C16 C116 PLUS/4

COMMODORE COMPUTING MONTHLY

THE ONLY COMMODORE C16 C116 & PLUS/4 NEWSLETTER THAT IS MONTHLY!

Software:

Reviews

Programs

ETC

Hardware:

FORSALE

&

WANTED

Hoky Poky!

LOADS

MORE

INSIDE!

VOLUME 2 ISSUE 5 & 6 Aug/Sept 1990

Sorry my TV is blown up again, I PROMISE PUT BACK [illegible]
IN OCT/NOV ISH
VERY SORRY AGAIN!

Ed's Notes

Hello members

Sorry again for the delay, anyway find enclosed your bumper double issue Aug/Sept 1990.

Firstly I must opologise to Peter Crack and Members for last issues error, I printed Blooping Bug Part 6 & Part 8, Part 7 and Part 9 are in this issue, sorry again for the confusion. It also seems that some C16 members have tried to enter the BLOOPING BUG prog, well I have to tell you that the prog is C16+64k or Plus/4 runable only, so if you have an unexpanded C16 them sorry but you can't run the prog, this was clearly stated on the front cover of the January 1990 issue of the mag, so Peter Appleby, I think that answers your query about the program.

Last issue I said that Kevin Wheals was the only entrant for the Comp, how wrong I was, Peter Appleby also sent in a prog, so a couple of free issues for you to for your effort. So Kevin and Peter you will receive 2 free issues in the new year, the issues you get free are April & May 1991.

NEW MAG!!!

As from April 1991 the mag will become BI-MONTHLY, the name will be changed to 'COMMODORE C16/C116/+4 COMPUTING ONLY', I think the title says it all really, just for these machines!

Anyway I'll cut the gabble and let ya get on reading, also monthly members please fill in the form below for OCTOBER & NOVEMBER DOUBLE ISSUE, many thanks, please fill in and send back before November 23rd.

Please send me Volume 2 Issues 7 & 8 Oct/Nov of 'C16/C116/+4 COMP. MONTHLY.

I enclose a PO/CHEQUE for the value of £2.00

NAME & ADDRESS:

