving flying at ground level along a power line between three towers.

It's quite difficult when you're being blasted by enemy tanks, stargliders and missiles, and stompers are trying to walk all over you...

It is also possible to dock with special silos where your ship will be patched up and re-armed.

You can easily recognise one of these - they are just



about the only things that don't fire at you.

When docked at one of these silos you may be given one of eight special missions and you can get a run down on the strength of the Ergon forces.

The package includes a poster, novella, playing guide and key chart. The novella is essential reading as it provides background information, playing tips and clues.

I can't praise this product highly enough. It is quite simply the best game on the PC you are likely to see. If you've got a PC, get out and buy Starglider and prepare to be amazed.

Roland Waddilove

Sound	10
Graphics	10
Playability	10
Value for money	10
Overall	10

One step further

ROLAND WADDILOVE'S Variable Dump takes the slog out of error checking

VARIABLE Dump is a short machine code routine to add an extra command 'VARDUMP' as an RSX. It simply lists all the variables that have been used by a Basic program.

When you want to list all the variables, you type in a listing and it doesn't give you a listing when run. It's most likely to be a simple typing slip.

One of the features of Locomotive Basic is that it will create an additional variable every time it creates a new word. It doesn't recognise 'S' if it's already a variable called 'score' and you accidentally typed 'score' instead it would add 'score' to its list of variables. By printing all the variables it's easy to spot any that have been entered incorrectly.

The program must be run before using 'VARDUMP'. After each variable name it waits for a key to be pressed, so you can see the list of

variable stack is the string descriptor giving the name, length and address of the start of arrays. Of course if there

Check your variables to spot your mistakes

From the February issue of Computing with the Amstrad

VARIABLE Find is a machine code program to complement Roland Waddilove's Variable Dump published in the February 1987 issue of *Computing with the Amstrad*. Having identified mistyped variable names with 'VARDUMP' you can now pinpoint the offending line number using 'VFIND', the RSX added by this program.

Type in Program 1 which is the Basic loader for the machine code. Save it before trying to run it because once you have debugged the data statements the Basic will erase itself from memory when it has installed the RSX.

The new command can be used with or without Vardump in memory and will accept upper or lower case as valid input. To use it put the variable name into a string, such as `A$="FRED"`. It doesn't have to be `A$`, you could use `F$` or just about anything else that takes your fancy. You should omit suffixes such as `$` and `%` from the name otherwise they will be treated as part of the name and produce a nil return.

Then type `'VFIND,@A$` and a list of line numbers will be printed out. Don't forget the `@` before `A$` as this is vital to the way your Amstrad extracts the variable name from the string.

The search routine has been kept fairly simple for the sake of speed, economy of memory and your fingers. Consequences of this are that the routine does not distinguish between string or numeric variable names and may print out spurious line numbers for one of two reasons.

Firstly, if your variable name also happens to be embedded at the end of a longer name then the line number

containing the longer name will be listed. An example of this would be printing a line number containing `OLDFRED$` when what you were really after was `FRED`. Of course line numbers containing `FRED` would also be listed out, so this is not too much of a problem.

Secondly, if your variable name is very short, particularly only one or two characters in length, it is quite likely that other bytes of program data will match the Ascii values of the name. These line numbers will be listed out for no apparent reason.

In practice you are unlikely to be searching for one or two character long variable names, and even if you are 'VFIND' will produce a shortlist of line numbers among that are correct lines which are easily checked.

Program 11 is a listing of the assembly language source code. Budding machine coders among you should find it helpful to look at the way it works, particularly the method for passing the contents of a string variable over to a machine code program, the search method and the routine for printing out a decimal.

You may know that if you make a call to a machine code routine either by RSX or direct CALL and if a parameter follows your call—such as `CALL &A000,3` or `CALL &A000,z` where `z` is a numeric variable—the IX register points to the low byte of the last parameter passed and the A register contains the number of parameters.

This is easy enough to deal with if a

number or numeric variable is being used but what happens if you want to pass a string? Well unfortunately you can't CALL `&A000,FRED` and hope that `FRED` will be available to your routine. You must put `FRED` into a string such as `A$` and then CALL `'VFIND,@A$`.

On arrival at the machine code the A register still contains the number of parameters passed, but the IX register points to the address of the storage location of the variable `A$`. The `@` before `A$` tells your micro to pass the address of a variable rather than its value.

This address stores the location of the string descriptor block, which in turn points to the address of the contents of the string as well as giving the number of characters it holds. After sorting all that out we're finally in a position to access the string from our machine code program.

Having found the start of the string it is transferred to a store while being converted to upper case if necessary. 128 is added to the value of the last character. The stored variable name now matches the form in which it appears in the Basic program, assuming upper case. As you may have originally used lower case or may have mixed upper and lower case the search routine converts all lower case to upper case before comparison.

The search begins at address `&170` where all Basic programs start. To help you understand how this

Pinpoint mistyped lines with
IAN SHARPE'S Variable Find

works you may like to refresh your memory about the structure of a Basic line with the articles *How Basic Works* in the November and December 1985 issues of *Computing with the Amstrad*.

The logical structure of the program runs like this:

Use the first two bytes of the line to generate the address of the next line. If the bytes are both zero then we're past the end of the program so return, else store new address. Get and store the line number, find the address of the first program byte in

the line and subtract the length of the variable name from the remaining line length.

If the answer is less than or equal to zero then the line is too short so jump to the next line, else make this value a loop counter for the next step.

Move along the line comparing the first byte of the variable name with the byte of program data converted to upper case if required.

If at the end of the line move to the next line. If a match is found then compare any remaining bytes of the variable name with subsequent bytes

of program data. If at any stage mismatch occurs then carry on the main search where it was left of otherwise if a total match was found convert the line number to decimal, print it and continue searching on the next line.

The routine to convert a 16 bit number to decimal and print it out is fairly simple and is just the job for printing out, for example, the score in your latest arcade masterpiece. Vardump and Vfind together should save you hours in debugging your Basic programs. Happy hunting.

Program I

```
10 REM VFINF
20 REM By Ian Sharpe
30 REM
40 REM (c) Computing with the Amstrad
50 REM
60 REM
70 MEMORY &9E5F
80 FOR addr=&9E60 TO &9FF6
90 READ byte$:byteval=VAL("&"+byte$)
100 POKE addr,byteval
110 check=check+byteval
120 NEXT
130 IF check<>4755 THEN PRINT"ERROR IN DATA":END
140 CALL &9F83
150 PRINT" A$=";CHR$(34); "variablename";CHR$(34)
160 PRINT" IVFIND,BA$"
170 PRINT" Delit $ and X suffixes"
180 NEW
190 DATA FE,01,00,00,6E,00,0D,66
200 DATA 01,E5,00,E1,00,7E,00,FE
210 DATA 00,C8,FE,29,00,32,A7,9F
```

```
220 DATA 47,0D,6E,01,0D,66,02,0D
230 DATA 21,AC,9F,7E,FE,61,0A,8B
240 DATA 9E,C8,AF,0D,77,00,0D,23
250 DATA 23,10,F0,0D,C8,FF,FE,21
260 DATA 70,01,22,A8,9F,3E,42,C0
270 DATA 1E,8B,C0,0D,2A,AB,9F,0D
280 DATA 5E,02,0D,56,03,ED,53,AA
290 DATA 9F,0D,5E,00,0D,56,01,7B
300 DATA B2,C8,03,0D,E3,0D,19,0D
310 DATA 22,A0,9F,0D,E1,E1,11,04
320 DATA 00,0D,19,AF,ED,52,3A,A7
330 DATA 9F,5F,16,00,AF,ED,52,0A
340 DATA 9D,9E,CA,9D,9E,45,0E,00
350 DATA FD,21,AC,9F,0D,7E,00,FE
360 DATA 61,0A,EE,9E,C8,AF,FD,9E
370 DATA 00,CA,F8,9E,0D,23,10,EC
380 DATA C3,9D,9E,0D,E5,FD,E5,C5
390 DATA 3A,A7,9F,47,0D,7E,00,FE
400 DATA 61,0A,1D,9F,C8,7F,C2,16
410 DATA 9F,C8,AF,C3,1D,9F,FE,07
420 DATA DA,1D,9F,C8,AF,FD,8E,00
430 DATA C2,29,9F,0D,23,FD,23,10
440 DATA DB,3E,00,8B,CA,37,9F,C1
450 DATA FD,E1,0D,E1,C3,F4,9E,2A
```

```
460 DATA AA,9F,0D,21,97,9F,FD,21
470 DATA A2,9F,0E,05,0D,5E,00,0D
480 DATA 56,01,06,00,AF,ED,52,0A
490 DATA 56,9F,04,C3,AC,9F,19,FD
500 DATA 70,00,FD,23,0D,23,0D,23
510 DATA 0D,CA,67,9F,C3,44,9F,86
520 DATA 04,0D,21,62,9F,0D,7E,00
530 DATA FE,00,C2,7D,9F,0D,36,00
540 DATA F0,0D,23,10,F0,06,06,0D
550 DATA 21,A1,9F,0D,7E,00,C6,30
560 DATA C0,5A,8B,0D,23,10,F4,C1
570 DATA FD,E1,0D,E1,C3,9D,9E,10
580 DATA 27,E0,03,64,00,0A,00,01
590 DATA 00,F0,00,00,00,00,00,00
600 DATA 00,00,00,00,00,00,00,00
610 DATA 00,00,00,00,00,00,00,00
620 DATA 00,00,00,00,00,00,00,00
630 DATA 00,00,00,00,00,00,00,00
640 DATA 00,00,00,00,00,00,00,00
650 DATA 00,00,00,00,21,F6,9F,3E
660 DATA 01,8E,C8,36,01,01,E7,9F
670 DATA 21,F2,9F,0D,01,8C,C9,EC
680 DATA 9F,C3,60,9E,56,46,49,4E
690 DATA C4,00,00,00,00,00,00,00
```

Program II: Variable Find source code

```
ORG &9E60      ;coexist with vardump
;**** Error Checking ****
.vfind
CP 1           ;1 parameter passed?
RET NZ
LD L,(IX+0)
LD H,(IX+1)
PUSH HL
```

```
POP IX
LD A,(IX) ;get string length
CP 0      ;empty string?
RET Z
CP 41     ;too long?
RET NC
```

```
LD A,(HL) ;get character
CP 97     ;lower case?
JP C,tflip
```

```
RES 3,A      ;make upper case
```

```
;**** Set String Contents ****
```

```
LD(namelen),A
LD B,A
LD L,(IX+1) ;point HL to start
LD H,(IX+2) ;of string
LD IX,namelen
```

```
LD(IX),A      ;and store
INC IX
INC HL
```



```

DJNZ tfrlp
SET 7,(IX-1) ;add 128 to last
;**** Search *****
LD HL,&170 ;start of basic
LD (linadr),HL
.searchloop
LD A,66
CALL &BB1E
RET NZ ;escape?
LD IX,(linadr)
LD E,(IX+2)
LD D,(IX+3)
LD (lineno),DE
LD E,(IX+0) ;get line length
LD D,(IX+1)
LD A,E
OR D ;if zero then finished
RET Z
.schlp1
PUSH DE
PUSH IX
ADD IX,DE ;find next line addr
LD (linadr),IX
POP IX
POP HL
LD DE,4
ADD IX,DE ;find start prog data
XOR A
SBC HL,DE ;no. bytes to search
LD A,(namelen)
LD E,A:LD D,0
XOR A
SBC HL,DE ;shorter than name?
JP C,searchloop
JP Z,searchloop
LD B,L
LD C,0
LD IV,namestore
.lineloop ;search line
LD A,(IX)
8 CP 97 ;lower case?
JP C,lloop1
RES 5,A ;make upper case
.lloop1
CP (IV+0) ;compare first char
JP Z,checkrest
.nextchar
INC IX
DJNZ lineloop
JP searchloop
.checkrest ;check following chars
PUSH IX
PUSH IV
PUSH BC
LD A,(namelen) ;get name length
LD B,A ;make it a counter
.chrstlp
LD A,(IX)
CP 97 ;lower case?
JP C,crulp2
BIT 7,A ;last char?
JP NZ,crulp1
RES 5,A
JP crulp2
.crulp1
CP 215 ;last char is +128
JP C,crulp2
RES 5,A
.crulp2
CP (IV)
JP NZ,crexit ;abort if mismatch
INC IX
INC IV
DJNZ chrstlp
.crexit
LD A,0
CP B ;if counter=0 then
JP Z,matchfound
POP BC
POP IV
POP IX
JP nextchar
LD A,0
CP B ;match found
JP Z,matchfound
POP BC
POP IV
POP IX
JP nextchar
;**** Print Line No. *****
.matchfound
LD HL,(lineno)
LD IX,decdigits
LD IV,decdigits
LD C,5
.fdnloop1
LD E,(IX+0)
LD D,(IX+1)
LD B,0
.fdnloop2
XOR A
SBC HL,DE
JP C,fdn1
INC B
JP fdnloop2
.fdn1
ADD HL,DE
LD (IV),B
INC IV
INC IX
INC IX
DEC C
JP Z,print
JP fdnloop1
.print
LD B,4
LD IX,decdigits
.sizlp ;strip leading zeros
LD A,(IX)
CP "0"-48
JP NZ,prno
LD (IX)," "-48
INC IX
DJNZ sizlp
.prno ;print number
LD B,6
LD IX,decdigits-1
.prloop ;with leading space
LD A,(IX)
ADD 48 ;to get ASCII value
CALL &BBSA
INC IX
DJNZ prloop
POP BC

```



```

POP IV
POP IX
JP searchloop ;carry on

.decdigits
DEFW 1000,1000,100,10,1
DEFB " -48 ;leading space

.decdigits
DEFS 5

.nameelen
DEFS 1

.linadr
DEFS 2

.linenc
DEFS 2

.namestore
DEFS 40

**** Initialise RSX's *****
.enablersx
LD HL,flag
LD A,1
CP(HL)
RET Z
LD(HL),1
LD BC,rsxtable
LD HL,workspace
CALL ABCDI
RET

.set flag
;log IVFIND

.done before?

.workspace
DEFS 4

.flag
DEFS 1

end

DEFB 'VFIND',0*+100

```



Give your fingers a rest....
All the listings from this month's
issue are available on cassette.

BOOKS

BOOKS

BOOKS

Amstrad Advanced Programming Techniques

This book is designed to show readers how to develop serious applications programs. It emphasizes the importance of good program planning and design, beginning with a look at modular programming, and then going on to consider debugging, string handling, inputting, error trapping, storage and retrieval, sorting, data structures and searching.

CAT #:3003 PRICE: \$24.95

The Amstrad Companion

This book shows how the power of the PCW 8256 and 8512 can be applied to more than word processing. *The Amstrad Companion* is an authoritative guide to working with the CPM operating system and to practical applications using the Logo and Basic programming languages. Included in the book are updated versions of programs which are being used in more than 20 countries around the world to file data, issue invoices, keep simple accounts, create budgets and display data in the form of graphs.

CAT #: 3101 PRICE: \$27.95

FOR A FULL LIST OF BOOKS
CALL STRATEGY SOFTWARE
ON [002] 29 4377

Games and Graphics Programming on the CPC 464,664,6128

This book assumes some knowledge of Basic and is written for those who wish to become more familiar with their Amstrad. The author introduces the Amstrad's graphic capabilities and demonstrates how to use them to produce animated games and static graphics displays. One of the main aims of the book is to help in the design of long programs by looking at program structure and the way in which a large program can be constructed practically from small segments.

CAT #:3009 PRICE: \$29.95

Practical Amstrad Word Processing

PAWP is a complete self-tutor for anyone trying to get to grips with Locoscript on the Amstrad PCW computers.

The self contained sessions in the book lead step by step through the concepts behind practical word processing and the use of the powerful Locoscript facilities. The areas covered in the sessions include not only the ways to improve the presentation of documents, but working methods which use Locoscript's file groupings, layouts and templates to dramatically cut the time needed to prepare a wide range of standard documents and forms

CAT #:3102 PRICE: \$21.95

BUSINESS COMPUTING WITH THE AMSTRAD

MAY 1987

42 BUSINESS NEWS

READ ALL ABOUT IT. HOW THE AMSTRAD IS CONTINUING TO EXPAND INTO THE BUSINESS WORLD. FIND OUT WHAT OTHERS THINK OF AMSTRAD'S NEW PC AND HOW IT CAME OUT IN A CLASH WITH BIG BLUE IN THE LATEST RATINGS.

43 PRINT DESIGN

THE CONCLUDING PART OF OUR SERIES ON HOW TO PRODUCE PROFESSIONAL LAYOUTS USING LOCOSCRIPT AND SOME ADVANCED PLANNING.

46 BRAINSTORM

CONSTANTLY LOSING TRACK OF YOUR THOUGHTS AND PLANS? THIS PACKAGE FROM CAXTON COULD HELP PUT YOUR IDEAS IN ORDER.

48 MINI OFFICE II

JO STORK CONCLUDES HIS SERIES BY MAKING COMPLEX CONCEPTS LUCID AND ATTRACTIVE. THOSE FOLLOWING OUR MINI OFFICE II SERIES NEEDN'T PANIC - WE'VE GOT A NEW SERIES STARTING NEXT MONTH.

50 BUSINESS MAIL

AS ALWAYS, YOUR LETTERS PROVIDE A CROSS SECTION OF INTERESTING AND ENLIGHTNING MAIL WITH HINTS AND TIPS TO SUIT ALL. YOU'LL READ ABOUT HOW LOCOSCRIPT FILES DISAPPEAR ON A 6128, HOW TO IMPROVE YOUR W's AND MORE IN THIS MONTH'S MAIL BAG.

51 PROTEXT REVIEW

ARNOR'S LONG-AWAITED CP/M-BASED ALTERNATIVE WORD PROCESSOR TO THE PCW'S LOCOSCRIPT HAS ARRIVED. PROTEXT IS AVAILABLE FOR THE CPC 6128 AS WELL AS THE PCW's. IS IT THE BEST EVER? - WE'LL SEE HOW IT SHAPES UP IN WILLIAM BENN'S COMPREHENSIVE REVIEW.

53 WE SEE TWO PC C COMPILERS!

WHILST YOU'RE TRYING TO SAY THAT QUICKLY, WE TAKE A LOOK AT THE C LANGUAGE AND COMPARE THE ZORLAND AND MIX PACKAGES, TWO LOW-COST IMPLEMENTATIONS FOR THE AMSTRAD PC 1512.

55 MICROFILE/MICROWORD

AN IN-DEPTH EVALUATION OF AMSOFT'S COMBINED PACKAGE - A FLEXIBLE DATABASE AND A FAST WORD PROCESSOR. IS IT THE ONE FOR YOU? - READ ON AND FIND OUT.



Computer Oasis
Grove Plaza Shopping Centre, Stirling
Hwy, Cottesloe, WA 6011
(09) 385 1885

SAVE

FREEZE FRAME

SUPER DEALS

Rated #1 **AMSTRAD** IBM Epson

ATARI'S COMMODORE

3" AMSOFT DISCS
NOW ONLY \$76.50
for a box of 10

PRICE PROMISE
We will beat any genuine price on **ATARI ST** - phone for details.

Amstrad! 5.25" Second Drives
40 track Single Sided Drive
40 track Double Sided Drive

SPECIAL!
AMSTRAD PC1512
Phone for latest prices.



AMSTRAD PC

This is only a sample of the items available please ring for details of items not shown. All items offered subject to availability

- LATEST SOFTWARE RELEASES
- PROCOPY
- MONITOR LEADS
- 5 1/4" DISC DRIVES
- TOS-ON-ROM
- MONITOR AND PRINTER STANDS
- DUST COVERS
- BACK-UP UTILITIES
- MONITOR EXTENSION LEADS
- MODEMS
- DUST COVERS
- PRINTER CABLES
- PRINTERS







ATTENTION

COMPUTER CLUBS • DEALERS
We offer big volume discounts!

CALL TODAY

CALL BEFORE YOU ORDER:
OUR PRICES MAY BE LOWER
& AND WE OFFER SPECIAL
MAIL ORDERS!

Computing With The Amstrad - May 1987

41

Amstrad surges ahead

AMSTRAD's latest trading results are not only good news for shareholders, they have ensured ongoing development of the PCW.

Critics had suggested that any success the PC1512 enjoyed would be at the expense of Amstrad's word processing micros, and that these would be allowed to quietly fade away.

But Alan Sugar's marketing instincts have again proved to be spot on and it is now clear that both the PC and PCW are market sector leaders in their own right.

A jubilant Sugar told shareholders: "The famous PCW8256 word processor continued with buoyant sales, a fact which was most encouraging as it was felt by some that the introduction of the PC1512 range would affect PCW sales.

"I am pleased to report that there was no such turnaround, which confirms our view that we have two separate markets in business computers - the word processing sector for typewriter replacement and the personal computer sector.

"This has scaled our desire to continue to develop more products in the PCW8256 word processing family."

Although Amstrad has not given details of what these products will be, Sugar went on to say: "Later this year we will launch a host of computer products to see us well into and beyond the next financial year".

Amstrad reported pre-tax profits of \$178 million on a turnover of \$681 million in the second half of 1986 - an increase in sales of 112 per cent and in profits of 159 per cent.

Major single factor in the

company's success was the PC1512 which accounted for a quarter of all sales in the period.

ICI approves

CHEMICAL industry giant ICI has given the PC1512 its most important boost yet after field trials lasting three months. In a letter to Amstrad, ICI director of information technology Derek Seddon said: "The PC1512 met all our requirements for an 808x processor-based personal computer and is judged to offer very good value for money.

"During trials no problems were experienced with overheating when the PC was connected to a token ring network.

"The Amstrad PC1512 has now been approved for purchase by ICI operating units".

Options on display

NEW for PCW owners who don't want to keep referring to their LocoScript manual or drop-down menus is the Menu Mate keyboard template from Richman Software.

It contains a complete list of all option abbreviations, special key combinations and copy, cut and paste instructions. Text is shown in the exact pitch size it appears on paper. \$17.50.

IBM LOSES TOP SPOT

Amstrad has pushed IBM off the number one spot in the UK business micro market for the first time.

Latest figures show Amstrad with a 26 percent share compared with 25 percent for IBM.

It is a remarkable achievement for

Amstrad, which was trailing IBM by 27 percentage points only two months earlier.

The statistics, compiled by leading market research company Romtec, show that Amstrad has overtaken its rivals due to sales of the PC1512.

Romtec says the majority of PC1512 sales have been to first time users.

PC 1512 IS BEST BUY

The first major survey of business micros since the arrival of the PC1512 has named the Amstrad machines "clear best buys" among IBM XT compatibles.

Independent assessment in What To Buy magazine rates the PC1512 ahead of Compaq, Epson, Tandy and even IBM itself.

The findings were based on hands-on experience of the magazine's own technical team, response to user surveys, software and dealer support.

What To Buy described the PC1512 dual floppy system as "outstanding value".

The hard-disk based version was lauded as "amazingly good value even though this model only gives you 10 megabytes against the more usual 20 megabytes".

2D CAD

New 2D CAD package Grafpac from Grafsales turns Amstrad PC and PCW micros into CAD workstations.

It operates with a desktop A tablet coupled with stylus, and comes with powerful software ready configured.

All operating commands and keyboard functions are printed on an overlay placed on the graphics tablet. Prices are \$499 for the PC version and \$375 for the PCW version.

THE word revolution is overworked, but it really is the only way of describing what is happening today in the world of print and design. It is a hidden, largely mysterious revolution to most people, but Amstrad owners can be there at the heart of it, reaping the benefits.

The revolution stems from the speed with which ideas can now be turned into print, the vast variety of forms that finished material can take and the relative cheapness of access to the most advanced processes.

Most people have heard about direct input at newspapers, where journalists key in their stories for typesetting. It is exactly that process which is now available to almost anyone who can create a document on a disk.

And if you're set up for electronic

Pages with panache...

Concluding Bruce Hugman's article on lost-cost printing with the PCW

communications the possibilities are even greater. You can now not only write a book at your keyboard but also send it off for direct typesetting down the telephone or from your disks.

Or, if you have no immediate plans for winning the Booker prize for 1987, there's a host of applications if you're running a small business.

In fact, if you're involved in any kind of organization which needs to produce programs, broadsheets, newsletters, promotional material, stock lists or catalogs there are great opportunities - and savings.

Here we'll just sketch in some of the possibilities and hope they'll give you the confidence to take the plunge and - above all - to experiment!

Figure 1: The copy for this newsletter was laser set from Amstrad disc on an Apple Macintosh and has screened photos included.



In the normal course of events, if you want something printed you send a hard copy of your material to a typesetting house or to a printer, where someone sits at a keyboard and inputs the material all over again.

At anything up to \$175 an hour this is an expensive process and if the material is very complex some errors are almost bound to creep in.

There are now many typesetting companies who have translation equipment for taking data directly from disks, some able to cope with over 700 different formats.

But all they do to your document is to insert typesetting commands. These determine the style and size of the type used, the length of lines and so on, and the amended text is then fed into the firm's machine.

All you have to provide is an Ascii file - some services will even do this for you - with a hard copy marked up as to how you want the

material set.

You'll get your text back on a strip of paper a galley, which is then ready for cutting up and turning into artwork for plate-making and printing.

If you have a modem and the appropriate software you can send your file down line to some companies.

This is a slightly more complicated process which requires you to embed the typesetting commands into your document - but if you've managed to get on top of LocoScript you'll have no trouble mastering that process from the manual supplied.

You'll probably find a typeface manual helpful while you're preparing your material. And if you have had no experience with print and design you may want the help of someone who has during the early stages.

However, careful study of existing printed material and referring to your type manual should get you well on the way.

There are several options in terms of the final cost and quality of typesetting and you can choose the one which fits your purposes and your budget.

Direct from disk, the cheapest is daisywheel setting which gives a clean, smart finish for things like company reports or internal documents at a cost of around \$12.50 per A4 page.

If you're going to want a lot of daisywheel output, however, it will almost certainly be cheaper and more convenient to buy your own printer.

The next option is laser setting from an Apple Macintosh. While some professionals have reservations about the quality of this, few amateur eyes would see anything wrong with it. This costs about \$25 per A4 page and Figure 1 is an example of the finished work.

Finally, you can go to the top of the market using high quality digital photo or laser setting on very expensive equipment.

Database Publications produces this magazine this way - but for the individual it can work out rather more expensive than the alternatives previously mentioned.

In general, these options will save you money when compared with conventional typesetting (which comes in at \$75 - \$100 per page when you have four or five A pages or more).

On the hand, down the list setting, for which you are charged pro-rata for the actual material set and which provides high quality photo setting, is far away the cheapest at about \$75 per foot (which is more or less a A4 page) and is good value for even the smallest item.

Otherwise letter heading business cards, invoices and other small items are likely to be more expensive than if you simply hand them over to a printer.

Your typesetting will usually come back through the post in a few days and you then have to lay out and paste it up ready for printing (see the panel for a step-by-step guide).

Two color printing is not a great deal more expensive than one color if you find a printer who has a machine that prints two colors at the same time. But full color (with color photos for example) is very expensive. However, if you're in that market you'll hardly be slaving away over your Amstrats to save a few dollars.

Remember that for short runs (up to a hundred or so) good quality photocopying can be a sensible option.

With little imagination and careful choice of typeface, paper and ink color you can produce very classy material as you can see from Figure II. And remember that once you have mastered the process you have a highly marketable skill.

The process may sound a bit complicated at first, but once you've tried your hand at it you'll

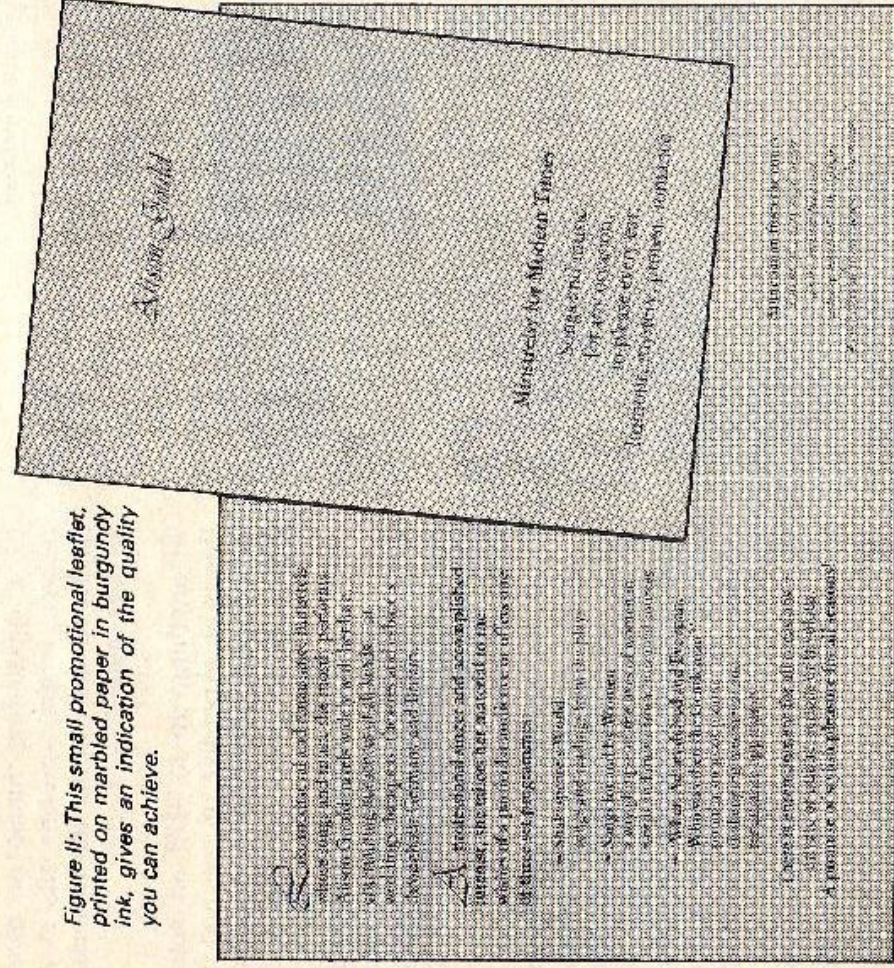


Figure 11: This small promotional leaflet, printed on marbled paper in burgundy ink, gives an indication of the quality you can achieve.

find it's quite manageable and you should be delighted with the results.

If you use your imagination you'll find there's no end to the projects you can undertake for yourself and for others: Programs for amateur plays, magazines for any kind of organization you can think of, stock records, catalogs, even a Christmas newsletter (you've just got time to get started for 1988!) - you'll find they're all well within

your grasp.

Once you have a high quality finished product, do make sure that it's presented well. If it's promotional material or a catalog you can place or staple it in a card folder or a clear plastic wallet with your company's name on the front.

If you're going to be doing a series of productions (monthly newsletters or catalogs, for example)

then there's much to be said for having the masthead title or company name printed in bulk in advance and copying each month's or quarter's publication on to the required number of sheets.

This is just a start, but here's hoping it's given you some of the information and inspiration you require to begin producing some impressive material from the keyboard of your Amstrad - and possibly bringing in some extra income as well.

A step-by-step guide to production

Here are the basic steps for the production of a simple, low budget, four page A4 newsletter, promotional brochure or report, printed on an A3 sheet folded once.

These notes are for preparing a disk to hand to a typesetter - the procedure for electronic communication is comparable but different in detail. Firms offering this service provide user manuals.

- Format a new disk.
- Create a standard LocoScript document with all text, including titles, headlines and sub-headings.
- Edit and correct (remember any mistakes you make will appear in the typesetting). Because you will be making an Ascii file (which strips all commands out) you don't have to worry about layout and style in your editing.
- Print out a draft copy.
- Do a rough layout of how you want this final document to look.
- Decide the type style and size you want for each title and block of text and write it on your hard copy (you'll need to have a type style manual to hand, ideally one from the company you're going to use for the setting).

- Back-up your disk.

- Using copy/cut/paste re-group the material so that everything in one style and size appears together (for example, all your 36 point Times Roman Bold titles, and all your 10 point Helvetica Medium text and so on).

- Check that the content is exactly as you want it to appear in print, then print out two copies (one for your reference and one to send off).

- Mark up the hard copies with the size and style of type for each section, what the maximum line length is to be, which lines are to be centered and so on. This has to be done manually because your Ascii file will be naked of layout and style commands.

- Make an Ascii file.

- Back up the disk.

- Send off your disk and marked up hard copy.

- While you're waiting for the setting to come back through the post collect any photos or other visual material you're going to include and refine your rough layout of how you want the final document to look.

- Prepare two white boards, slightly larger than A3, by drawing an A3 sized outline on them and dividing each into two A4 sized pages. Remember the pages will be 4 and 1 (in that order, left to right) and 2 and 3 when they are flat.

- When your typesetting arrives, cut out all the bits and pieces with a scalpel and start arranging the jig-saw on your boards, leaving space for photos and illustrations.

- Measure up the spaces for photos and take your originals along to a copy shop which can make you a screened PMT (photo mechanical transfer) to the exact size you require. Get copies of other visual material in the same way, though it probably won't need to be screened.

- Next paste everything down using aerosol adhesive, which allows you to move stuff about for some time after it's applied.

- When you're quite happy with the look of it - and all your lines are straight! - take it along to the copyshop or printer who has offered you the best price for the job.

- Finally, bask in the glory of having achieved what you previously thought impossible.

Get your mind into top gear

BILL AGNEW takes a thoughtful look at BrainStorm, a program designed to produce structured ideas

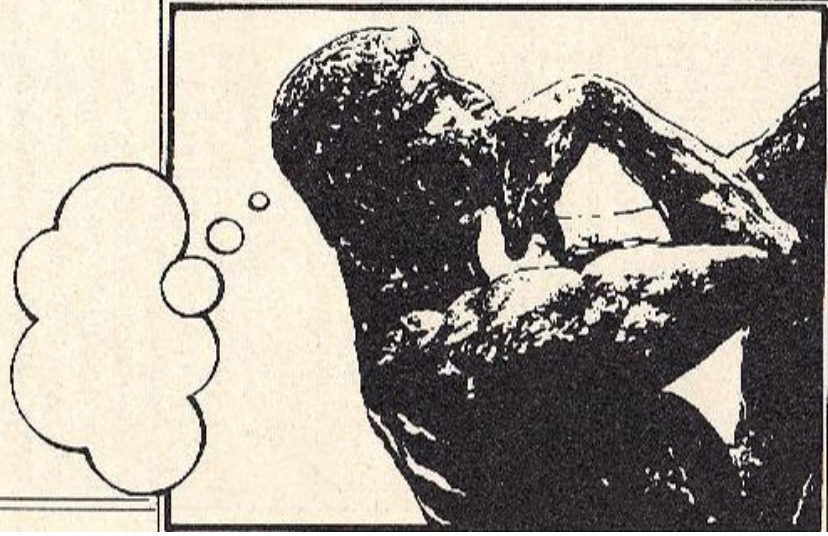
further descendants from them, and so on. The topic, the issues, the key points, and (say) one set of details - in that case, four levels.

BrainStorm will help you delve deep, planting your thoughts at each level so that they stand a chance of sprouting into other levels, while noting any ideas that recur.

The result of your ideas session can be examined at any time on screen, or sent to printer - in which case BrainStorm will indent each level appropriately, thus providing a visual feedback of structure.

The attractive glossy black packaging, with its logo of The Thinker really sets the mood to enjoy the program. Unfortunately, mood isn't everything, and my first twenty minutes of BrainStorm were nearly my last.

My initial discovery was far from enjoyable - different meanings for the PCW's keys instead of the comfortably dedicated LocoScript set-up. To have to give up the hard-won mastery of the keyboard was hardly welcome.



BRAINSTORM from Caxton is effectively an electronic scratchpad which can be used to create structured lists of items from random jottings and to show the relationship between them.

Basically, you decide on a topic and make a list of the main issues, then take each of the issues and make a list of key points. Then you detail each key point, create

And in addition to getting used to an unfamiliar keyboard configuration there were a dozen or so special control to learn - Control D = Next, Control + H = Home (Search for), and so on.

However, BrainStorm awaits and I persevered. On the whole, it was worth it.

So, what are the benefits of the ideas processor, supposedly the greatest thing since pencil and paper?

Brainstorming is a well-tried technique used in group work for generating ideas. Everyone says the first thing that comes to mind on a given topic, and these random ideas are noted down, usually on a blackboard or something similar for everyone to see.

They are then put into some kind of order by the group leader, and the brainstorming and ordering continue until all ideas are exhausted.

They are then put into some kind of order by the group leader, and the brainstorming and ordering continue until all ideas are exhausted.

BrainStorm the program works in a similar way but is meant to be used by an individual rather than a group. Of course, it does not generate the lateral thinking and cross-fertilization of group interaction, but it does invite you to dig into your own personal construct, with your own logic, and to make connections between ideas. It invites you to take a topic, and look into it at the next level down

Figure 1: A typical level structure

Training and development	(General	-1st level)
Management development	(Topic	-2nd level)
Mid-career refreshing	(Phase	-3rd level)
Political/interpersonal skills	(Content	-4th level)
Technological up-dating	(Content	-4th level)
Career planning	(Content	-4th level)
Learning techniques	(Methods	-5th level)
Measurements of progress	(Assess	-6th level)
Self assessment	(Assess	-7th level)
Performance evaluation	(Assess	-7th level)
Cost effectiveness	(Assess	-7th level)
Review and redesign	(Phase	-3rd level)

From that level, one item may lead into the next level and so on, practically ad infinitum.

In this way BrainStorm provokes the development of thinking, using keywords to take your thought process to the next level - and in doing so, attempts to clarify structure and content for you. You do the brainstorming, and Brainstorm tries to help you do the ordering.