SIGNED: ____________________

All the best

Roy Robinson

```
********************************** BLOOPING BUG  part seven *****************
* By PETER and MELANIE CRACK.
* This month I am going to continue the programme where I lef off in May.
* I have submitted the programme in running order so watch the listing
* addresses as the do jump about a bit, first load both parts of your
* programme, then type in D443F and press return then type in this months
* additions, here is what it does.
443F-4444 Clear 'Y' register and transfer original sprite data to working
*         area byte by byte.
4447-444C Increase 'Y' register by one and compare it to $D9 (sprite string
*         length) if it is not equal then branch to $4441 and do it again
*         if it is then return from subroutine.
4322-433E Get random number =BASIC RND(Ø).
4322-4325 Switch in ROM and allow interupts.
4326-4328 Clear register $61 tells computer to get reseeded random number.
432A      gosub ROM random number routine.
432D-4333 Load 'A' register with timer one low byte and clear the six
*         leftmost bits (this register changes at a fast rate) transfer it
*         to 'X' register for use as an offset, 'X'register will now contain
*         a value between #$ØØ and #$Ø3, therefore our base random number
*         will be stored in either $62,$63,$64 or $65 load one of these into
*         'A' register.
4335-4338 Clear the five leftmost bits and divide by two (LSR) thus giving a
*         random number in the range #$ØØ-#$Ø3 finally store this number in
*         $D9 (delay reload).
433A-433E Disable interupts switch out ROM switch in RAM and return from sub
45ØØ-45F5 Sprite print subroutine.
45ØØ-45Ø8 Load 'A' register with $DF (points to centre of sprite down the
*         screen) clear carry flag add #$34 (to ensure value now points to
*         somwhere just below sprite)  and compare it with screen raster
*         position $FF1D, if it is not there then wait else.........
45ØA-45ØC Load 'X' register with #$Ø2 and gosub get old sprite position.
45ØF-4514 Clear 'X' register, load 'Y' register with $DA (sprite depth
*         pointer) transfer 'Y' to 'A' regiter and save a copy on the stack.
4515-451E Clear 'Y' register,load old sprite data into 'A' register, invert
*         it, that is to say any bits which were switched on will now be off
*         and vice-versa, AND it with screen byte, this will remove any bits
*         which are not switched on in both the 'A' register and the screen
*         byte (this also removes everything that was behind the sprite so
*         background details cannot be used with this system, also when two
*         sprites collide some flickering can be seen, I AM WORKING ON THIS!
*         I have also developed a system using a single sprite which does
*         allow background to be used!!) and then store the result back into
*         the screen using 'Y' register as an offset.
452Ø-4528 increase 'X' register by one, transfer 'Y' to 'A' register clear
*         the carry flag and add #$Ø8 (this is to position the next print
*         byte command in $4516 over the correct byte in the next character
*         square to the right of the last one), transfer 'A' to 'Y' register
*         and compare it with the sprite width register ($DB) have we done
*         one row, no? then branch to $4517 else.............
452A      Gosub move down one byte (see article in MARCH issue).
452D-453Ø Pull sprite depth pointer back off stack, tranfer it to 'Y' reg.
*         decrease it, check to see if it has been counted down to -1 (#$FF)
*         have we done all the rows of the sprite?, no, then branch to $4513
*         yes, then sprite removal is complete (by placing a RTS command at
*         $453B the above routine is also used as part of the sprite switch
*         off routine, the address $4519 is changed to point to the data
*         page of the sprite which is being processed.
*         The following part of the routine is also used as part of the
*         switch  sprite on routine.
4532-4536 Clear 'A',collision and collision flag registers.
4538      Gosub, 'get sprite new position'.
453B-454Ø Clear 'X' register, load 'Y' register with sprite depth pointer
*         tranfer 'Y' to 'A' register and save a copy on stack.