Take a look at Figure I: The levels are rather like the headings in a filing system. Into each of them you can put your list of details, and any one of these can be extended at the next level, as shown in the example for levels 4 and 7. Note that level 3 is not only the Phase, "Mid-career refreshing", but also the "Review and redesign" - a simple example of what is called recursion, and one which could of course be taken much further.

How does all this happen in practice? As soon as a word is put in, the model implicitly questions its meaning, position, content and relevance. It may, for instance, show you the relationship between a parent idea and its descendant - something you may well otherwise have missed.

This is a useful discipline, often fun but sometimes rather disturbing - the mental construct of your expertise may not be as straightforward as you thought.

To start you off, Caxton have included some sample constructs - diary, address book and program design. They also invite users to let them know of other applications.

BrainStorm provided a unique aid to the hierarchal development and structuring of ideas in depth, and at \$125 is cheap if you are the kind of person who generally finds it hard to get some order into those brilliant concepts that flood from your brain day and night.

Product: BrainStorm
Price: \$125

= Training and Development

- Induction
- Knowledge Base
- Operating Know-how
- Data
- Credibility development
- Basic Skills
- Social Skills
- Managerial Skills
- Key Issues

Enter text at > or press command key (! to identify command keys)

Line 14 of 14 Level 1 New left 43K

A sample screen at level 1

Drive is B:

= Induction

-)- To belong
- To work
- To identify with firm
- To relate to others
- To know the rules
- = To know the ground rules

Enter text at > or press command key (i to identify command keys)

Line 1 of 7 Level 2 New left 43K

A sample screen at level 2

Drive is B:

= To know the ground rules

- Dress
- Manners
- Work-compliance
- Timekeeping
- Use of time
- Sickleave
- Enterprise
- Drinking
- Eating

Enter text at <

RET finish ESC abandon TI tab TA del left TS left TD right
Line 3 of 10 Level 3 Men left 438 TF del right

A sample screen at level 3

Make complex concepts lucid and attractive

JO STORK concludes his series on making the most of Mini Office II

We looked last month at the basic options within Mini Office II's Graphics module. This month we will look at the remaining facilities - designing and shading pie charts.

Since a key feature is its ability to generate graphs from the spreadsheet, you should start this session by creating the simple spreadsheet shown in Figure 1. Do not worry about creating formulae, just enter the variables directly.

Having done this, select the Save graphics data option. This allows you to create on file, in a form which is readable by the Graphics module, one data set at a time. Before replying to the prompt Select from row/col (R/C) stop to think for a moment.

You will recall from last month that each data set can contain no more than 15 values, therefore had you created a large spreadsheet you would not have been able to graph the whole of a row or column. Furthermore, when selecting data for a data set, you can only select from single rows or columns.

If you do need to graph data from

several rows or columns, you can either create separate data sets or re-design the spreadsheet itself. Fortunately this latter is very unlikely.

For our example press R. The spreadsheet we created has its titles in row 1 therefore, to the prompt Row for TITLE enter 1. The prompt Row for NUMBER refers to the spreadsheet row containing the values to be graphed. In this instance enter 2, but when completing what follows work through this portion again and enter 4. Finally repeat once more and enter 6.

You will now be asked the filename under which the data set will be saved. I suggest you enter ROW2 - with ROW4 and ROW6 for the subsequent attempts. You will then be presented with a series of cell references and the data they contain. Respond Y or N depending on whether you wish that cell to be incorporated in the data set. Since A2 is empty the obvious answer is N.

To the following five cells offered reply Y and you will see the data set slowly being built up on the screen. Reply N when G2 and

subsequent cells are presented. Once the row has been completed you will be asked Save more data (Y/N). Reply Y and repeat the above for ROW4 and ROW6.

Sufficient data to work with has now been prepared, so return to the main Mini Office II Menu and select Graphics. Once the graphics menu is displayed select Load Data so that the data sets created above can be made available to this module. To the question Data Set 1,2 or 3 reply 1 and then enter the filename ROW2. Repeat the sequence for data set 2 - ROW4 and yet again for data set 3 - ROW6.

You have now completed transporting three data sets from the spreadsheet to the graphics module. You can prove this by choosing a three line set bar chart as described last month. What is satisfied that the data sets transportation was successful select the Pic Chart option and you will see that the same principle of icons has been adopted. The icons reading from top to bottom stand for the following:

- An ordinary Pie Chart
- An emphasized Pie Chart where some or all segments are slightly offset from the main circle
- Choice of segment shading
- A screen dump

Select the ordinary Pie Chart option and data set 1. The pie will immediately be drawn. To the question Text (Y/N) reply Y and again for Underline (Y/N). Type SEGMENTS CLEARLY, press Enter and move the text to the position shown in Figure 1. Repeat similar text entry with the word DIFFERENT. Note that the under

A.....		B.....		C.....		D.....		E.....		F.....	
1		COST 1		COST 2		COST 3		COST 4		COST 5	
2	ITEM A	1.00		2.00		3.00		4.00		5.00	
3	ITEM B	6.00		7.00		8.00		9.00		10.00	
4	ITEM C	11.00		12.00		13.00		14.00		15.00	
5	ITEM D	16.00		17.00		18.00		19.00		20.00	
6	ITEM E	21.00		22.00		23.00		24.00		25.00	

Figure 1: Spreadsheet data for transfer

lining has a break in it.

When you are ready move the cursor to the emphasized pie chart icon and press Enter. To the question Data set 1,2 or 3: reply 3. You can either isolate all the segments or individual ones. Consequently when asked Isolate all Segments? reply N in this instance.

As each segment is presented reply YNNNN in turn. Again reply Y to the question Text (Y/N). Enter the text SEGMENTS SIMILARLY, move it to the right and when the prompt appears reply Y and enter the text SIZED. Move this into position and the final chart will look like Figure III.

The final option we'll look at is the selection of shading characteristics. Before continuing try and memorize the shading, or better still, print off the pie chart. When ready move the cursor to the third icon and press Enter. Choose data set 3 again.

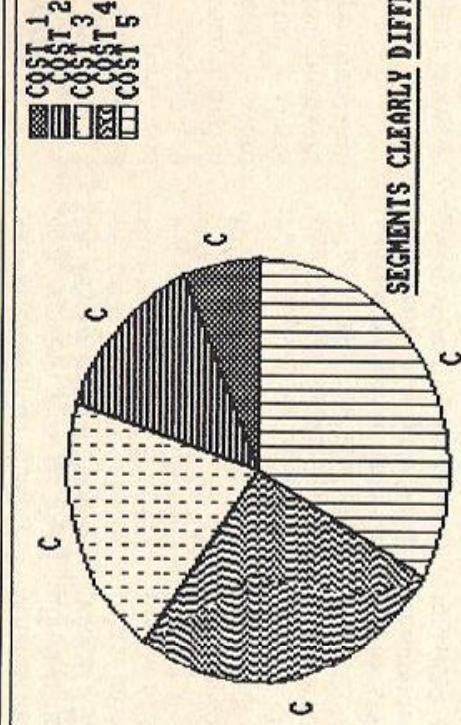
You will see a series of 20 shading options appear along the bottom of the screen. Move the cursor arrow to the last option on the right and press Enter. The first segment will now shade solid black. Now move the cursor left to the eighteenth and press Enter. Note the following two points:

- A marker will appear over the twentieth option indicating it has already been selected.
- While you have not changed your choice from the emphasized pie chart option, the graph has reverted to ordinary.

Repeat with the thirteenth, tenth and seventh shading styles. Reply Y to the Text (Y/N) prompt and enter SHADING AS CHOSEN. Locate the text as shown in Figure IV and press Enter.

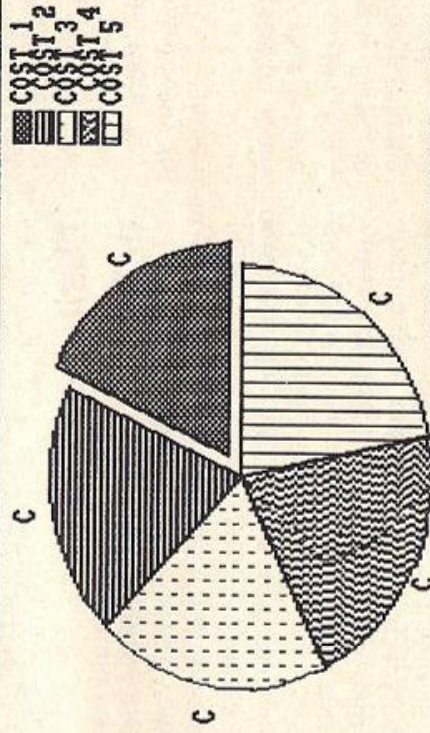
It should now be obvious that

with a little pre-planning of the spreadsheet, plus suitable selection of the graphing facilities discussed these last two months, you should be able to present complex mathematical concepts in a clear, concise and highly attractive fashion.



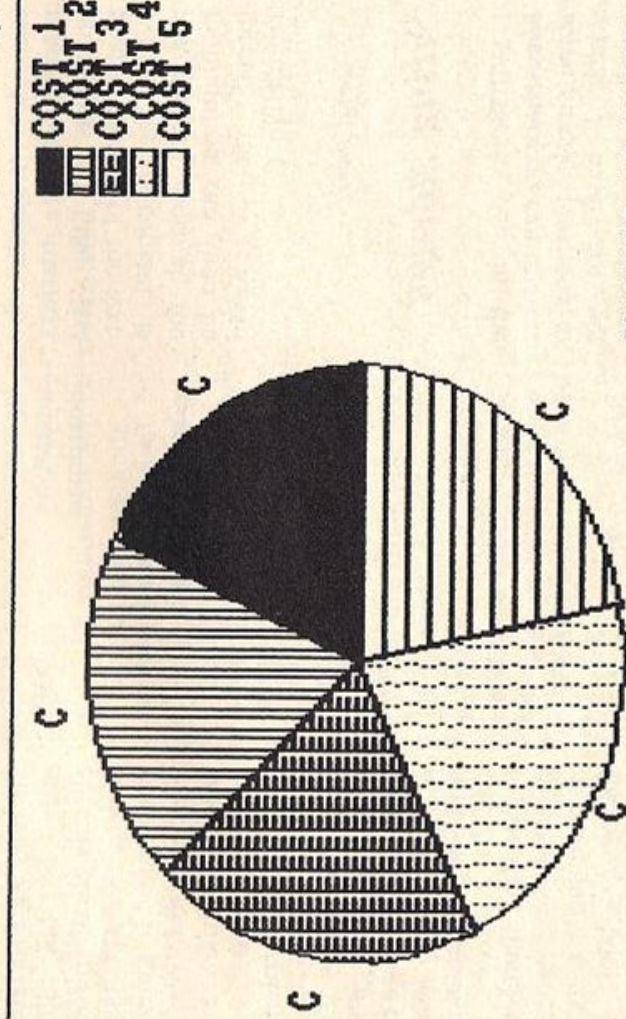
SEGMENTS CLEARLY DIFFERENT

Figure II: A simple pie chart



SEGMENTS SIMILARLY SIZED.

Figure III: A segmented pie chart



SHADING AS CHOSEN

Figure IV:
Choosing
different
shading

Page breaks

I BOUGHT an Amstrad PCW8256 last June and have been using it extensively since. I find it an excellent word processor but have experienced two problems which I hope you can solve for me.

I find it is very slow in scrolling down. If I have a document of, say, six pages and want to edit on page five, I press the Page Down key, but the text scrolls down line by line stopping at the end of each page. Is there any way of getting directly to page five?

Secondly, a question about printing. Say that I have 10 files each of 12k and I want to print them in sequence without having to start off each individual print run. What do I do? - Edmund M. Hogan

• The Page key on the PCW is not the kind of Page Down key found on a number of word processors. It doesn't allow you to go to any page number but merely sends you to the next page. Hence the fact that the scrolling stops when it reaches a page break.

If you want to scroll down automatically, use the Doc key, then the Stop key twice when you reach the page you are looking for.

Alternatively, to get to page five from page one, press the Page key four times without waiting for the text to start scrolling.

LocaScript is slow when scrolling because it formats the text ready for the printer whenever the cursor moves over it, and very little can be done to speed things up (using drive M is marginally quicker, but dangerous).

So if you want to get to page 25 of a document, press the Page key 24 times, and put the Keilite on. LocaScript is excellent for those who write long documents and need regular breaks, but unfortunately of the tea or coffee, not the page, variety.

I'm also afraid that there is no way of doing a multiple print run of joined files. You have to set each one in motion separately, which means once again leaving the PCW to print one file, and coming back to re-start the next one. So you get some exercise too. Who said LocaScript can damage your health?

Disappearing act

WITH WordStar on the PCW, when you are scrolling through a long document using Shift plus the down cursor key, rather disconcertingly about a page and a half of text occasionally seems to go missing.

However the document is in good health, it is just that WordStar is temporarily befuddled by the amount of updating it has to do.

You can get the missing text to re-appear by placing block markers anywhere in the last proper section of the text. When the document is restored to normal the highlighting can then be turned off. - Kevin Donnelly

• It works! Thanks.

String storage

I WRITE to you in the hope that you can answer some simple questions to which I have not yet found a satisfactory answer from hardware suppliers.

what is the best way to go about it on a 6128? Is the answer Pascal or some other Basic than the one on the 6128?

The reason I ask these questions is that I am finding the 6128 a bit small for some jobs.

I was converted from a Commodore just 12 months ago and ported by own program structure and battery of business user interfacing sub-routines across from the Commodore to the 6128.

I could use the PCW with off-the-shelf software which computer suppliers tell me will let me do anything. I don't believe them.

I have spent enough of my time with dBase III Plus on a Compaq to realize that it is a dinosaur and that I can program a 6128 in Basic - given my battery of sub-routines - much more easily and quickly than I can program the Compaq using dBase III.

And the result is much better, with more ergonomic interfacing. Stan Boardman

• I'm not sure exactly what you mean by 'string storage'.

Most programming languages limit string length to a maximum of 255 bytes, full stop. But I suspect you really mean Transient Program Area (TPA). If so, your problem is a common one.

The TPA represents the amount of active memory available for temporary storage of program and data (whether string or numeric).

On 8 bit CP/M machines this is limited to about 61k, and that figure can't be extended either by using a different programming

language or by adding a upgrade.

Some programming languages work in a modular fashion external routines, allowing (even all) the 61k of TPA to be used for data - but the limit applies. A PCW8512 has 512k ram, but this gives you a big drive, not a big TPA.

Like many other programs dBase gets round the problem, even 16 bit machines with a large TPA, by making frequent accesses.

Basic, Pascal and most other languages can be made to do the same. When used in that way however, the code can be just as complex as that of dBase.

Basically you need either upgrade to a 16 bit machine like the PC1512, or to write programs which do not confine themselves to data held in memory, or to both.

Assembly point

I HAVE owned a PCW8256 for the last 18 months and recently decided to take a plunge and learn assembly language.

To this end I bought a copy of Ian Sinclair's excellent book *Introducing Amstrad CP/M Assembly Language*. I found it very easy to follow and next to no time I was able to produce simple programs of my own.

However, I now feel that I would like to go further with this language but at a loss as to where to find more information.

I would therefore be grateful if you could recommend a suitable follow-up book particularly with reference to the use of graphics. - A Chessum

• Ian Sinclair's book, excellent though it is, restricts you to 8080 assembler language. There is a huge Z80 instruction set that most books dedicated to CP/M would not cover.

The best book we've found on Z80 is *Programming the Z80* by Rodney Zak's published by Sybex, but although it is probably the best reference manual around it does not teach you how to use the information contained within its pages...

You will also find one of either of the following books essential if you are to pursue CP/M:

CP/M Plus Handbook by Digital Research published by William Heinemann; or *The Amstrad CP/M Plus* by David Powys-Lybbe and Andrew R.M. Clarke published by The Lavenham Press.

The latter also contains information on G.S.X. CP/M's graphics environment.

Roland Waddilove has also done some valuable research in directly accessing the PCW's graphics screen and his article will appear in a future issue of *Computing with the Amstrad* and should be of some valuable assistance to anyone wanting to dabble in this.

LATERAL THINKING

The appalling W on the PCW can be improved in the following way. If you are using 12 pitch or proportional spacing, type:

+p10 (Enter)
W+pp (Enter)

This will produce a 10 pitch W in a 12 pitch document. Save this as a phrase under W and use when needed. The improvement is not astounding but it helps. T.J.Nott

• An excellent idea, an example of lateral thinking (as it were). However, if you substitute W-p the result is even better.

The amiable alternative

WILLIAM BENN enthuses over Protex on the PCW

PROTEXT first appeared on the CPCs nearly two years ago and was quickly recognized as the definitive word processor on these machines, a position it still holds today.

LocoScript on the PCW seems to be a bit of a dinosaur in comparison - powerful but slow.

I am always conscious of LocoScript when using it, whereas CPC Protex has become instinctive. So it was with a feeling of anticipation that I loaded Arnor's upgraded version for the PCW and CPC6128.

It runs under CP/M and comes complete with spelling checker, mail merge, tutor, disk utilities and manual.

The first job is to install Protex for your machine and prepare working disks. This is automated and provided you follow the instructions carefully should not present any problems.

It soon becomes apparent that Protex is designed very differently from LocoScript. Instead of the familiar menu system you are provided with a set of commands which are entered from Command mode.

This is distinct from Edit mode where the text entry is done and pressing Stop on the PCW or Escape on the CPC6128 toggles between the two modes.

Selecting Command mode (shown in Figure 1) opens a window at the bottom of the screen and a prompt awaits your instructions.

The range of commands is extensive but the use of sensible names makes them easy to remember. Many can be abbreviated as you become familiar with the package and I find this

system to be quicker than LocoScript's menus.

For instance PRINT, which can be abbreviated to P, prints your text. One of the variations on this command is PRINTS (or PS) which prints to the screen so you can see how the formatting will appear on the printer.

CALC (CA) will perform simple calculations, the results of which can be inserted in the text. COUNT does a word count, DIR or CAT will catalog a disk and so on.

When editing you have the choice between Document and Program modes. The latter is a text editor and does away with many of the formatting features.

It produces pure Ascii files suitable for compilers or loading into Basic. Document is the default mode and is for normal word processing.

If your experience is limited to LocoScript you will find the speed with which you can move about the document a revelation. Protex achieves this mainly by formatting for the printer only when you tell it to.

Pressing Alt+F (or Control+F on the CPC) will reformat the current paragraph, or exiting to Command mode and typing FORMAT will process the whole document. This contrasts with LocoScript which regularly reformats as you work.

Commands which are executed when printing can be embedded in the document. These might include setting margins and other print options, loading a printer driver or reading in a block of text from disk.

Protex tries to hold everything in

High level language

IAN SHARPE compares the merits of two low-cost PC packages that bring C within everyone's reach

AMSTRAD's low cost IBM clone has brought many new users into the PC software market. Several will have upgraded from a home micro and are likely to be horrified at the high price of software.