************************************************** CONTINUED **************
```

```
*****************************************************************************
4541-4545 Clear 'Y' register,load a screen byte offset by 'Y' register and
*         save a copy on stack.
4546      AND it with a byte if sprite working data (that is to say compare
*         both bytes bit by bit and only those bits which are switched on
*         in both bytes will remain switched on in 'A' register hence the
*         need to save a copy on stack).
4549-454B Is the 'A' register now zero?, if not there must be another sprite
*         in collision with this one, if 'A' register is zero then branch to
*         $454D else save 'A' register in $E4 (collision flag).
454D-4551 Pull original screen byte off stack mix in sprite working data
*         byte and store it back into the screen.
4553-455B Increase 'X' register by one, transfer 'Y' to 'A' registers clear
*         the carry flag add #$Ø8 transfer 'A' to 'Y' register and compare
*         it with sprite width pointer $DB, (have we printed one row?), if
*         not equal branch to $4543 and do it all again else............
455D-455F Load 'Y' register with $E4 (collision flag) check to see if it is
*         zero if it is then branch to $458B else establish where collision
*         occured.
4561-4562 Pull sprite depth counter off stack (this automaticaly puts it
*         into the 'A' register), push a copy back onto stack, this has to
*         be done because every PHA or PLA command increases or decreases
*         the stack pointer, and later in this routine we wish to pull (PLA)
*         this value back of the stack.
4563-4567 Compare 'A' register with zero if it is then then we have printed
*         all of the sprite and collision has occured on the bottom row so
*         branch to $4581, else compare 'A' register with the original value
*         of $DA if it is equal to $DA then collision has occured at the top
*         so branch to $457C, if it is not top or bottom then it must be in
*         the middle so.........
4569-456F Do a BIT test on $EØ, remember $EØ contains #$8Ø if the sprite is
*         moving left and #$4Ø if moving right, this test checks to see if
*         bit seven or bit six are set (switched on) and sets the minus flag
*         and/or the overflow flag accordingly, first we check to see if the
*         minus flag is set, if yes then branch to $4577, if the overflow
*         flag is set then branch to $4572 and if neither are set then jump
*         to $4587.
4572-4581 Sets the correct bits in 'A' register.
4583-4585 Mix collision occured register with 'A' register and store the
*         result in $D8, remember this sprite could be in collision with
*         several other sprites and all four collision flags could be
*         switched on.
4587-4589 Clear collision flag ($EØ) ready for next sprite row.
458B      Gosub 'increase row pointers'.
458E-4591 Pull sprite depth counter off stack transfer it to 'Y' register
*         and decrease it by one, check to see if it has been counted down
*         to -1 (#$FF) if no branch to $453F else.............
4593-45AC Set $62 and $64 to zero, set $63 and $65 to point to sprite
*         working and old sprite data pages, and transfer old sprite data to
*         from working page to old sprite data page, this is to ensure that
*         only old sprite data is removed from the screen in the routine
*         from $45ØØ to $453Ø.
45AE-45B8 Store $D2 and $D3 in $D4 and $D5 that is to say transfer new
*         positions from sprite temp. variables to position pointers, switch
*         off voice two and finally return from subroutine.
45E1-45EB Clear 'X' register and transfer $D2 and $D3 offset by 'X' register
*         to (very) temporary screen address pointers, this routine is also
*         entered at $45E3 with 'X' register set to #$Ø2.
45EC-45F4 Transfer (very) temporary screen pointers to temporary screen
*         pointers.
41AØ-41C5 Returns all variable sprite pointers to sprite data group, set in
*         $46ØØ to $46FF and is the reverse of the routine at $41EF to $4261
*         AND THATS IT FOR THIS MONTH......................