Until recently you would not have been able to get an MS-DOS C compiler for less than \$250 and could easily have paid many times more than that. However this situation has changed with the two low-cost packages from Zorland and Mix reviewed here.

Zorland is well documented but the Mix manual is superb. In fact the Mix offering would save the beginner the cost of a separate book.

There is an extensive tutorial and full explanation of all the library functions with examples - all a newcomer to C could want.

The Zorland documentation, while complete with a tutorial, would in no way substitute for a book on C, so this would have to be added to the cost of the package.

Neither compiler is sold specifically for the Amstrad PC. Most PCs come with a linker on the MSDos disk which is needed by a compiler to link the object code with the run-time routines. Unfortunately this is not the case with the PC1512.

Mix comes with a linker, but Zorland does not, so it must be bought as an extra. Zorland can supply one at \$25 but the cost of this and a book bumps the price up by a total of \$50 to \$75 for the first-time user.

Both packages are supplied with WordStar-compatible editors. In the case of Mix the editor is sold separately though a discount is available if you buy the compiler and editor together.

The phrase WordStar-compatible does not mean that you are getting a full function WordStar lookalike but that whatever facilities are provided are accessible using WordStar commands.

The Zorland editor supports only a small number of these and I felt that it would not be adequate for any serious program development.

So when considering Zorland you must also bear in mind the possibility of having to buy a separate editor. Though any word processor capable of producing an Ascii file - which most people will have anyway -

will do the job.

Mix scores once again on documentation, as a 98 page manual for text editor is included.

The Mix editor allows you to execute Dos commands and compile from within it. Any compiler errors will return you to the offending line with the cursor positioned ready for the necessary alterations. The compile/edit cycle is therefore quicker and the editor provides a friendly and productive environment.

The documentation is due for revision, as extra features not mentioned in the manual have been added. For the record, pressing Esc and typing ON will bring up a menu detailing the options available such as compiling and linking, all of which are accessed using the function keys.

When you first use either package you must make working copies of the disks. With Mix this is a simple matter, but those unfamiliar with MSDos may find installing Zorland a little less easy. However neither presented any great problems.

Unix compatibility is a phrase often mentioned in connection with C compilers, but what does that actually mean, and how important is it?

The idea of producing a compatible compiler is that Unix provides a standard to work to so that programs written to run under Unix or another compatible compiler can be used without alteration.

This ideal is not attainable under MSDos because a Unix C program can use system calls which of course will not work on your PC and file handling will not be quite the same.

However apart from these points a compiler can approach compatibility very closely, and both Zorland and Mix do so very well.

When buying third-party libraries the closer your compiler adheres to the Unix standard the more likely you are to be able to use them without problems.

C is a very small language with only about 30 reserved words and relies on libraries of functions to provide the facilities implemented as reserved words in most other languages. Both compilers come well equipped with libraries of standard functions, though Mix numbered 150 functions against approximately 100 for Zorland.

You can build up libraries of your own or buy them designed for specific applications. Unfortunately if you want to exploit the Amstrad's graphics potential you will have to do just that, as neither compiler comes with graphics functions.

Two-in-one package has much to offer

PETER WALKER
*looks at Amsoft's
database and word
processor system*

MICROFILE and MicroWord, written by Saxon Computing for Amsoft, together make up a system giving users of the PCW and the CPC6128 a database and an alternative word processor to LocoScript.

An integral mailmerge facility means that information stored on the database records, such as names and addresses, can be inserted into a document at predefined places. It is therefore a useful tool for anyone who needs to send out standard letters to addresses on mailing lists.

The database is flexible and capable of storing large amounts of information and carrying out simple calculations. You can therefore design a record-keeping system to meet your own requirements - a typical example is shown in Figure 1.

You will have plenty of room to manoeuvre because MicroFile, the

database part of the system, can deal with a maximum of 65,536 records depending on available disk space. Each record may contain up to a total of 1600 characters under 20 different headings (fields).

Even if you have only the one disk drive of the PCW8256 you will at least be able to use that to the full. The programs are loaded into the ram disk, so the program disk may be replaced with one for data storage.

Because information in a database may be confidential there are two levels of password restricting other users to specific files. This is a very important facility - the Data Protection Act insists that many computer users holding personal data on disk must protect it against unauthorized access.

Once through the password barrier you will need to define the database. The menu and screen instructions are helpful, giving access to many useful features. These include simple statistical functions such as standard deviation, and the mean value of all records can be calculated.

[ABRM] keys move the cursor around the screen - [f8] inserts a space
[DEL] deletes character behind the cursor - [CLR] deletes a character
[f2] deletes a line - [f3] inserts a line - [f4] clears the screen
[f5] restores the screen to its original form - [f6] toggles draw on/off

LABOUR... 2ND LINE 3RD LINE 4TH LINE 5TH LINE 6TH LINE 7TH LINE 8TH LINE 9TH LINE 10TH LINE

QUESTION ANSWER RECEIPT

Figure 1: A record-keeping system to meet your own requirements

It is also capable of doing simple calculations on each individual record. For example, if you select a constant such as the current VAT rate, you can then set up the system to accept a retail price and calculate the net sales value for each record as can be seen from Figure 2.

In addition to individual record calculations you can obtain totals for groups of them. Businesses could use these facilities in a variety of ways - for instance, to obtain a simple sales analysis.

Alternatively you may just wish to use the package to keep personnel records or to form the basis of a mailing list. In the latter case the records will have to be transferred to an external file for printing later. Similarly you can create an import file so that the information can be transferred into another MicroFile database.

This is one area where the menu is not very clear. And the manual does not seem to be particularly helpful, especially for beginners, although it at least guides you to set the cursor at the right part of the screen. There are some useful worked examples but exporting and importing information are not among them.

The manual tells you not to be afraid to experiment, so by trial and a lot of error I transferred my mailing list to an export file to be processed by MicroWord. Since that would be the main reason for using this part of MicroFile it seems illogical that the default response is "no" when you are asked if you want your export file to be used by MicroWord.

The export file can also be used for printing labels in one of two ways. The first is to send the data to the standard CP/M Plus file with the name LST: so that it will be

C COMPILER COMPARISON

If you need graphics I would suggest you investigate the cost of a library before you invest in C, because the only ones I have seen advertised are expensive.

Zorland is capable of producing EXE or COM files, whereas Mix only produces the latter. Zorland will recognize and use the 8087 maths co-processor if present and will also generate rom-able code if that's what you need.

One thing I liked about Mix was that you have the option to compile programs that load in the run-time routines they need when they are executed.

These routines are held in the RUNTIME.OVY file which must be present on the disk. The advantage is that you can have several COM files on the disk all using the single overlay file.

This means that multiple COM files will take up less disk space than if the run-time routines had been included in each file. As supplied, Zorland uses either a compact (total 64k program and data) or small (64k program plus 64k data) memory model.

A developer's upgrade is available at \$50 to allow you to use up to one megabyte each of program and data.

This aspect of the compiler's operation is not explained very well in the Mix manual, but it appears that only the small memory model is used, though Analytical Engines tells me that an upgrade is due to be released soon.

Zorland was the slower compiler of the two although both are fairly fast. Mix is supplied with utilities to optimize your program for speed or size, though the Zorland manual implies that these functions are part of the compilation process.

Zorland has a compiler option to generate an integer-only file which will be still smaller and faster.

I wrote a few simple programs based on the Personal Computer World benchmarks, to test the efficiency of the compiled code. They involved 1000 iterations of tasks such as writing to disk files, printing text on the screen, integer, floating point and trig/log calculations.

The absolute timings for each test are not important but the relative timings for code generated from the same source file are of interest.

As you can see from Table 1, the results are quite surprising in that Zorland produced the fastest code even after the Mix version had been optimized with the speedup utility supplied.

These benchmarks are by no means an exhaustive test of the code generated by either compiler under all possible circumstances, but I believe they will prove fairly representative.

I noticed that the floating point calculations were not

as accurate with the Mix compiled version. As they were the result of 1000 iterations this would probably be of major significance in scientific work.

So which one should you buy? Both packages offer outstanding value for money, but I think the beginner will find the Mix compiler and editor both cheaper and easier to work with.

The manual is good enough to substitute for a book, linker is provided and the ability to work from within the editor will make learning less frustrating. Analytical Engines says that a trace utility is under development to allow easy debugging of your programs and this will make Mix even more attractive to beginners.

The comparative cost of the Zorland compiler package and linker will be slightly more than the Mix package even if you decide not to invest in a better editor.

For the more experienced programmer the choice is so easy. A few will consider Zorland's ability to produce rom-able code or use a maths co-processor to be important. The resulting code is faster than Mix despite the speedup utility but on the other hand a Mix-compiled program can be made to take up far less disk space if more than one COM file is to be present.

Those experienced in MS-Dos will probably have a linker and a good text editor, so Zorland would prove less expensive. Its greater accuracy and speed make it the first choice for scientific applications or heavy number crunching.

Product: Zorland C
Price \$75

Product: Mix C Compiler and Editor
Price \$140

	Zorland	Mix		
		Normal	Speedup	Shrink
Integer arithmetic	0.225	0.533	0.243	0.56
Floating point	3.700	5.800	5.600	5.800
Trig/log	76.000	109.000	109.000	109.000
Text Printing	67.000	84.000	84.000	84.000
Disk filing	14.000	12.000	12.000	12.000
File sizes (bytes)				
Floating point	22493	Not including runtimes		
.COM file				
Floating point	23168			
.EXE file		5531	6563	5398
		Plus 22016 runtime		
		Including runtimes		
		24186	25128	24053

Table 1: Benchmark timings in seconds plus file lengths

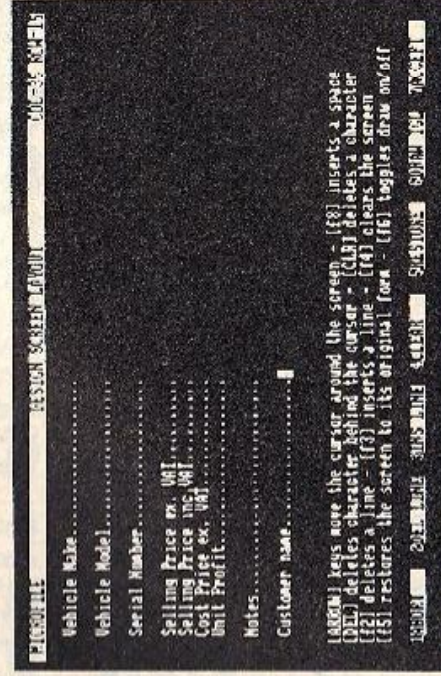


Figure 11: Calculations for each record

diverted to the printer. You have to take care that the number of data fields equals the number of lines for each label, so you will need to set up some dummy fields.

The other - easier - method is to use MicroWord. But before you can do anything with that you must obviously learn how to use it. The manual points out that if you are familiar with "the popular WordStar word-processing program" you will find many similarities.

Most of the commands are claimed to be WordStar compatible which means that it is very different from LocoScript on the PCW with it's system of pop-up menus.

For example, if in LocoScript you want to save a document on disk, you press Exit, guide the cursor the right option on the menu and press Enter. With MicroWord, you press the Alt and K keys followed by D, LS or X to Save, Save and Continue, or Exit. A menu at the top of the screen then gives you the usual prompt about the document name.

The Alt key and a combination of one or two characters is used for other editing functions such as setting right and left margins.

Other instructions rely on what are called dot commands. For example, if you type PA, you force a page break. These dot commands also give you the usual word-processing features such as setting page headers or footers.

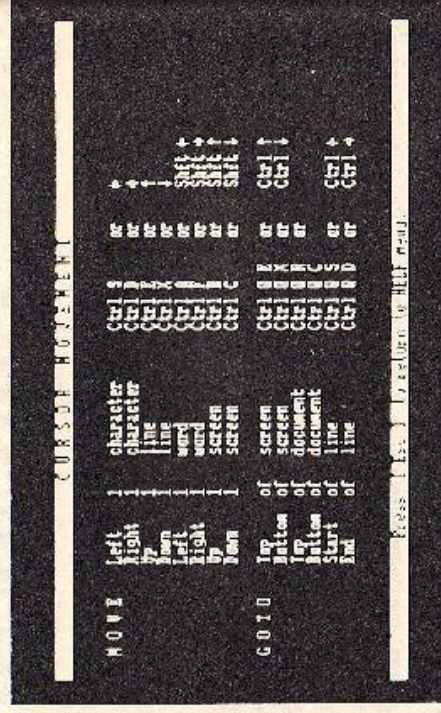


Figure III: Commands can take a long time to learn

I've found that such programs take a long time to learn and initially typing in documents is slow because the Help menu is used a lot. But once you have mastered the system - high input speed can be achieved - some people prefer this to a menu-driven system.

It is also very flexible, having many useful editing functions. You can move the cursor up or down a page or to the beginning or end of a document. Whole or part sentences can be deleted with just two keystrokes. (The range of cursor movements is shown in Figure III.)

Blocks of text can be moved, copied, deleted and read in from or written to disk, using the sequence Alt+K+ a letter. For instance, Alt+K+B means mark the beginning of a block.

All this is achieved more quickly than in LocoScript since cutting a block ready for pasting or saving, for example, is instantaneous.

The Search and Replace facility too is faster - and more flexible. In addition to options to search for whole words only, ignore case, replace without confirmation and find or replace in a whole file, there is also the option to search backwards from the cursor position.

Most important of all is the fact that no matter where the cursor is in a document the speed of a save is the same - unlike LocoScript, where a save executed with the

cursor at the top of a long list means you have time to take a nap!

All the facilities described above are to be found in WordStar. If MicroWord has one very useful feature not found on standard WordStar, or for that matter many other word processors - yet current document will be written a back-up file every ten minutes. Although this briefly interrupts your work, it solves the problem that can be caused by accidentally turning the machine off. In other words, you need no longer worry about Save and Continue commands.

Once a document has been saved, it may be printed. You will not be restricted to the Amstrad printer or its Epson equivalent because the system allows you to create your own print control file.

Microsoft's print menu lets you add features, such as right justification, which you may have omitted from the document. Users of the PC will have to study it closely before beginning with. Unlike LocoScript, the program assumes that you are using continuous stationary, so you have to be instructed to paginate between pages.

Also you do not have the attractive choice of LocoScript typefaces. Furthermore, if you are

CONTINUED ON
PAGE 78

LAST month we introduced the concept of the sound queue and saw how we could use just one channel parameter to ensure that the same note plays on all or any of the three sound channels available. We did this by adding together the channel parameters 1, 2 and 4 as necessary.

Also we saw how we could cause notes to wait for sounds on other channels by using the appropriate rendezvous factor - 8, 16, or 32.

Bearing all this in mind, Program I should cause you no difficulties:

```
10 REM PROGRAM I
20 SOUND 17,239,100,5
30 SOUND 1,190,100,5
40 FOR delay=1 TO 200:NEXT
   delay
50 SOUND 1,213,100,5
60 SOUND 1,239,100,5
70 SOUND 2,119,100,7
80 SOUND 2,127,100,7
90 SOUND 2,159,200,7
```

Program I

Here there are four SOUND commands producing one-second notes on channel A, and three giving notes on channel B. However, the tune doesn't start playing until after the delay loop of line 40 has finished, hence the slight pause before the "music".

The reason is that the SOUND command of line 20 has a channel parameter of 17. This pops a note on the channel A sound queue, but it won't start playing it as the parameter of 17 (1+16) tells the Amstrad to wait for a note on channel B.

However, it's not just any note - it has to be one with a parameter of 10 (2+8) which marks it as one waiting for a rendezvous with channel A.

This means that the note produced by line 20 has to wait until the program gets to the note produced by line 70 before the rendezvous is complete and both notes can play. In the meantime three more notes are put on the channel A queue and the program has worked its way round the delay loop of line 40.

Those with a long memory may

It takes two to rendezvous

recall that Program I is very much like the Program VIII of last month. The only difference is that I haven't bothered to rendezvous any of the notes after the first pair.

Program II is identical to the old Program VIII except that the delay loop has been increased so that there's a longer pause before the tune starts.

```
10 REM PROGRAM II
20 SOUND 17,239,100,5
30 SOUND 17,190,100,5
40 FOR delay=1 TO 1000:NEXT
   delay
50 SOUND 17,213,100,5
60 SOUND 1,239,100,5
70 SOUND 10,119,100,7
80 SOUND 10,127,100,7
90 SOUND 10,159,200,7
```

Program II

As you can hear, it works but there's no real need for all the 10s and 17s in the channel parameters. In this case, so long as the first two notes start together, the rest will be fine.

However if we introduce the delay loop at different points in the program we can get problems. Try deleting line 40 and adding:

```
75 FOR delay=1 TO 1000:NEXT
   delay
```

to Program I and see what happens. Now you have to rendezvous the next notes on each channel to overcome the hiccup.

Sometimes, when you've got a string of notes all over the program and you're not sure what delays may occur between them, it's better to rendezvous the lot. You may get some odd gaps, but it's better than the notes getting out of step.

But if you're going to rendezvous notes, remember that both notes have to have the fact flagged in

their channel parameters. Have a look at Program III.

```
10 REM PROGRAM III
20 SOUND 17,239,100,5
30 SOUND 17,190,100,5
40 FOR delay=1 TO 200:NEXT
   delay
50 SOUND 17,213,100,5
60 SOUND 1,239,100,5
70 SOUND 2,119,100,7
80 SOUND 2,127,100,7
90 SOUND 2,159,200,7
```

Program III

Here we've got seven SOUND commands, yet we only get three notes. Where are the other four?

The answer is that they're still hanging round in the channel. A sound queue waiting forlornly for a rendezvous with some gentle, well-meaning notes on channel B.

After popping these notes on the queue the program came to the last three SOUND commands and played them. These were the three notes we heard.

Of course we know that we meant them to rendezvous with the channel B notes, but we didn't tell the Amstrad that, so channel A remained mute.

The notes are still lurking there, waiting to be summoned. Try:

```
SOUND 10,119,100,7:
SOUND 10,127,100,7:
SOUND 10,159,100,7
```

and you'll hear them. It's a kind of magic. You enter three SOUND commands and you get seven notes. However it's not the kind of magic that you want in a tune.

Those phantom notes were taking up room on the channel A queue. If we'd tried to put any more on the queue it would have been full and the program would have ground to

a halt. If you don't believe me, try adding:

55 SOUND 1,100,100,7

to Program III and explain where the three notes went!

Incidentally, you might find it worth your while to set up the small Enter key with:

KEY 129, "SOUND 135,0,0,0" +
CHRS(13)

Now, when the sound channels get out of hand, just press the small Enter key and all the garbage will be cleared.

So far we've been using the SOUND command to make noises. There can be times when we want it to remain silent. Look at Program IV and you'll see what I mean.

The program uses channels B and C to produce a series of notes. However, the second and third notes on channel C are silent, corresponding to the "rests" in a piece of music. Let's take a closer look at the listing.