*         As always any problems or hints or comments ring me on Ø81-367-3152*        or write
 to the magazine....................PETER.........
*****************************************************************************
```

```
. 45ØØ  A5 DF     LDA $DF
. 45Ø2  18        CLC
. 45Ø3  69 34     ADC #$34
. 45Ø5  CD 1D FF  CMP $FF1D
. 45Ø8  DØ FB     BNE $45Ø5
. 45ØA  A2 Ø2     LDX #$Ø2
. 45ØC  2Ø E3 45  JSR $45E3
. 45ØF  A2 ØØ     LDX #$ØØ
. 4511  A4 DA     LDY $DA
. 4513  98        TYA
. 4514  48        PHA
. 4515  AØ ØØ     LDY #$ØØ
. 4517  BD ØØ 8Ø  LDA $8ØØØ,X
. 451A  49 FF     EOR #$FF
. 451C  31 DØ     AND ($DØ),Y
. 451E  91 DØ     STA ($DØ),Y
. 452Ø  E8        INX
. 4521  98        TYA
. 4522  18        CLC
. 4523  69 Ø8     ADC #$Ø8
. 4525  A8        TAY
. 4526  C4 DB     CPY $DB
. 4528  DØ ED     BNE $4517
. 452A  2Ø CØ 45  JSR $45CØ
. 452D  68        PLA
. 452E  A8        TAY
. 452F  88        DEY
. 453Ø  1Ø E1     BPL $4513
. 4532  A9 ØØ     LDA #$ØØ
. 4534  85 D8     STA $D8
. 4536  85 E4     STA $E4
. 4538  2Ø E1 45  JSR $45E1
. 453B  A2 ØØ     LDX #$ØØ
. 453D  A4 DA     LDY $DA
. 453F  98        TYA
. 454Ø  48        PHA
. 4541  AØ ØØ     LDY #$ØØ
. 4543  B1 DØ     LDA ($DØ),Y
. 4545  48        PHA
. 4546  3D ØØ 81  AND $81ØØ,X
. 4549  FØ Ø2     BEQ $454D
. 454B  85 E4     STA $E4
. 454D  68        PLA
. 454E  1D ØØ 81  ORA $81ØØ,X
. 4551  91 DØ     STA ($DØ),Y
. 4553  E8        INX
. 4554  98        TYA
. 4555  18        CLC
. 4556  69 Ø8     ADC #$Ø8
. 4558  A8        TAY
. 4559  C4 DB     CPY $DB
. 455B  DØ E6     BNE $4543
. 455D  A4 E4     LDY $E4
. 455F  FØ 2A     BEQ $458B
. 4561  68        PLA
. 4562  48        PHA
. 4563  FØ 1C     BEQ $4581
. 4565  C5 DA     CMP $DA
. 4567  FØ 13     BEQ $457C
. 4569  24 EØ     BIT $EØ
. 456B  3Ø ØA     BMI $4577
. 456D  7Ø Ø3     BVS $4572
. 456F  4C 87 45  JMP $4587
. 4572  A9 Ø8     LDA #$Ø8
. 4574  4C 83 45  JMP $4583
. 4577  A9 Ø4     LDA #$Ø4
```

```
. 4579  4C 83 45  JMP $4583
. 457C  A9 Ø1     LDA #$Ø1
. 457E  4C 83 45  JMP $4583
. 4581  A9 Ø2     LDA #$Ø2
. 4583  Ø5 D8     ORA $D8
. 4585  85 D8     STA $D8
. 4587  A9 ØØ     LDA #$ØØ
. 4589  85 E4     STA $E4
. 458B  2Ø CØ 45  JSR $45CØ
. 458E  68        PLA
. 458F  A8        TAY
. 459Ø  88        DEY
. 4591  1Ø AC     BPL $453F
. 4593  A9 ØØ     LDA #$ØØ
. 4595  85 62     STA $62
. 4597  85 64     STA $64
. 4599  AD 1C 44  LDA $441C
. 459C  85 63     STA $63
. 459E  AD 48 45  LDA $4548
. 45A1  85 65     STA $65
. 45A3  A4 D9     LDY $D9
. 45A5  B1 64     LDA ($64),Y
. 45A7  91 62     STA ($62),Y
. 45A9  88        DEY
. 45AA  CØ FF     CPY #$FF
. 45AC  DØ F7     BNE $45A5
. 45AE  A5 D2     LDA $D2
. 45BØ  85 D4     STA $D4
. 45B2  A5 D3     LDA $D3
. 45B4  85 D5     STA $D5
. 45B6  A9 ØØ     LDA #$ØØ
. 45B8  8D 11 FF  STA $FF11
. 45BB  6Ø        RTS
. 45BC  EA        NOP
```

```
. 41AØ  A5 E5     LDA $E5
. 41A2  ØA        ASL
. 41A3  ØA        ASL
. 41A4  ØA        ASL
. 41A5  ØA        ASL
. 41A6  A8        TAY
. 41A7  18        CLC
. 41A8  69 ØE     ADC #$ØE
. 41AA  85 E4     STA $E4
. 41AC  A2 ØØ     LDX #$ØØ
. 41AE  B5 D4     LDA $D4,X
. 41BØ  99 ØØ 46  STA $46ØØ,Y
. 41B3  E8        INX
. 41B4  C8        INY
. 41B5  C4 E4     CPY $E4
. 41B7  DØ F5     BNE $41AE
. 41B9  AD ØA 43  LDA $43ØA
. 41BC  99 ØØ 46  STA $46ØØ,Y
. 41BF  AD 18 43  LDA $4318
. 41C2  99 Ø1 46  STA $46Ø1,Y
. 41C5  6Ø        RTS
```

```
. 443F  AØ ØØ     LDY #$ØØ
. 4441  B9 Ø1 82  LDA $82Ø1,Y
. 4444  99 ØØ 81  STA $81ØØ,Y
. 4447  C8        INY
. 4448  C4 D9     CPY $D9
. 444A  DØ F5     BNE $4441
. 444C  6Ø        RTS
. 444D  EA        NOP
. 45E1  A2 00     LDX #$00
. 45E3  B5 D2     LDA $D2,X
. 45E5  85 D0     STA $D0
. 45E7  B5 D3     LDA $D3,X
. 45E9  85 D1     STA $D1
. 45EB  60        RTS
. 45EC  A5 D0     LDA $D0
. 45EE  85 D2     STA $D2
. 45F0  A5 D1     LDA $D1
. 45F2  85 D3     STA $D3
. 45F4  60        RTS
. 45F5  EA        NOP
```

```
. 4322  8D 3E FF STA $FF3E
. 4325  58       CLI
. 4326  A9 00    LDA #$00
. 4328  85 61    STA $61
. 432A  20 07 A7 JSR $A707
. 432D  AD 00 FF LDA $FF00
. 4330  29 03    AND #$03
. 4332  AA       TAX
. 4333  B5 62    LDA $62,X
. 4335  29 07    AND #$07
. 4337  4A       LSR
. 4338  85 DC    STA $DC
. 433A  78       SEI
. 433B  8D 3F FF STA $FF3F
. 433E  60       RTS
. 433F  EA       NOP
```

CONT'T ON
NEXT PAGE

CLUB NEWS

Encore have just released a game for the C64, CPCs and SPECTRUM. It is a two game compilation of two of the best flight sims on Budget lable, they are ACE and ACE2, for only £2.99 (when each one costs that alone).

Rumour has it that they might soon release a Plus/4 version of the game. A C-16 version is slightly out of the question as although there is a C-16 ACE there is no C-16 ACE2 available in the shops.

Commodore has just finished a brand new Commodore Console with the same capabilities of a C64 or Plus/4. You never know there might even be a few C-16 Plus/4 compatilbilities (or however you spell it!).

Away now from the Commodore 8-bits; Commodore has finish and will release soon the Commodore Amiga A5000, yes an A5000.

That's all on the grape vine for the moment so bye for now, Matt (Matthew Newton-Lewis).

```
*************************** BLOOPING BUG part nine **************************
************************ by PETER and MELANIE CRACK *************************
*****************************************************************************
* This month I want to add the routine which makes the sprites bounce off
* the edge of the screen and the scoring routines, so first load both parts
* of the programme then enter D43EØ and press return, here is the
* explanation. BUT WATCH THE LOCATION ADDRESSES!!!
43EØ-43FD This short routine changes the direction address at $43ØA or $4318
*         and then jumps back into the opposite movement routine to the one
*         from which it came thus the sprite hits the edge of the screen
*         and without seeming to stop bounces off at the same speed.
4E48-4E4D Set $DA (first number not moved flag) to #$8Ø -128 DEC, if this
*         flag is not reset to zero by the folowing routine then no numbers
*         will be moved.
4E5Ø-4E58 Load first number (right hand number of display) data pointer
*         into $D5-$D8 (all three numbers are held in groups of four data
*         pointers from $4E36-$4E39,$4E3A-$4E3D and $4E4Ø-$4E43 and are used
*         as follows ... $D5=data address low,$D6=number of rows (for this
*         number) still to be printed,$D7=the number of hits still to be
*         added to your score,$D8=data addres high byte.
4E5A      Gosub evaluate and print number.
4E5D-4E65 Return updated pointers to their correct psitions.
4E67-4E69 Check to see if $DA is still #$8Ø or -128 DEC if it is then branch
*         to $4E8B else.........
4E6B-4E71 Clear $DA and place correct offset into print routine.
4E74-4E7C Get data pointers for this (centre or tens) number.
4E7E      Gosub print.
4E81-4E8B Return updated pointer to pointer list and JuMP to $4F1Ø.
4F1Ø-4F12 Check $DA if minus then branch to $4F3Ø else........
4F14-4F16 Put correct offset into print routine.
4F19-4F23 Get data pointers and gosub print routine.
4F26-4F2E Put updated pointers back into data list.
4F3Ø-4F35 Put original offset back into print routine and jump to next part
*         of routine.
4EØØ-4E2F First part of print routine.
4EØØ-4EØ1 Stop interrupt and switch out rom.
4EØ4-4EØ6 Check $D6 (number of rows still to be printed to bring this
*         number fully into position) branch to $4E13 if not zero else.....
4EØ8-4EØB Load 'Y' register with $D7 (number of hits still to be added to
*         your score) decrease 'Y' register by one (this is also going to
*         bring up the next number) check to see if it has been counted down
*         to -1 dec. or #$FF hex if yes then branch to $4E2B else..........
4EØD-4E11 Store 'Y' register back into $D7, and load 'A' register with #$18
*         (dec24=number of times this routine has to be performed to bring