Line 20 and 30 produce one-second notes on channels B and C respectively. So do line 40 and 50, but take a look at line 50's SOUND command. This produces a one-second note on channel C but it's a strange note. Its pitch and volume parameters are both 0.

```
10 REM PROGRAM IV
20 SOUND 2,239,100,7
30 SOUND 4,119,100,5
40 SOUND 2,213,100,7
50 SOUND 4,0,100,0
60 SOUND 2,190,100,7
70 SOUND 4,0,100,0
80 SOUND 2,179,100,7
90 SOUND 4,239,100,5
```

Program IV

The result is one second of pure silence on channel C. The last pair of SOUND commands sound a final note on each channel.

As you use the Amstrad's sound facilities to produce more complex tunes, with two and three-part harmony, you'll find that there are lots of times when you want to produce silence. And you'll also find that the above method can have drawbacks. A much neater

way of ensuring that a channel stays silent until you want it to sound is shown by Program V:

```
10 REM PROGRAM V
20 SOUND 34,239,100,7
30 SOUND 30,119,100,5
40 SOUND 2,213,100,7
50 SOUND 2,190,100,7
60 SOUND 34,179,100,7
70 SOUND 20,239,100,5
```

Program V

This produces the same notes as the previous program but uses two fewer SOUND commands. The four commands that produce the notes on channel B are the same. What's happened is that we've dropped the silent dummy notes.

This is made possible by using the rendezvous facility. The channel parameters of lines 20 and 30 are arranged so that they rendezvous with each other. These two notes will start playing straight away while the channel B notes produced by lines 40 and 50 are popped on the queue.

Finally the SOUND commands of lines 60 and 70 are made to rendezvous. This means that the second note on channel C won't sound until the fourth note on channel B starts sounding.

In other words, channel C is silent while the second and third notes are played on channel B. We have our periods of silence without the need for dummy notes.

Strictly speaking, there's no need to rendezvous the first two notes of Program V - the program will work without it. However, I always like to rendezvous the notes that start a tune just in case. The last 2 notes have to be periods.

Program VI shows what happens if they're not:

```
10 REM PROGRAM VI
20 SOUND 2,239,100,7
30 SOUND 4,119,100,5
40 SOUND 2,213,100,7
50 SOUND 2,190,100,7
60 SOUND 2,179,100,7
70 SOUND 4,239,100,5
```

Program VI

The result is chaos!

One thing you may have noticed about using the rendezvous facility is that you can only make a wait until another note is reached. In other words, it takes to rendezvous.

If you're only using once channels and you want to stop a playing straight away, this causes problems. Of course you use delay loops or rendezvous dummy notes on other channels it's not very neat.

What you need is a way of putting a note on the sound queue telling it to wait there until it's different. What you want is the ability to "hold" a note.

You can get this by adding 64 to the channel parameter of the note that you want to be held on the queue. Suppose you want a second note on channel A but want it held on the queue until you need it. One way you could do it would be to use:

SOUND 17,200,100,7

and then invoke it with a dummy note on channel B such as:

SOUND 10,0,0,0

However, if there are already some notes in the channel B queue you've got problems.

The way to do it is to add 64 to the channel parameter of the note you want held. In this case 64 + 1 is so the SOUND command you want is:

SOUND 65,200,100,7

Try it and you'll hear nothing until the note has been held on the queue. To hear the note you've got to tell the Amstrad to let it out of custody. This is done with the aptly named RELEASE command followed by the appropriate channel parameter. So to hear the note we would use:

RELEASE 1

It's the same for channels B and C. To hold a note on them we add to the channel parameters (2 and 3 respectively) and release the note with:

RELEASE 2

and

RELEASE 4

as needed.

From our previous experiences of combining channel parameters you shouldn't be surprised to learn that:

SOUND 71,500,100,7

holds the same note on all three channels (1+2+4+64) while:

RELEASE 7

(1+2+4) releases them. Program VII shows how it works.

```
10 REM PROGRAM VII
20 FOR CHANNEL=1 TO 7
30 SOUND CHANNEL+64,200-
  CHANNEL*10,100,7
40 FOR DELAY=1 TO 1000:NEXT
  DELAY
50 RELEASE CHANNEL
60 PRINT "SOUND WITH
  CHANNEL PARAMETER
  "CHANNEL
70 NEXT CHANNEL
```

Program VII

Let's go back to the case of holding a note on just one channel for a moment. Once a note is held on a queue, any more notes for that channel get piled up behind it. For example:

SOUND 65,100,100,7;
SOUND 1,400,100,7

results in silence. The first note has a hold on it while the second is stuck behind it, waiting for its release. Try:

RELEASE 1

and you'll hear both notes. By releasing the first note you've freed the logjam.

However, if you've two held notes on a queue you've got to release them both.

If you've entered something like:

SOUND 65,600,100,7;
SOUND 65,300,100,7

then

RELEASE 1

only gives you the first note. The second moves up the channel A sound queue and sits there. It's held until you free it with

another:

RELEASE 1

Let's recap on what we've covered on channel parameters. We've seen that 1, 2 and 4 refer to channels A, B and C respectively. These can be combined as necessary to produce notes on more than one channel.

number	result
1	uses channel A
2	uses channel B
4	uses channel C
8	rendezvous with A
16	rendezvous with B
32	rendezvous with C
64	hold until RELEASEd

Table 1: Channel parameters

We've also seen that by adding 8, 16, and 32 to these channel parameters we can cause the notes to rendezvous with notes on channels A, B and C respectively. And on top of all this we've learned that adding 64 to the parameter holds that note.

It's a sort of mix and match. You decide what channel you want a note to sound on and what are the conditions that will cause it to play and add together the necessary parameters.

Try entering:

SOUND 97,200,100,7

The channel parameter of 97 means that the note is to be on channel A, it is to rendezvous with a note on channel C and also it is to be held. You can see this from the fact that $97=1+32+64$. Now give it a note on channel C for it to rendezvous with:

SOUND 12,500,100,7

The result is silence. The reason is that while we've given it the right note for a rendezvous ($12=4+8$) the channel A note is held until further notice.

The channel A note needs releasing while the channel C note needs to rendezvous with one on channel A. So both notes are held on queue. Put them out of their agony with:

RELEASE 1

which will cause both notes to sound.

Notice that the sound they make is exactly the same as that produced by:

SOUND 1,200,100,7;
SOUND 4,500,100,7

the result of taking away all the rendezvous and hold values in the channel parameters of the previous pair of notes.

I'll leave it to you to figure out why:

SOUND 97,800,100,7

followed by:

RELEASE 1

produces silence until you give it a note such as:

SOUND 12,400,100,7

Remember, it takes two to make a rendezvous, even if one has just been released from custody.

And that's where I'll stop for this month. We've covered a lot of ground, some of it fairly obscure at first glance. The hold and rendezvous facilities are one of those things that can be difficult in theory but become clear in practice. And practice is the key word.

Have a go at playing notes on all three channels using combinations of the channel parameters we've looked at so far. Table 1 gives a summary of them all.

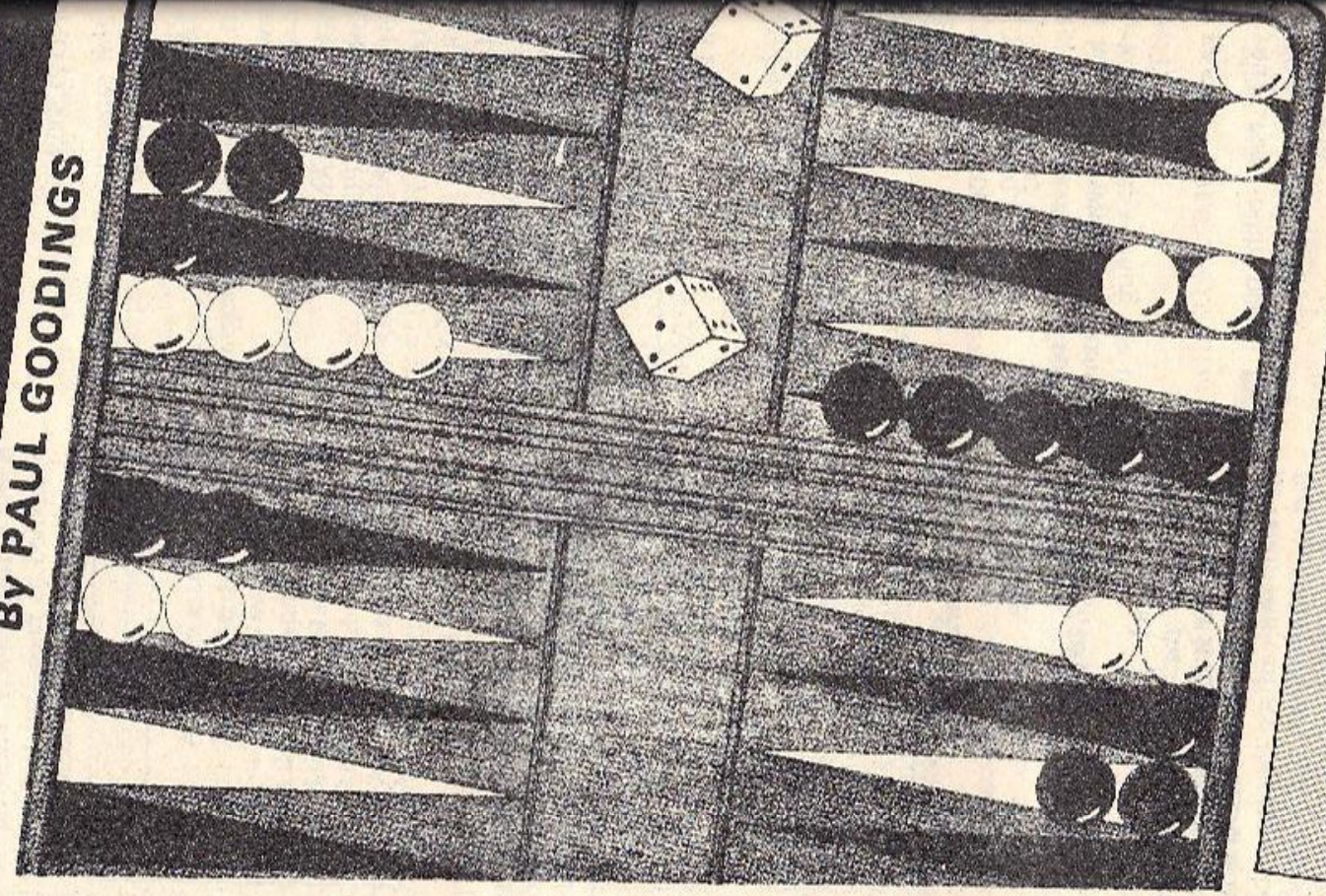
Make sure you understand what's happening, why you get the notes you do, and, more especially, why you don't get the notes you intended!

And, when you get completely stuck, don't forget that you've got the small Enter key set up as a panic button. Press this and you'll clear all the queues.

We'll see how that works next time.

Backgammon

By PAUL GOODINGS



THIS version of the traditional game for the Amstrad CPC series pits you against your micro.

It is played on a rectangular board which is divided into two halves, the Outer table (left) and Inner table (right). The tables are in turn divided horizontally so that each player has half the Outer and half the Inner table. Each table contains 12 points between which your 15 pieces move.

The object is to move all your pieces into your Inner table (the bottom righthand quarter of the board) and then off the board. Each player in turn throws the two dice and may move two pieces an equivalent number of points according to the value on the dice.

Alternatively a single piece may be moved according to the sum of the values of the dice. You have double moves when a doublet is thrown. The computer will move its pieces in a clockwise direction while you must move anti-clockwise.

Your piece may only land on a point which is either vacant, occupied by up to four of your own pieces or occupied by one of the computer's pieces - this is called a hit. Then the computer's piece goes to the middle of the table and the computer may not move again until the dice permit the movement of the hit piece to a vacant point (or a point occupied by one of your pieces) in your Inner table. The computer may then resume play. You may be hit by the micro in a similar fashion.

Only when all your pieces reach your Inner table may you remove them from the board as the dice allow.

If you remove all your pieces from the board first, you are the winner. If the computer is hit or still has a piece in your Inner table, you win a Backgammon. If the computer has not begun to remove its pieces from the board you win a Gammon or, if the computer has begun to remove its pieces, you win a hit.

A game may be saved to tape or disc at any stage and detailed instructions can be called during a game while preserving the pieces' positions.

MAIN VARIABLES

col (24,3)	Array holding coordinates of each point and number of pieces on each point.
order (24,2)	Array holding priority of each point in relation to the computer's next move.
hitc(), hitp()	Arrays holding coordinates of hit pieces.
on, pn	Numbers of pieces on board.
hc, hp	Numbers of hit pieces.
point	Point currently under examination.
score, pscore	Scores.
op, pp	Character codes of pieces.
from, toa	Origin point and destination point of piece moved.


```

10 REM Backgammon
20 REM by Paul Goodings
30 REM (c) Computing with the Amstr
ad
40 REM
50 CALL &BC02:cp=231:pp=230:sp=32:S
PEED WRITE 1
60 cscore=0:pscore=0:game=0:m1$="HI
T a key"
70 help=0:m2$=SPACE$(11)+"<< "+m1$+
">>":m3$="Press ENTER for Menu"
80 INK 0,0:INK 1,24:INK 2,1:INK 3,6
:INK 4,9:INK 5,8
90 DIM col(24,3),order(24,2),hic(24
,2),hip(24,2)
100 GOSUB 2960:BORDER 0:GOSUB 3050
110 IF a=2 THEN GOSUB 2740:GOSUB 25
50 ELSE pn=15:cn=15:hp=0:hc=0:GOSUB
3180:GOSUB 2740:GOSUB 2630
120 REM *****
130 REM ***** MAIN LOOP *****
140 WHILE pn>0 AND cn>0
150 PAPER 0
160 LOCATE 13,22:PRINT cn:LOCATE 13
,23:PRINT pn
170 IF toss=0 AND a<>2 THEN GOSUB 8
80 ELSE IF toss=0 THEN a=0:GOSUB 94
0
180 toss=0
190 IF help THEN GOSUB 3050:help=0:
a=2:GOTO 110
200 REM ***** COMP MOVE *****
210 IF pn THEN LOCATE 1,25:PRINT"I
shall throw my dice."SPC(15)
220 FOR f=1 TO 2000:NEXT
230 LOCATE 1,25:PRINT SPC(39)
240 GOSUB 2190:GOSUB 2250:REM throw
and print dice
250 IF double THEN m=4 ELSE m=2
260 ctt=m+1:ply=0
270 WHILE m>0 AND pn>0 AND cn>0
280 LOCATE 1,25:PRINT"My Move"ctt-m
290 IF (double=0 AND m=2) OR double
THEN rn=r(1) ELSE IF double=0 AND
m=1 THEN rn=r(2)
300 ceg=1
310 FOR x=7 TO 24
320 IF col(x,1) OR hc THEN ceg=0
330 NEXT
340 REM ***** sort moves *****
350 count=0:flag=0
360 IF ceg THEN GOSUB 1320:REM end
game
370 IF ceg AND flag=0 THEN 420 ELSE
IF flag THEN 540
380 IF hc THEN GOSUB 1280:GOTO 420:
REM hit move
390 FOR point=24 TO 1 STEP-1
400 IF col(point,1)>0 AND (point-rn

```

```

>>0 THEN GOSUB 2010:REM prioritise
410 NEXT
420 IF count=0 THEN 630
430 REM ***** find best moves *****
440 best=0:ct=0
450 FOR z=1 TO count
460 IF order(z,1)>best THEN best=or
der(z,1)
470 NEXT
480 FOR z=1 TO count
490 IF order(z,1)=best THEN ct=ct+1
:moves(ct)=order(z,2)
500 NEXT
510 IF ct=1 THEN point=moves(1) ELS
E point=moves(INT(RND*ct)+1)
520 REM ***** random choice of move
*****
530 ply=ply+rn:LOCATE 37,22:PRINT p
ly
540 IF point<25 THEN LOCATE 26,22:P
RINT CHR$(point+64) ELSE LOCATE 26,
22:PRINT"*"
550 FOR f=1 TO 1000:NEXT
560 IF hc=0 THEN GOSUB 1430:REM rem
ove piece
570 point=point-rn:PAPER 0
580 IF flag=0 THEN LOCATE 26,23:PRI
NT CHR$(point+64) ELSE LOCATE 26,23
:PRINT"*"
590 FOR f=1 TO 1000:NEXT
600 IF flag=0 THEN GOSUB 1530:REM a
dd piece
610 PAPER 0
620 IF hp THEN LOCATE 20,2:PRINT"Yo
u're HIT" ELSE LOCATE 20,2:PRINT SP
C(10)
630 m=m-1:WEND
640 WEND
650 REM *****
***
660 REM ***** finish *****
670 hci=0:hpi=0
680 FOR x=19 TO 24
690 IF col(x,1) THEN hci=1
700 NEXT
710 FOR x=1 TO 6
720 IF col(x,2) THEN hpi=1
730 NEXT:LOCATE 13,2
740 IF cn=0 AND (hp OR hpi)THEN csc
ore=cscore+3:PRINT"MY BACKGAMMON"
750 IF pn=0 AND (hc OR hci)THEN psc
ore=pscore+3:PRINT"YOUR BACKGAMMON"
760 IF hp=0 AND hpi=0 AND pn=15 THE
N cscore=cscore+2:PRINT"MY GAMMON"
770 IF hc=0 AND hci=0 AND cn=15 THE
N pscore=pscore+2:PRINT"YOUR GAMMON
"
780 IF cn=0 AND hp=0 AND pn<15 THEN
cscore=cscore+1:PRINT"MY WIN"
790 IF pn=0 AND hc=0 AND cn<15 THEN
pscore=pscore+1:PRINT"YOUR WIN"

```



```

800 FOR z=300 TO 1 STEP-21: SOUND 1,
300+z: SOUND 1, 300-z: SOUND 1, 119: NEX
T
810 PAPER 2: CLS: game=game+1
820 LOCATE 13,3: PRINT "Game: " game: LO
CATE 13,4: PRINT STRINGS$(7,"#")
830 LOCATE 5,10: PRINT "Your Score: "
pscore
840 LOCATE 5,15: PRINT " My Score: "
cscore
850 LOCATE 2,24: PRINT m1$ " for anot
her game....."
860 IF INKEY$="" THEN 860 ELSE GOTO
110
870 REM
880 REM **** PLAYER MOVE ****
890 REM
900 LOCATE 1,25: PRINT SPC(39): ply=0
910 LOCATE 1,25: PRINT m1$ " to throw
your dice."
920 IF INKEY$="" THEN 920 ELSE GOSU
B 2190
930 IF double THEN m=4 ELSE m=2
940 ctt=m+1: GOSUB 2250
950 WHILE m>0: LOCATE 37,22: PRINT p1
Y
960 peg=1
970 FOR z=1 TO 18
980 IF col(z,2) OR hp THEN peg=0
990 NEXT
1000 LOCATE 1,25: PRINT "POINT & MOVE
?" ctt-m; INPUT "Z for Help! ", pm$
1010 pm$=UPPER$(pm$)
1020 IF pm$="" THEN 1220
1030 IF pm$="Z" THEN help=1: RETURN
1040 from=ASC(LEFT$(pm$,1))-64: v=VA
L(MID$(pm$,2))
1050 IF hp THEN too=v ELSE too=from
+ v
1060 REM **** illegal moves ****
1070 IF peg=0 AND (from<1 OR from>2
4 OR too>24) THEN 1000
1080 IF too<25 THEN IF col(too,1)>1
OR col(too,2)=5 THEN 1000
1090 IF hp=0 AND (col(from,1) OR co
l(from,2)=0) THEN 1000
1100 IF NOT(v=r(1) OR v=r(2)) THEN
1000: REM original IF NOT(v=r(1) OR
v=r(2) OR v=r(1)+r(2)) THEN 1040
1110 IF peg AND too>24 AND col(25-v
,2)=0 THEN GOSUB 1240: IF flag THEN
1000
1120 IF hp THEN IF (col(too,1)>1 OR
col(too,2)) THEN 1000
1130 REM **** execute move ****
1140 ply=ply+v: LOCATE 37,22: PRINT p
ly
1150 IF hp=0 THEN LOCATE 26,22: PRIN
T CHR$(from+64) ELSE LOCATE 26,22: P
RINT "*"
1160 IF too<25 THEN LOCATE 26,23: PR
INT "*"
1170 IF hp=0 THEN point=from: GOS
1720
1180 FOR f=1 TO 1000: NEXT
1190 IF too<25 THEN point=too: GO
1820 ELSE pn=pn-1
1200 PAPER 0: IF pn=0 THEN RETURN
1210 IF hc THEN LOCATE 20,2: PRIN
"m HIT " ELSE LOCATE 20,2: PRIN
PC(10)
1220 m=m-1: WEND
1230 RETURN
1240 REM **** player end game **
1250 flag=0: FOR x=19 TO from-1
1260 IF col(x,2) THEN flag=1
1270 NEXT: RETURN
1280 REM **** comp hit move ****
1290 IF col(25-rn,1) OR col(25-rn
)>1 THEN RETURN
1300 count=count+1: order(count,1)
9: order(count,2)=25
1310 RETURN
1320 REM **** comp end game ****
1330 IF col(rn,1) THEN point=rn: f
lag=1
1340 z=6: WHILE z AND flag=0
1350 IF flag=0 AND col(z,1) AND r
z THEN point=z: flag=1
1360 IF flag=0 AND col(z,1) THEN
1370 z=z-1: WEND
1380 IF flag THEN cn=cn-1: RETURN
1390 FOR point=6 TO 1 STEP-1
1400 IF col(point,1) AND (point-r
)>0 THEN GOSUB 2010
1410 NEXT
1420 RETURN
1430 REM **** remove comp piece *
*
1440 ofs=col(point,1)
1450 IF point>12 THEN yc=19: ofs=-
s ELSE yc=5
1460 PAPER 2+(point MOD 2)
1470 FOR x=1 TO 20
1480 LOCATE col(point,3),yc+ofs: PR
INT CHR$(cp)
1490 LOCATE col(point,3),yc+ofs: PR
INT CHR$(sp)
1500 NEXT: PRINT CHR$(7);
1510 col(point,1)=col(point,1)-1
1520 RETURN
1530 REM **** add comp piece ****
1540 PAPER 0: PEN 1
1550 IF hc THEN GOSUB 1990: LOCATE
ic(hc,1),hic(hc,2): PRINT " :hc=hc-
1560 ofs=col(point,1)+1
1570 col(point,1)=col(point,1)+1
1580 IF point>12 THEN yc=19: ofs=-o
s: j=13 ELSE yc=5: j=11
1590 IF col(point,2)=1 THEN GOSUB

```



```

660:REM hit
1600 PAPER 2+(point MOD 2)
1610 FOR x=1 TO 5
1620 LOCATE col(point,3),yc+ofs:PRI
NT CHR$(sp)
1630 LOCATE col(point,3),yc+ofs:PRI
NT CHR$(cp)
1640 NEXT
1650 RETURN
1660 REM ** hit **
1670 hp=hp+1:hip(hp,1)=col(point,3)
:hip(hp,2)=j
1680 LOCATE hip(hp,1),hip(hp,2):PRI
NT CHR$(pp)
1690 col(point,2)=col(point,2)-1
1700 SOUND 1,63:FOR x=1 TO 250:NEXT
:SOUND 1,478
1710 RETURN
1720 REM **** remove player piece *
***
1730 ofs=col(point,2)
1740 IF point>12 THEN yc=19:ofs=-of
S ELSE yc=5
1750 PAPER 2+(point MOD 2)
1760 FOR x=1 TO 20
1770 LOCATE col(point,3),yc+ofs:PRI
NT CHR$(pp)
1780 LOCATE col(point,3),yc+ofs:PRI
NT CHR$(sp)
1790 NEXT:PRINT CHR$(7);
1800 col(point,2)=col(point,2)-1
1810 RETURN
1820 REM **** add player piece ***
*
1830 PAPER 0:PEN 1
1840 IF hp THEN SOUND 1,478:FOR x=1
TO 250:NEXT:SOUND 1,63:LOCATE hip(
hp,1),hip(hp,2):PRINT " :hp=hp-1
1850 ofs=col(point,2)+1
1860 col(point,2)=col(point,2)+1
1870 IF point>12 THEN yc=19:ofs=-of
S:j=13 ELSE yc=5:j=11
1880 IF col(point,1)=1 THEN GOSUB 1
950:REM hit
1890 PAPER 2+(point MOD 2)
1900 FOR x=1 TO 5
1910 LOCATE col(point,3),yc+ofs:PRI
NT CHR$(sp)
1920 LOCATE col(point,3),yc+ofs:PRI
NT CHR$(pp)
1930 NEXT
1940 RETURN
1950 REM ** hit **
1960 hc=hc+1:hic(hc,1)=col(point,3)
:hic(hc,2)=j
1970 LOCATE hic(hc,1),hic(hc,2):PRI
NT CHR$(cp)
1980 col(point,1)=col(point,1)-1
1990 SOUND 1,63:FOR x=1 TO 250:NEXT
:SOUND 1,478

```

```

2000 RETURN
2010 REM **** move priorities ****
2020 too=point-rn:pf=col(point,1):p
tc=col(too,1):ptp=col(too,2)
2030 IF ptc=5 OR ptp>1 THEN RETURN
2040 count=count+1:pri=0
2050 IF pf=1 AND ptc=1 AND ptp=0 TH
EN pri=20
2060 IF pf=1 AND ptc=0 AND ptp=1 TH
EN pri=19
2070 IF pf=1 AND ptc>1 AND ptp=0 TH
EN pri=18
2080 IF pf>2 AND ptc=1 AND ptp=0 TH
EN pri=17
2090 IF pf>2 AND ptc=0 AND ptp=1 TH
EN pri=16
2100 IF pf=1 AND ptc=0 AND ptp=0 TH
EN pri=15
2110 IF pf>2 AND ptc>1 AND ptp=0 TH
EN pri=14
2120 IF pf>2 AND ptc=0 AND ptp=0 TH
EN pri=13
2130 IF pf=2 AND ptc=1 AND ptp=0 TH
EN pri=12
2140 IF pf=2 AND ptc=0 AND ptp=1 TH
EN pri=11
2150 IF pf=2 AND ptc>1 AND ptp=0 TH
EN pri=10
2160 IF pf=2 AND ptc=0 AND ptp=0 TH
EN pri=9
2170 order(count,1)=pri:order(count
,2)=point
2180 RETURN
2190 REM **** throw dice ****
2200 RANDOMIZE TIME
2210 FOR z=1 TO 2
2220 r(z)=INT(RND*6)+1:NEXT
2230 IF r(1)=r(2) THEN double=1 ELS
E double=0
2240 RETURN
2250 REM **** print dice ****
2260 PAPER 0:LOCATE 33,2:PRINT SPC(
6):PAPER 3
2270 LOCATE 1,1:PRINT"Die A">r(1):L
OCATE 1,2:PRINT"Die B">r(2)
2280 IF double THEN PAPER 3:LOCATE
33,2:PRINT"DOUBLE"
2290 PAPER 0:RETURN
2300 REM **** save game ****
2310 CLS:PEN 1:PRINT"SAVE A GAME ON
TAPE":LOCATE 9,25:PRINT m3$
2320 LOCATE 1,10:PEN 3:INPUT"What i
s the name of the game";names$
2330 names$=UPPER$(names$):IF names$="
" THEN RETURN ELSE OPENOUT names$
2340 PRINT#9,m,ply,hc,hp,cn,pn,r(1)
,r(2)
2350 FOR x=1 TO hc:FOR y=1 TO 2
2360 PRINT#9,hic(x,y):NEXT y,x
2370 FOR x=1 TO hp:FOR y=1 TO 2
2380 PRINT#9,hip(x,y):NEXT y,x

```



```

2390 FOR x=1 TO 24:FOR y=1 TO 3
2400 PRINT#9,col(x,y):NEXT y,x
2410 CLOSEOUT:RETURN
2420 REM *** load game ***
2430 CLS:PEN 1:PRINT"LOAD A GAME FROM TAPE":LOCATE 9,25:PRINT m3$
2440 LOCATE 1,10:PEN 3:INPUT"What is the name of the game";name$
2450 name$=UPPER$(name$):IF name$=" " THEN a=0:RETURN ELSE OPENIN names
2460 INPUT#9,m,ply,hc,hp,cn,pn,r(1),r(2)
2470 FOR x=1 TO hc:FOR y=1 TO 2
2480 INPUT#9,hic(x,y):NEXT y,x
2490 FOR x=1 TO hp:FOR y=1 TO 2
2500 INPUT#9,hip(x,y):NEXT y,x
2510 FOR x=1 TO 24:FOR y=1 TO 3
2520 INPUT#9,col(x,y):NEXT y,x
2530 CLOSEIN:toss=0:IF r(1)=r(2) THEN double=1 ELSE double=0
2540 RETURN
2550 REM ** reprint game **
2560 FOR point=1 TO 24
2570 IF col(point,1) THEN FOR y=1 TO 3
2580 IF col(point,2) THEN FOR y=1 TO 3
2590 NEXT:PAPER 0:PEN 1
2600 FOR y=1 TO hc:LOCATE hic(y,1),hic(y,2):PRINT CHR$(cp):NEXT
2610 FOR y=1 TO hp:LOCATE hip(y,1),hip(y,2):PRINT CHR$(pp):NEXT
2620 RETURN
2630 REM *** read coordinates & print initial placements ***
2640 RESTORE 2700
2650 FOR point=1 TO 24
2660 READ col(point,1):READ col(point,2):READ col(point,3)
2670 FOR z=1 TO col(point,1):ofs=z:GOSUB 1580:NEXT
2680 FOR z=1 TO col(point,2):ofs=z:GOSUB 1870:NEXT
2690 NEXT:RETURN
2700 DATA 0,2,37,0,0,34,0,0,31,0,0,28,0,0,25,5,0,22
2710 DATA 0,0,18,3,0,15,0,0,12,0,0,9,0,0,6,0,5,3
2720 DATA 5,0,3,0,0,6,0,0,9,0,0,12,0,3,15,0,0,18
2730 DATA 0,5,22,0,0,25,0,0,28,0,0,31,0,0,34,2,0,37
2740 REM *** draw board ***
2750 BORDER 0:PAPER 0:PEN 1:CLS:PAPER 2

```

```

2760 LOCATE 20,22:PRINT"From:":LOCATE 20,23:PRINT" To:"
2770 LOCATE 1,22:PRINT" My Pieces:"
2780 LOCATE 1,23:PRINT"Your Pieces:"
2790 LOCATE 30,22:PRINT"Played:"
2800 PAPER 2:LOCATE 1,5:PRINT SPC(9):LOCATE 1,19:PRINT SPC(39)
2810 PAPER 3:LOCATE 5,5:PRINT"outer table":LOCATE 24,5:PRINT"inner table":PEN 1
2820 FOR y=6 TO 18:LOCATE 1,y:PRINT" ":LOCATE 20,y:PRINT" ":LOCATE 30,y:PRINT" ":NEXT z=0
2830 FOR x=3 TO 18 STEP 3:PAPER 0
2840 LOCATE x,4:PRINT CHR$(76-z)
2850 LOCATE x,20:PRINT CHR$(77+z)
2860 PAPER 3-(x MOD 2):FOR y=6 TO 18:LOCATE x,y:PRINT" ":NEXT
2870 PAPER 2+(x MOD 2):FOR y=14 TO 18:LOCATE x,y:PRINT" ":NEXT
2880 z=z+1:NEXT z=0
2890 FOR x=22 TO 37 STEP 3:PAPER 0
2900 LOCATE x,4:PRINT CHR$(70-z)
2910 LOCATE x,20:PRINT CHR$(83+z)
2920 PAPER 2+(x MOD 2):FOR y=6 TO 18:LOCATE x,y:PRINT" ":NEXT
2930 PAPER 3-(x MOD 2):FOR y=14 TO 18:LOCATE x,y:PRINT" ":NEXT
2940 z=z+1:NEXT
2950 RETURN
2960 REM ***** introduction ****
2980 MODE 0:PEN 5:PAPER 0:BORDER 0
2990 LOCATE 2,1:PRINT STRING$(18," "):LOCATE 2,25:PRINT STRING$(18," ")
3000 FOR x=2 TO 24:LOCATE 1,x:PRINT" ":LOCATE 20,x:PRINT"*":NEXT
3010 PAPER 1:LOCATE 3,10:PRINT"> Backgammon <"
3020 PAPER 3:LOCATE 3,20:PRINT"by 1 Goodings"
3030 FOR x=1 TO 5000:NEXT
3040 PAPER 0:RETURN
3050 REM ***** menu *****
3060 WHILE-1:MODE 1:PEN 1:a=0:LOCATE 14,1:PRINT"Backgammon":PEN 3
3070 LOCATE 14,2:PRINT STRING$(10,"+"):PEN 1:LOCATE 17,4:PRINT"MENU"
3080 LOCATE 1,8:PRINT"1. Instructions."
3090 LOCATE 1,10:PRINT"2. Load and save game from tape."
3100 LOCATE 1,12:PRINT"3. Save and load game on tape."
3110 LOCATE 1,14:PRINT"4. Play Backgammon."

```



```

3120 PEN 3:LOCATE 1,25:PRINT"Choose
...
3130 a=VAL(INKEY$):IF a=0 THEN 3130
3140 IF a=4 THEN RETURN
3150 ON a GOSUB 3290,2420,2300
3160 IF a=2 THEN RETURN
3170 WEND
3180 REM ***** who moves first ****
**
3190 CLS:PEN 3
3200 LOCATE 4,8:PRINT"Throw the DIC
E to decide who"
3210 LOCATE 1,9:PRINT"makes the fir
st move.Highest DIE wins."
3220 LOCATE 1,11:PRINT SPC(8)"[You,
re DIE:":PEN 1:PRINT"B,":PEN 3:P
RINT"]
3230 PEN 1:LOCATE 1,25:PRINT m1$
3240 IF INKEY$="" THEN 3240 ELSE GO
SUB 2190:GOSUB 2250
3250 IF double THEN LOCATE 1,20:PRI
NT"A tie! Try again...":GOTO 3240
3260 PAPER 2
3270 IF r(1)>r(2) THEN toss=1:LOCAT
E 1,25:PRINT"I Win!!!" ELSE toss=
0:LOCATE 1,25:PRINT"You Win.."
3280 FOR f=1 TO 3000:NEXT:RETURN
3290 REM ***** instructions ****
3300 MODE 1:PAPER 0:PEN 1
3310 LOCATE 13,1:PRINT"Instructions
":LOCATE 13,2:PRINT STRING$(12,"*")
3320 LOCATE 1,4:PRINT"1.Based on th
e classic Board Game you compete
with your computer.You may need to
call these instructions after you
begin - just enter 'Z,'."
3330 LOCATE 1,9:PRINT"2.The board i
s divided into two TABLES: Left-OUT
ER/Right-INNER. Each TABLE is intur
n divided horizontally so that each
PLAYER has half the INNER & OUTER
TABLE.[You have the BOTTOM HALF OF
THE BOARD]"
3340 LOCATE 1,15:PRINT"3.Each TABLE
has 12 POINTS between which your 15
pieces move towards your INNER TA
BLE(bottom right).Two dice are thro
wn.One or two pieces may be moved a
number of POINTS equivalent to the
ir value."
3350 PRINT"e.g If 6 thrown enter A6
to move piece from A to G"
3360 LOCATE 1,23:PRINT"4.You may on
ly land on a POINT if it is:"
3370 LOCATE 1,25:PRINT m2$:IF INKEY
$="" THEN 3370 ELSE CLS
3380 LOCATE 1,1:PRINT"a) Unoccupied
"
```

```

3390 LOCATE 1,3:PRINT"b) Occupied b
y up to 4 of your pieces."
3400 LOCATE 1,5:PRINT"c) Occupied b
y 1 of my pieces:I am then HIT & ma
y not move until I throw a die wit
h a value equal to a POINT in your"
3410 LOCATE 1,8:PRINT"INNER table(C
ount from right of board) which is
vacant or occupied by 1 of yourpie
ces."
3420 LOCATE 1,11:PRINT"           e.g I
f I throw a 2 I may move to POINT W
(if it is vacant or holds one ofyo
ur pieces)"
3430 LOCATE 1,16:PRINT"4.When all y
our pieces reach your INNER TABLE t
hey may be removed from the tableas
follows:"
3440 LOCATE 1,20:PRINT"a) If you th
row 5 & you have a piece on POINT T
,ENTER T5 & it will be removed."
3450 LOCATE 1,25:PRINT m2$:IF INKEY
$="" THEN 3450 ELSE CLS
3460 LOCATE 1,1:PRINT"b) If you thr
ow 3 but do not occupy"SPC(5)"POINT
V (or any POINT to left of V) you
may remove a piece from the POINT y
ou occupy which is nearest to & t
o the"SPC(5)"right of V. e.g if W,E
ENTER W3."
3470 LOCATE 1,8:PRINT"5.When both d
ice thrown have the same value(a
DOUBLE) the throw may be played twi
ce e.g. if 2's thrown 4 pieces may
be removed 2 POINTS or 2 pieces 4 POIN
TS etc."
3480 LOCATE 1,15:PRINT"6.If you can
not move HIT ENTER."
3490 LOCATE 1,25:PRINT m2$:IF INKEY
$="" THEN 3490 ELSE CLS
3500 LOCATE 1,3:PRINT"7.If you remo
ve all your pieces from theboard fi
rst you win & score:"
3510 LOCATE 1,6:PRINT"a) BACKGAMMON
[3 points] if I still havea piece
in your INNER TABLE or I am HIT."
3520 LOCATE 1,11:PRINT"b) GAMMOM [2
points] if I still have allmy piec
es on the board."
3530 LOCATE 1,15:PRINT"c) WIN [1 po
int] if I have started to remove
my pieces."
3540 PEN 3:LOCATE 1,20:PRINT"PIECES
: Mine "CHR$(231)" Yours "CHR$(23
0)
3550 PEN 1:LOCATE 1,25:PRINT m2$:IF
INKEY$="" THEN 3550
3560 RETURN
3570 REM ***** END *****
```


ORGANIZE YOUR DISKS WITH THESE PUBLIC DOMAIN TREATS!