*         next number fully into the window on screen) and store it in $D6
4E13-4E16 Gosub setup print routine and gosub print (explained in march).
4E19-4E1C Load 'Y' register with $D6, decrease it by one and check to see if
*         it has been counted down to #$FF if it has then branch to $4E23.
4E1E-4E2Ø Store 'Y' register in $D6 and jump to $4E27.
4E23-4E2F Set both $D6 and $DA to zero, switch ROM back in,allow interupts
*         and return from subroutine.
4EC8-4ECA Load 'A' register with $D8 and store it in the print routine.
4ECD-4ED2 Add #$03 to $D5 (remember each number is three bytes wide so to
*         bring it one row up the display area we have to take the data
*         bytes from three places further along the data string) .
4ED2-4ED6 Check to see if $D5 has been counted over #$FF if yes then
*         increase page number by one in $D8 and $4EAC.
4ED9-4EF7 This part of the routine is a bit complex for it has to check
*         (1) if it has reached the end of the number data string and
*         (2) which of the three numbers it is printing, so first we check
*         to see if $D5=#$88 and $D8=#$BA if they have then this means we
*         have just put a nine completely into the units column and are
*         about to pull up a zero, therefore we must increase $4E42 (the
*         tens column amount not yet added to your score flag), if the above
*         test is 'true' then the next test will fail and the programme will
*.............................................CONTINUED............
```

```
*        branch to $4FØ8, next we check if $D5=#$DØ and $D8=#$BA if yes
*        then a trailing zero has just been pulled up into the tens column
*        of the score window (remember the number data string is laid out
*        as follows...Ø,1,2,3,4,5,6,7,8,9,Ø the last or trailing zero is
*        there to simplify the printing and testing routines) if this
*        second test is 'true' we must be sure that the last number printed
*        was in the tens column, to do this we check on the offset value in
*        $4E91 if it was not #$18 then it was not the tens column if it was
*        #$18 then increase the hundreds column by one that is increase
*        $4E38 by one.
4EFF-4FØC Finally if any number last pulled completely up into the score
*        window was a trailing zero then the data pointers have reached the
*        end of the numbers data string, so load 'A' register with #$B8
*        (address of data string high byte) and store it in the print
*        routine and the data pointer for this number ($D8), load 'A'
*        register with zero and store it in $D5 (address of data string
*        low) and $D4 (print routine page offset pointer), if this test at
*        $4EE6-$4EEC fails then the programme branches to $4FØ8 and $D5 and
*        $D4 are updated to the new page offset values.
*        AND THATS THE END OF THIS MONTHS ARTICLE.
*
*        LASTLY DO NOT WORRY IF YOU CANNOT MAKE SENSE OF THIS LOT AT FIRST
*        SIGHT I FOUND IT VERY DIFFICULT TO EXPLAIN AND I WROTE IT!!!!!
*        BUT IF YOU DO HAVE ANY PROBLEMS THEN GIVE ME A RING OR WRITE
*        IN TO THE MAGAZINE OUTLINING ANY QUERIES....
*
*                        all the best till next month
*                            PETER CRACK (Ø81-367-3152)
```

```
. 43EØ  A9 ØØ     LDA #$ØØ
. 43E2  8D ØA 43  STA $43ØA
. 43E5  4C ØØ 44  JMP $44ØØ
. 43E8  A9 5Ø     LDA #$5Ø
. 43EA  8D ØA 43  STA $43ØA
. 43ED  4C 5Ø 44  JMP $445Ø
. 43FØ  A9 D8     LDA #$D8
. 43F2  8D 18 43  STA $4318
. 43F5  4C D8 44  JMP $44D8
. 43F8  A9 AØ     LDA #$AØ
. 43FA  8D 18 43  STA $4318
. 43FD  4C AØ 44  JMP $44AØ
```