Those of you who peruse Bulletin Boards may often be puzzled by the presence of files with the extension .LBR. These files are actually collections of files that are related in some way and grouped together under a meaningful name. The reasons for this are many and varied, but with the AMSTRAD's lack of disk space these .LBR's (or library files) make good sense. To understand the benefit of library files you must first understand how CP/M stores information on the disks.

Each disk is made up of concentric magnetic rings that are known as tracks. The AMSTRAD has 40 tracks per side on its 3 inch disks. Each track is divided into sectors. On the AMSTRAD, each sector is 512 bytes (or 1/2 k long). There are nine of these per track. Some elementary arithmetic shows:

$$40 * 9 = 360 \text{ sectors of 512 bytes per disk}$$

$$360 / 2 = 180 \text{ k bytes per disk side}$$

of that 180 k, two tracks or 9 k is reserved (on a system disk) for what are known as the system tracks which is where the setup parameters are stored, along with various other things. Now, $180 \text{ k} - 9 \text{ k} = 171 \text{ k}$, so why can we only store 169k on a system disk? The answer is because CP/M reserves a further 2k for the directory. The directory is the part of the disk that tells CP/M where on the disk the files are stored and also some status information about the file. Each directory entry is 32 bytes long. Since the directory reserved space is 2k or 2048 bytes with some more elementary arithmetic we can see that $2048 / 32 = 64$ which is why you have only 64 directory entries before getting that nasty simplistic explanation of how CP/M stores information in its directory, but it serves to illustrate my point. If you want more information on the 'guts' of CP/M I recommend 'Inside CP/M' by David Cortesi or 'CP/M User Guide' by Tom Hogan. On with the confusion. Each file (on standard AMSTRAD disks) takes up at least 1k bytes. Why? The short answer is that CP/M handles files in blocks and a block on the AMSTRAD is 1k long. This means that a file even if it is only 20 bytes long takes up at least 1k. Pretty wasteful isn't it? Well, if CP/M didn't work that way it could take forever to load a file into memory!

Don't worry, I haven't lost sight of why we started this discussion. Now we come to the point. What this means to us is that any file that is not an exact multiple of 1k bytes will have wasted disk space at the end of the file. Common sense tells us that this will be on average 1/2 k. So you can see that if you have 64

files on a disk and an average of one half of 1k not in 64 of these files then you have some 32k of unused disk space (if my math is right!). To reclaim majority of this space, you use a library file. Gather together related files and put them into a library under an appropriate name. You may delete the files you have just placed in the library and so free directories for further use and also reclaim the majority of the wasted disk space.

Now, there is a penalty to pay for this. These files are no longer accessible from the command line and if they are used by other files they cannot be used without being extracted from the library. This, in actual fact, is not as bad as it seems because the majority of files consigned to libraries are files that are not used often. The exception to this is the default standard library that is used for all those commands that usually run on the system disk. This library is called COMMAND.LBR and should contain all those commonly used commands like PIP, STAT etc. This means that directory display is not cluttered with their names. Using the program LRUN <command> you can access these files as if they were stand alone.

Ok, that's the good news and the bad news on libraries. One final point: - NEVER CONSIGN YOUR OWN COPY OF ANY FILE TO A LIBRARY. This is a variation on the theme always keep backups. It's sound advice.

Continuing on our theme of organizing the computer users out there we also have for you a cataloging system that I have been using myself for about 18 months. This is the original version of this type utility and was written by Ward Christensen, a name you have probably seen before in this column. It consists of three files: - CAT, FMAP and UCAT. To use this programs you must first give your disks a name that begins with a minus sign (-) and has an extension of three alphanumeric characters. In other words, put a file name with a dash as it's first character on the disk and with a 0k file length. To put a 0k length file on a disk from the A> prompt (UNDER CPM 2.2) type:-

A>SAVE0-MYDISK.ANY

This now serves as the disk name. To actually catalog the disk the FMAP program is used. On a single drive system (on a 464) this involves copying FMAP onto the disk to be cataloged, and ensuring there is enough room for the file that FMAP creates, typing FMAP A: F

Once the file NAMES.SUB is created, copy it onto your catalog disk and then run UCAT to update your catalog. For those of you with single drive 664/6128/8000 AMSTRADS use the command line A>FMAP B: F and you will get the usual Drive B message. See CATALOG.DOC for a greater explanation of using the cataloging program.

If you are religious about cataloging your files you will find the advantages are that you can find any file very quickly and not spend hours looking through each of your disks for that elusive file. Before you go cataloging, read the DOC file first. Same goes for LUDOC!!

GO.COM is a great idea! How many times have you accidentally left an application program to find yourself back at the A> prompt? I have, many times. Just type GO and press return. If your program does not overwrite the CCP you will be back into it with no fuss.

Here is a list showing which of the files work with which CP/M

FILE	2.2	3.0	NOTE
CAT.COM	Y	Y	EXTRACTS FROM .LBR FILES ALL MEMBER FILES
CATALOG.DOC	Y	Y	
FMAP.COM	Y	Y	
UCAT.COM	Y	Y	
LU.COM	Y	Y	
LUDOC	Y	Y	ORIGINAL FILE IN STATUS.ORG
LRUN.COM	Y	Y	
LDIR.COM	Y	Y	
DELBR.COM	Y	Y	GIVES INCORRECT NUMBER OF K
FINDU12.COM	Y	Y	
FINDU12.DOC	Y	Y	
STATUS.COM	Y	Y	
GO.COM	Y	N	
D.COM	Y	Y	
FREE	Y	Y	
ERAQ.COM	Y	Y	
UNERA.COM	Y	N	

There is a great program for cataloging disks in the Public Domain called FATCAT. I would have liked to present it here for you but after investigation the minimum configuration that it runs on is two floppy drives of at least 180k plus a TPA of at least 42k which would preclude all 464 owners. That is the reason why I tend to stick to early versions of PD software as they were usually written for minimum memory systems and single drives.

OK, thats all folks for this time. I can't tell you what is on for next time, as every time I do that, a source file fails to turn up, or it just isn't good enough to warrant selection. So you'll be surprised as I am. Till next time.

SHANE KELLY

STOP PRESS!!!

I have just received a phone call regarding PD DISK number 3. The caller could not get DASM assembled using Z80MR. On a single drive system, this is how you go about it.

- 1) On a system disk copy DASM, DASM1, DASM2, DASM3, DASM4, & DASM5 and also Z80MR.
- 2) Invoke Z80MR on dasm by typing this:
A>Z80MR DASM.AAZ
- 3) In about 3 minutes you should have a copy of DASM.HEX on drive A: Now, use the CP/M utility LOAD.COM to produce a .COM file from the .HEX by typing this:

A>LOAD DASM (FIRST COPY DASM.HEX TO A DISK WITH LOAD.COM ON IT.)

And thats all there is to it, but to be fair I did forget to explain that invoking Z80MR is done in the exact same way as the assembler supplied with your system, ASM. That is, the extension of the target file (.AAZ of DASM above) specifies the following to Z80MR:- the first position is the drive letter of the drive containing the source files. The second letter of the extension specifies which disk drive shall receive the output file (DASM.HEX in the above) while the third letter says where the listing file should go. As you will know, Z is not a legal CP/M disk drive, and a Z in this position tells Z80MR to skip producing a listing file. If we had placed an X in this position the listing file would have been sent to the console. A few examples might help:-

A>Z80MR DASM.BAX - This would look for source file on B:, place the .HEX file on A: and put the .PRN file to the console.

A>B:Z80MR DASM.BZZ - this would look for Z80MR on B:, which would look for the source file on B: and skip producing both the hex file and the .PRN file. This is most useful for doing a check assembly as it is the quickest way of finding the number of errors in a program.

Apologies to all who were inconvenienced by my omission. By far the quickest way to contact me is to mailbox me on VIATEL 534876510. Failing that, a letter to the mag is slow but sure. Thanks to ANDREW van der STOCK for alerting me to this omission.

**Full details of how to
order Shane Kelly's
Public Domain disks are
on pages 72 & 73**

Tasword in combination with your old textfiles because of the awkward effects arising from corrupted left and right values.

Furthermore the first two characters will have disappeared from the file.

Also, there is a simple method for making a hard copy of the Command Reference Guide in Hisoft Pascal 4D's ED80.

Everything will fit nicely on one page and you need no more than type Ctrl+J to obtain a list of all possible commands, thereby obscuring the actual source text you are editing.

Type the following after the CP/M prompt:

```
A> ed80 ed80.hlp
```

Delete the two messages "Space for more, [ENTER] to quit" and the empty lines above it. Put a title at the beginning and finally press Ctrl + O + Q.

Replace in the right upper corner the filename ED80.HLP by LST: - note the colon - and press Enter to start printing.

For anyone with the Amsoft Devpac 464 here's a solution for the exacting Z80 code programmer faced with the problem of using two - ORG - assembler directives in one assembly text and saving the two sections of object code.

Why not split the assembly into two independent sections?

Doing so disables the cross-referencing at labels set up in the other part. For example:

```
ORG 20000      ;part 1
LD HL, omega
show: LD A, (HL)
      JP #BB5A
alpha: DEFB "A"
;
ORG 30000      ;part 2
LD HL, alpha
JP show
omega: DEFB "Z"
```

Using O,,s saves only the machine

code of the second part, even when option 16 is chosen. Fear not, there is a way out.

Save the second part as usual - O,,PARTTWO.

For the first part when option 16 is used. Note the address of the end of the assembly text and use the X-command. [A].

Note also the symbol table length [B] and deduct from the assembler list the length of the object code of part one. [C]. Go to Basic and enter the command:

```
SAVE "PARTONE", b, [A] + [B] +
2, [C]
```

For the first part when option O is used it is the same as the second step of the solution, but replace the start address [A]+[B]+2 by the ORG value of the first part, or, as in this example, 20000. - Patrick De Geest

RECIPE FOR TREES

I have tried out your various attempts to construct binary trees using Basic. Here is my own using DR Logo.

Just load Logo, type in the two procedures listed, then type:

```
tree n
```

with n between 30 and 50 for best results. A quite realistic symmetrical tree will be drawn in random colors.

The procedure tree1 is recursive - it calls itself - and is far more elegant than anything Basic can manage. Change any of the angles or the scale factor - .75 - to produce different trees. - A.M. Scott

```
to tree: size
ht
f5
c5
setpc 1 + random 3
pu
bk 60
pd
tree1: size
end
```

```
to tree1: size
if (:size < 2) [stop]
fd :size
rt 15
tree1 (:size * 0.75)
lt 30
tree1 (:size * 0.75)
rt 15
bk :size
end
```

CYCLE ON SHOW

A good way to see the effect of the repeating cycle of the pseudo random generator can be illustrated as follows:

```
MODE 2: WHILE TIME: PLOT
RND*640,RND*&400: WEND
```

To fill the whole screen in Mode 2 all the 640 x 200 - or 128000 - pixels have to be plotted. But because the random generator delivers much less numbers than this total the plotting will come prematurely to an end. I timed 5 min. 18 sec.

All of a sudden the sparkling extincts as if a hangup occurred. In fact the computer is replotting every pixel already plotted before.

The sequence repeats itself. Insert in the loop the instruction RANDOMIZE TIME and the plotting will go on until finally it covers the whole screen.

```
MODE 2: WHILE TIME:
RANDOMIZE TIME: PLOT
RND*640,RND*&400: WEND
```

Now the plotting will continue and the screen will get filled. I had not enough patience for timing this event. - Patrick De Geest

QUAINT CURSOR

While using my Amstrad CPC464 - no peripherals attached - I noticed strange things happening to my cursor when a certain combination of keys are pressed:

CHARACTER GENERATOR SNAG IRONED OUT

After attempting to run the character generator listed in the October 1986 issue of your magazine I stumbled across a minor problem.

I now have a 6128 having previously owned a 464. The problem lies in the memory used by the disk interfacc causing the address of the RAM copy of the UDGs to be lowered by about 1800 bytes after a SYMBOL AFTER command.

The address is now &9FFC instead of &A500 and line 870 should read:

870 DATA &01, &00, &07, &21,
&FC, &9F, &11, &58, &98, &7E

This solves the problem as far as I can see as there is still enough room below the UDGs for the second copy of the characters which is made by the program.

To load a saved set into your own program you must use:

SYMBOL AFTER 32: LOAD
"CHARS", &9FFC

I hope this solves the problem as it is truly an excellent utility with a really neat layout on screen.

John Thompson

ADD-ON SNAG

I was one of the first to buy the CPC464 and it has given me hours of pleasure ever since, but I am now irate about the software being written for it.

After having had the micro for a few months I bought a disk drive and immediately found that some programs would not load because of the space the drive had taken up.

After much frustration I have got round that problem only to find that it has now come up once again

because I have bought the Pace Nightingale modem and Cominstar pack.

My 3D Grand Prix, for example, won't load unless I unplug the modem from the back of the machine.

All this annoying messing about is just because software companies don't make allowances for everyone's add-ons.

Can anyone cure my 3D problem please? - D.G. Winston

• It's virtually impossible for software houses to make allowances for everyone's add-ons, mainly because they don't know what add-ons everyone's got.

Apart from that, 3D Grand Prix was written long before the Nightingale modem even saw the light of day.

We're afraid it's a fact of life that certain peripherals will always need memory required by various games. You have accepted that fact in relation to the disk drive and you must now accept it for the modem.

However utility ROMs such as Arnor's Utopia have the ability to disable both itself and any other ROMs in your micro.

This can be used to reclaim the memory and should allow you to run your software.

SOURCE OF ERROR

The other week I ordered some back copies from you, one of which contained the all singing, all dancing character generator program from the October 1986 issue of Computing with the Amstrad.

I duly typed it in, sorted out all my inevitable syntax errors and tried it out.

It worked beautifully until I tried to use the DATA facility, and then I got "NEXT MISSING IN 565".

I tried it out further and got "NEXT MISSING IN 620". I looked carefully through the program and counted all the FORs and all the NEXTs, and discovered

there were two fewer NEXTs than there were FORs.

I am at a loss to understand how a program could be printed with such an obvious error in it. Can you help me? - A. Entwistle

• There is no error in the program. I haven't copied the listing exactly printed.

You will have almost certainly missed a small REM tick before the ELSE in line 565 and the one before the FOR in line 620.

If you copy the program exactly as printed, it works perfectly.

TASWORD MARGIN ADJUSTMENT

A feature not present in Tasword that I had to include myself loading a textfile so that the left and right margins are automatically adjusted.

It is obvious that the margin values must be saved along with the textfile itself, so that when loading this file afterwards the default margin settings of the system are circumvented.

The locations of these two values can be found at the following addresses:

Left margin 29124 or &71C4 or s+20

Right margin 29128 or &71C8 or s+24

Lines 1610 and 1750 of Tasword can be amended as follows:

1610 MEMORY mh: OPENOUT a\$;
WRITE #9, PEEK (s+20), PEEK
(s+24): CALL c+39: CLOSEOUT:
MEMORY ml: RETURN

1750 MEMORY mh: OPENIN a\$;
INPUT #9, left, right: CALL c+45-
3*a: CLOSEIN: POKE s+20, left:
POKE s+24, right: MEMORY ml:
RETURN

Insert extra commands in the original lines and back-up Tasword. Do not use this version of

Clear the screen.

Press Shift and cursor up key.

Press Shift and cursor up key.

Press cursor up key.

Press cursor up key.

Press cursor down key.

Press cursor down key.

Press Shift and cursor down key.

Press Shift and cursor down key.

This sequence of operations causes two permanent cursor blobs - until you clear the screen or change mode - and the two normal cursor blobs which can be moved about the screen. Could you tell me what is happening? - Kevin Young

• *We're not sure what is wrong with your Anstrad. It certainly isn't normal behaviour. Has anyone else come across this problem?*

BUG DEFEATED

I seem to remember reading somewhere that there is a bug in the CPC464 disk drive which prevents the use of the merge command, and I must say that each time I try to use it I get an EOF MET error.

Is there any way Amstrad can be persuaded to carry out a merge, and thus make possible a subroutine library? - M.N. Bland

• *It's quite easy to get round the Merge bug. Simply save any programs or routines you want to merge in Ascii form using:*

SAVE "PROG",A

This saves the program in memory in Ascii form with the name PROG.

NOW FOR A REALLY BIG SCREEN DUMP

DURING Christmas my thoughts turned to producing signs and banners for the usual office parties. The program I came up with is called Banner and it prints any string in a very large style.

The string is printed at the top of the screen, then TEST is used to

scan the screen and where a pixel is set a group of characters is sent to the printer.

Because the scan covers the entire screen, with a simple modification a screen could be loaded and the program used to create the biggest screen dump you have ever seen!

I must admit to not having the paper or printer ribbon to do a full screen but I have done about a tenth of one. By my reckoning, when all the strips are joined it should be about 12 feet square.

I hop you agree that this program, as well as being easy to modify to your own needs, can give a lot of fun for very little typing. - John Blessing

```
10 REM BANNER
20 REM by John Blessing
20 MODE 0
30 PRINT "Make sure printer is
   ready"
40 PRINT #8, CHR$(27);
   CHR$(51); CHR$(20);
50 num=5
60 do$=STRING$(num,"*")
70 blank$=SPACES(num)
80 INK 0,26;INK 1,0;BORDER 26
90 INPUT "string";txt$
100 IF txt$="" THEN 90
110 CLS:PRINT txt$
130 FOR y=400 TO 1 STEP -16
140 FOR x=1 TO 640
150 FOR n=15 TO 0 STEP -1
160 IF TEST(x,y,n)=1 THEN
   PRINT #8, do$; ELSE
   PRINT #8, blank$;
170 NEXT
180 PRINT #8, PRINT #8
190 NEXT
200 PRINT #8, CHR$(12)
210 NEXT
```

RELOCATING MEMORY

WE HAVE taken your magazine since it inception and have found it to be the best, and most helpful on a practical basis for both our 464 and 6128 computers.

We would like to know the answers to a couple of points. Firstly, how can the memory be relocated in a Basic program to

accept longer strings on the number keys and allow more than characters, as is done with Tasword 6128 program?

Secondly, we have written a Basic program for invoicing and would like to use the extra memory available on the 6128. However we are not sure how the bank commands should be used. Indication for say, the Telephone program in the user manual would help. - Gerald J. Elliott

• *You can use strings of up to 255 characters in length if you relocate the extra buffer to an area of memory large enough to accommodate them. The best place for this is above 91969M and the following program will put it there.*

Experiment with the value of bufisz to obtain the minimum size of buffer sufficient for the number of strings you need.

```
10 FOR i=0 TO 14: READ v$
20 POKE&50+i,VAL("&"+"v$")
30 NEXT
40 DATA dd,6e,0,dd,66,1,dd,
   5e,2,dd,56,3,c3,15,bb
50 bufisz=1000:MEMORY HIME
   bufisz
60 CALL &50, HIME+1, bufisz
70 REM rest of program
```

We get many letters about the bank manager utility supplied with the 6128 and we hope to publish an article later in the year concerning its use. Storing strings in the extra ram requires your program to use the bank manager commands in a similar way to a random-access disk filing system. We more on that anon.

OUTLOOK BRIGHTER

I HAVE always thought that the screen of my CPC6128 was rather on the dark side. Even with the brightness turned up to full, some color combinations such as purple on black or green on red were very difficult to distinguish.

As I mainly use utilities and do a certain amount of programming, up to now this has been no great problem.

However I recently bought Dan Dare and the screen designs were

almost impossible to make out. As the micro is now out of warranty I decided to take a peek inside the monitor to see if any adjustments could be made - and indeed they can.

First of all make sure the computer is unplugged from the mains. Then disconnect the keyboard and lay the monitor screen-side down on some soft cloth to protect it.

On the main circuit board, halfway between the tube and the brightness control knob, there is a small trimmer or variable resistor.

It is about the size of a small tablet and has a horseshoe shaped piece of metal on its surface and in its centre there is a small slot. On the circuit board next to it is printed SUB RIGHT.

Using a small insulated screwdriver turn the slot to the right. On my Amstrad it needed only 1/16th of a turn to do the job and the results were stunning. Ink 6

is now red and not brown, and pastel blue really is that and not midnight blue.

A final word of warning. Even though you've removed the mains plug, don't touch any components, as capacitors can store a lethal charge, and microchips can be damaged by static discharge from your body.

If you're in any doubts as to what you're doing - get some assistance. - David Redwood

DISABLING ROMS ROUTINE

THIS short program was written for the CPC 464 and will disable all background ROMS except number 7 (the disk rom) without resetting the micro.

It does this by changing the operating system variables so that it thinks that only the disk rom is

present.

```
10 REM -- CPC 464 ONLY -
20 IF PEEK (&4E8)=255 THEN 70
30 FOR a=1 TO 6: POKE
  &B1AA+2*a,0: POKE
  &B1AB+2*a,0: NEXT
40 READ a,b: IF a<>0 THEN
  POKE a,b: GOTO 40
50 DATA &ae7d, &fb, &ae7e,
  &a6, &b294, 0, &b295, 0,
  &b296, 0, &b297, 0
60 DATA &b1a6, &fc, &b1a7,
  &a6, &a6fc, 0, &a6fd, 0,
  &a6fe, 7, 0, 0
70 REM rest of program
```

If you put the routine at the start of a disk menu program, all the programs on the disk will work even if they need external roms disabled. The routine is bypassed if Caps Lock is on.

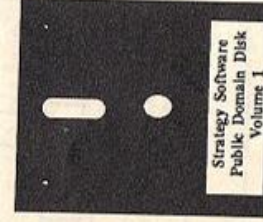
Gary Vine

END

Public Domain Disk Volume 1

NuSweep file manipulation program, Chequerbook program (CP/M 3.0 only), Sort for Ascii files, File comparator, plus various other useful file and disk utilities.

CAT #: 2801



Public Domain Disk Volume 3

PD3 features a full featured assembler and disassembler, again with full documentation - one of the best around and fully configured for your Amstrad.

CAT #: 2803



Public Domain Disk Volume 2

All you need to get you communicating with the outside world. Modem 9 with full documentation.

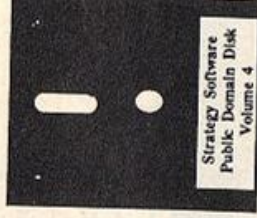
CAT #: 2802



Public Domain Disk Volume 4

Huge (300k) catalog of CP/M User Group software available. You'll spend weeks drooling over this lot - write and tell us what you'd like to see coming disks.

CAT #: 2804



See overleaf for ordering information

ORDERING INFORMATION

SOFTWARE ON TAPE

1001	TASWORD	\$36.95
1006	TOOLBOX	\$19.95
1007	FLEXIFREND	\$19.95
1008	GRASP	\$19.95
1009	CHAOS FACTOR	\$15.95
1010	MUZICO	\$17.95
1011	DRUMKIT	\$16.95
1012	MUSIC COMPOSER	\$17.95
1013	EASIDATA II	\$29.45
1014	EASIDATA III	\$41.45
1015	DATABASE/MAIL LIST	\$29.45
1022	QWERTY	\$14.95
1024	MYRDDIN FLIGHT	\$17.95
1030	AMS-FORTH	\$25.00
1201	PLAN-IT (CPC)	\$36.95
1202	MINI-OFFICE II	\$36.95
1203	MAGIC SWORD	\$21.95
1204	FUN SCHOOL (2-5)	\$15.95
1205	FUN SCHOOL (5-8)	\$15.95
1206	FUN SCHOOL (8-12)	\$15.95
1207	CHARTRUSTERS	\$15.95
1208	CLASSIC GAMES	\$15.95

SOFTWARE ON DISK

CAT #	TITLE	PRICE
2001	TASWORD	\$48.95
2009	CHAOS FACTOR	\$27.95
2012	MUSIC COMPOSER	\$29.95
2014	EASIDATA III	\$53.45
2015	DATABASE/MAIL LIST	\$41.45
2016	EASI-WORD COMBO	\$49.95
2017	GENESIS	\$39.95
2018	EASY MUSIC	\$34.95
2019	POT POURRI VOL. 1	\$19.95
2020	POT POURRI VOL. 2	\$19.95
2022	QWERTY	\$26.95
2024	MYRDDIN FLIGHT	\$29.95
2201	PLAN-IT	\$48.95
2202	MINI-OFFICE II	\$48.95
2203	MAGIC SWORD	\$33.95
2204	FUN SCHOOL (2-5)	\$27.95
2205	FUN SCHOOL (5-8)	\$27.95
2206	FUN SCHOOL (8-12)	\$27.95
2207	CHARTBUSTERS	\$27.95
2208	CLASSIC GAMES	\$27.95
2301	PLAN-IT (PCW)	\$27.95
2401	MT-BASIC (CPC)	\$59.95
2402	MT-BASIC (PCW)	\$125.00

PUBLIC DOMAIN DISKS

CAT #	TITLE	PRICE
2801	PD VOL. 1	\$19.95
2802	PD VOL. 2	\$19.95
2803	PD VOL. 3	\$19.95
2804	PD VOL. 4	\$19.95
2805	PD VOL. 5	\$19.95
2806	PD VOL. 6	\$19.95

ORDER FORM

CAT #	TITLE	PRICE
TOTAL		

Bankcard ☐ Mastercard ☐ Visa ☐

Signed _____ Expiry _____

Name _____
Address _____

Phone

State _____ Postcode _____

MAIL ORDERS TO:

**STRATEGY SOFTWARE
P.O. BOX 11
BLACKMANS BAY
TASMANIA 7152**

**ENQUIRIES
ORDERS**

ORDER TOLL-FREE WITH YOUR
VISA, MASTERCARD OR BANKCARD
SEE SIDE PANEL NEXT PAGE

SUBSCRIPTIONS

CAT #	TITLE	PRICE
5001	MAGAZINE ONLY	\$45.00
5002	MAGAZINE + TAPE	\$90.00
5003	MAGAZINE + QUARTERLY DISK	\$110.00

BACK ISSUES

CAT #	TITLE	PRICE
6008	CWTA PREMIERE EDITION	\$4.95
6009	CWTA SEPTEMBER ISSUE	\$4.95
6010	CWTA OCTOBER ISSUE	\$4.95
6011	CWTA NOVEMBER ISSUE	\$4.95
6012	CWTA DECEMBER ISSUE	\$4.95
6101	CWTA JANUARY ISSUE	\$4.95
6102	CWTA FEBRUARY ISSUE	\$4.95
6103	CWTA MARCH ISSUE	\$4.95
6104	CWTA APRIL ISSUE	\$4.95

TAPES

CAT #	TITLE	PRICE
7008	DIAMOND DIG	(8/86) \$7.50
7009	ICE FRONT	(9/86) \$7.50
7010	da BELLS	(10/86) \$7.50
7011	DISCMAN	(11/86) \$7.50
7012	ROBOT RON	(12/86) \$7.50
7101	OTHELLO	(1/87) \$7.50
7102	SPACE BASE	(2/87) \$7.50
7103	SMILEY	(3/87) \$7.50
7104	GALACTIC INV.	(4/87) \$7.50

DISKS

CAT #	TITLE	PRICE
7051	7008/7009/7010 AS ABOVE	\$19.95
7151	7011/7012/7101 AS ABOVE	\$19.95
7152	7102/7103/7104 AS ABOVE	\$19.95

BOOKS

CAT #	TITLE	PRICE
3001	AMSTRAD HANDBOOK	\$ 9.95
3002	AMSTRAD COMPUTING	\$17.95

ORDERING
INFORMATION

1. MAIL ORDER

Should you wish to order by mail but not wish to deface your magazine please photocopy P.72 or handwrite your order being careful to include all information requested on the order form. Please make sure you have enclosed your name and address (you'd be surprised!) and the correct amount for the goods you require.

If sending a cheque or ordering using Visa, Mastercard or Bankcard please ensure that the date on your cheque is valid (i.e. 1987 not 1986) and that your credit card has not expired.

2. TELEPHONE ORDER [008] 030930

This month we have installed a new toll-free order line. Please follow the instructions below carefully **before** ringing. Note that this number will only be answered by a machine and cannot be used for general enquiries or messages. Anything other than an order will be ignored - you have been warned!

- A) Complete the order form on Page 72 as though you were going to order by mail. Do not wait until ringing before deciding which titles you require or trying to find your credit card. The answering machine in use is voice activated and any pause over a couple of seconds will result in the machine hanging up on you.
- B) When the machine answers, read the order from your order form slowly, clearly and distinctly giving all the information you have written down. Where possible leave a telephone number just in case we can't understand or hear your order.
- C) This service will be in operation 24 hours a day, every day of the year.
- D) Allow 28 days for the delivery of your order - orders which we cannot despatch within 2-3 days of receipt will be advised of the likely delay by mail. Back issue orders will be mailed at the same time as the current issue.

**All prices include
postage & packing**

Five views on LocoScript

By **MICHAEL STERNE**

TO be worth buying, a book on LocoScript must offer more than the Amstrad manual does. Of these books, the first two are introductory texts, the next two are more complex, and the fifth defies classification.

Title: 15 Hour Word Processing Using the Amstrad Word Processor.
Author: Anna Ruthven
Price: \$19.95
Publisher: National Extension College

BACKED by the reputation of the National Extension College, this book is simple, clearly written and easily understood.

It leads you logically through the word processing maze, beginning with making a start-of-day disk. Everything is explained and each key press is highlighted on the page.

You are taken through text editing, with clear explanations of printing, other features, and word processing jargon - nothing essential is omitted.

If you have struggled to understand the manual, this book is for you. I thoroughly recommend it for the beginner.

Title: Getting Started with the Amstrad PCW8256/8512 Word Processor
Author: Susan Rogers
Price: \$24.95
Publisher: Phoenix Publishing

THIS also aims at the beginner's market, but I found it a little patronizing. Do you really need to be told: "Locate the printer ribbon. This is in a long oblong box." Or: "The screen, sometimes called a monitor, is similar to a television screen."

Susan Rogers reminds me of teachers I used not to get on with. "As you continue through this training guide, please be sure to read carefully and slowly. Do not attempt to carry out any of the instructions until actually told to do so", she italicizes. Good trainers know that you learn from your mistakes.

This is an easy reference book. You are given a clear explanation of each process and each menu. You also get a useful appendix listing the majority of the menus and the special keys.

As a beginner, however, you will want to understand the cursor movement keys before typing a letter and correcting errors. You are not taken through the process logically.

Moreover, the author hasn't understood that the capital letters within the menus provide codes so that these are incorrectly rendered. And you will not find descriptions of more advanced word-processing procedures.

Title: Amstrad Word Processing on The PCW8256
Author: Ian Sinclair
Price: \$29.95
Publisher: William Collins Sons & Co. Ltd.

THIS could be the book for you if you have some experience and need a straightforward but more complicated guide to LocoScript.

However beginners could well be confused by starting with direct printing, and only on page 72 will they find a description of the cursor movement keys.

The clear explanations of layout and how to use the features of the system in relation to various letters, documents and envelopes could save you much experimenting on your own to achieve the same results.

You will find excellent advice on the use of printer options, continuous printing and (from an experienced author) handling longer texts. You must, though, harden yourself to the occasional introduction of unfamiliar jargon.

This is a comprehensive book on LocoScript with useful menu Help pages at the end, and a good index - written by a knowledgeable author who has thoroughly investigated rather than just used LocoScript.

Title: Word Processing with the Amstrad PCW8256 and PCW8512
Author: Stephen Morris
Price: \$21.95
Publisher: Glentop Publishers

THIS book begins with an introduction to the equipment, the terminology, and also CP/M - however it's descriptive rather than instructional.

The book gave me no impression that the author had actually worked through the manual, testing all the features.

You are not guided in the use of LocoScript in practice -

the Amstrad manual is just paraphrased, but in clear and comprehensible prose. I found it hard to see the point.

Title: A Practical Reference Guide to Word Processing on the Amstrad
IPCW8256 and PCW8512
Author: FA Wilson
Price: \$19.95
Publisher: Bernard Babani

HERE you have the most idiosyncratic of these books, written by an enthusiast. It is quarto sized, printed on greyish paper using masters made directly from the output from the PCW's dot matrix printer.

It starts with an explanation of binary numbers, moves on to direct printing (you may not have bought a space disk), and then progresses straight into the complexities of the menus.

There's the first of a number of charts showing which keys to press and the menus you obtain in consequence. They seemed to me confusing and no more helpful than a simpler layout.

You are provided with suggestions for templates (the measurements to help you design appropriate layouts) and with such matters as the details of the key presses to produce the subscripts and superscripts for chemical and mathematical formulae.

In Appendix 4 there's a very useful tabulation of the ruler settings for different paper sizes and typefaces, and Appendix 6 is one of the most comprehensive

glossaries I have seen.

The book does not have sufficient clarity either of language or of explanation for beginners.

If you understand the system, but are seeking advice on handling address files, boilerplating and general techniques for continuous printing, producing envelopes and labels, it is not really adequate.

If you buy it, it can only be for use as a reference work when designing layouts.



FLEXI-FREND

HOME BUDGETING MADE EASY!

Among the many useful and creative applications of home computers, budget planning is one of the most popular. Flexi-Frend sets out to combine all the facilities needed for planning home finances into a program which is both user friendly and accurate.

Available now on both tape and disk
 See pages 72 & 73 for ordering information

Stranger than fiction

As we advised last month, the U.K. publishers of Computing With The Amstrad have purchased Amtix magazine. This will have significant effects on the way Computing With The Amstrad is presented and in the way it is published. At this stage we have not finalized preparations for what will essentially be a different magazine but it is likely the format will change with the July or August issues. We fully expect to see an increase in the number of pages to 96 or even more and along with the recent sales tax imposition on CWTA, will no doubt see a further price increase.

Apart from having drastic changes on CWTA as we know it today, the purchase of Amtix has resulted in a total re-organization of Database Publications with the result that CWTA will soon be split into two magazines - CWTA Leisure and CWTA Business. The obvious affects of this are more software reviews and listings in CWTA Leisure and a concentration on the more serious aspects of CPC and PCW computing in CWTA Business.

There is no doubt that these changes will strengthen CWTA's position as the world's number one Amstrad magazine and we hope that you find the changes to your liking.

As well as the aforementioned changes to CWTA we are pleased to announce a totally new Amstrad magazine - PC Amstrad. I don't think there is any need for even three guesses as to what the intention of PC Amstrad is! PC Amstrad will have it's inaugural issue in July and we'll be offering a special subscription rate to existing CWTA subscribers as we will on CWTA Business. Hopefully, we'll be able to give full details in next month's CWTA.

One other thing that may be of interest is that this is my last issue of CWTA. As of next month my good friend Rob McKenzie will be taking over the reigns while I do some fishing on doctors orders. Rob's a true pro and I'm sure that, with his influence as well as the aforementioned changes, you'll be well pleased with future issues of CWTA.

That's the good news, now for the bad. As most of you would be aware, last month the cover price of CWTA

went to \$4.50. I can assure you that this isn't greed on our part.

Just prior to the printing of the April issue we were advised by the Taxation department that Computer magazines are now, and have been since the last budget, subject to Sales Tax. This is apparently due to the decision handed down in last year's budget when a sales tax of 20% was invoked on all computer software.

As stated in 'My Say' in the October 1986 issue of CWTA, this new sales tax is pernicious because it isn't fair. We must all accept that taxes have to be paid but we also want the manner in which we pay our taxes to be as fair and equitable as possible. Without going over old ground too much we repeat that the Software Sales Tax act 1986 is the ONLY tax in Australia which can be levied at the retail level. I said in October that the act was a mess - and it is! Not only can the tax be levied at a retail level, but it also applies to published software *regardless of the media*. There is only one exception and that is software transmitted over data lines and air waves. Have you ever heard such nonsense?

The upshot of all this is that we now have to pay thousands of dollars in additional taxes each month and because we can't afford to pay out of our own pockets our readers have to fork out again. We're sorry about this but it is something which is totally beyond our control.

Now that it has been confirmed that computer magazines are taxable how long will it be before the vultures go for computer books? My guess is that it will be a long time - simply because the large book publishers have the resources to fight this and we haven't.

Rupert Murdoch took them on years ago when they tried to tax 'Best Bets', a horse racing journal aimed at giving punters a chance against the odds. There were millions at stake and he won because 'Best Bets' was designated 'educational in nature'. What a joke! If it wasn't so pitiful we could all have a good laugh. Still, anyone who has watched 'Yes Minister' will know that truth is stranger than fiction.

Kevin Poynton

MicroFile/MicroWorld review continued from page 56

using the facility to merge names, addresses and other information from MicroFile, the instructions have to be typed into the document itself.

Like LocoScript the program rightly assumes that printers give trouble from time to time and allows you to recover from a paper wreck and start printing from the page of your choice.

All in all a lot of thought has gone

into the system, although it has some limitations. MicroFile is particularly good and only the shortcomings of the manual let it down. The sections on both MicroFile and MicroWord should have included more worked examples - their omission means that you'll have to spend a lot longer on discovering how things work.

MicroWord is difficult to learn. And while it has some useful extra

features it lacks LocoScript's range of typefaces. Although my introduction to word processing was on a similar system to MicroWord, I now prefer LocoScript's pop-up menus.

Anyone looking for a database which can do simple calculations will like MicroFile. As a bonus, MicroWord, with it's mailmerge facility, is a faster word processor than LocoScript. Together the two make an impressive combination.