CONT OVER →

```
* Dear ROY,
*          The mag is as good if not better then ever keep it up!!!!
* did you know that you had put the wrong part in the JUNE/JULY issue !!
* the next bit is for the letters page.
*                all the best
*                          PETER.
*
```

Thanks peter (Ed)

DEAR ED.

```
* I have received a letter from KEVIN WHEALS asking how to print out a
* memory dump or a dissasembly and I thought it might be of interest to
* other club members, so here goes.
* First make a note of all the areas you want to print out like this D12ØØ
* 1268, for a dissasembly or M12ØØ 1268 for a hex dump having written down
* this list I now press 'X' and enter basic, use the 'OPEN' command to open
* the printer channel,press return then 'CMD file number', just as it sayes
* in the manual, and press,return,both these commands are in the immediate
* mode, I.E no line numbers, If all is well then the printer will have
* printed 'READY' or at least made a noise, now enter 'MONITOR', and press
* return, the printer should now print out the status registers, on a new
* line, enter the commands as written on your side note that is to say the
* list of areas you want printed, and that is it, any problems let me know.
*                        all the best
*
*
*                            PETER CRACK.
```

```
. 4E00  78        SEI
. 4E01  8D 3F FF  STA $FF3F
. 4E04  A5 D6     LDA $D6
. 4E06  D0 0B     BNE $4E13
. 4E08  A4 D7     LDY $D7
. 4E0A  88        DEY
. 4E0B  30 1E     BMI $4E2B
. 4E0D  84 D7     STY $D7
. 4E0F  A9 18     LDA #$18
. 4E11  85 D6     STA $D6
. 4E13  20 C8 4E  JSR $4EC8
. 4E16  20 90 4E  JSR $4E90
. 4E19  A4 D6     LDY $D6
. 4E1B  88        DEY
. 4E1C  30 05     BMI $4E23
. 4E1E  84 D6     STY $D6
. 4E20  4C 27 4E  JMP $4E27
. 4E23  A9 00     LDA #$00
. 4E25  85 D6     STA $D6
. 4E27  A9 00     LDA #$00
. 4E29  85 DA     STA $DA
. 4E2B  8D 3E FF  STA $FF3E
. 4E2E  58        CLI
. 4E2F  60        RTS
. 4E30  EA        NOP

. 4E48  A9 80     LDA #$80
. 4E4A  85 DA     STA $DA
. 4E4C  EA        NOP
. 4E4D  EA        NOP
. 4E4E  EA        NOP
. 4E4F  EA        NOP
. 4E50  A2 03     LDX #$03
. 4E52  BD 3A 4E  LDA $4E3A,X
. 4E55  95 D5     STA $D5,X
. 4E57  CA        DEX
. 4E58  10 F8     BPL $4E52
. 4E5A  20 00 4E  JSR $4E00
. 4E5D  A2 03     LDX #$03
. 4E5F  B5 D5     LDA $D5,X
. 4E61  9D 3A 4E  STA $4E3A,X
. 4E64  CA        DEX
. 4E65  10 F8     BPL $4E5F
. 4E67  A5 DA     LDA $DA
. 4E69  30 20     BMI $4E8B
. 4E6B  A9 00     LDA #$00
. 4E6D  85 DA     STA $DA
. 4E6F  A9 18     LDA #$18
. 4E71  8D 91 4E  STA $4E91
. 4E74  A2 03     LDX #$03
. 4E76  BD 40 4E  LDA $4E40,X
. 4E79  95 D5     STA $D5,X
. 4E7B  CA        DEX
. 4E7C  10 F8     BPL $4E76
. 4E7E  20 00 4E  JSR $4E00
. 4E81  A2 03     LDX #$03
. 4E83  B5 D5     LDA $D5,X
. 4E85  9D 40 4E  STA $4E40,X
. 4E88  CA        DEX
. 4E89  10 F8     BPL $4E83
. 4E8B  4C 10 4F  JMP $4F10
. 4E8E  EA        NOP
. 4E8F  EA        NOP
```

```
. 4EC7  EA        NOP
. 4EC8  A5 D8     LDA $D8
. 4ECA  8D AC 4E  STA $4EAC
. 4ECD  A5 D5     LDA $D5
. 4ECF  18        CLC
. 4ED0  69 03     ADC #$03
. 4ED2  90 05     BCC $4ED9
. 4ED4  E6 D8     INC $D8
. 4ED6  EE AC 4E  INC $4EAC
. 4ED9  C9 88     CMP #$88
. 4EDB  D0 09     BNE $4EE6
. 4EDD  A4 D8     LDY $D8
. 4EDF  C0 BA     CPY #$BA
. 4EE1  D0 03     BNE $4EE6
. 4EE3  EE 42 4E  INC $4E42
. 4EE6  C9 D0     CMP #$D0
. 4EE8  D0 1E     BNE $4F08
. 4EEA  A4 D8     LDY $D3
. 4EEC  C0 BA     CPY #$BA
. 4EEE  D0 18     BNE $4F08
. 4EF0  AE 91 4E  LDX $4E91
. 4EF3  E0 18     CPX #$18
. 4EF5  D0 03     BNE $4EFA
. 4EF7  EE 38 4E  INC $4E38
. 4EFA  EA        NOP
. 4EFB  EA        NOP
. 4EFC  EA        NOP
. 4EFD  EA        NOP
. 4EFE  EA        NOP
. 4EFF  A9 B8     LDA #$B8
. 4F01  8D AC 4E  STA $4EAC
. 4F04  85 D8     STA $D8
. 4F06  A9 00     LDA #$00
. 4F08  85 D5     STA $D5
. 4F0A  85 D4     STA $D4
. 4F0C  60        RTS
. 4F0D  EA        NOP

. 4F10  A5 DA     LDA $DA
. 4F12  30 1C     BMI $4F30
. 4F14  A9 00     LDA #$00
. 4F16  8D 91 4E  STA $4E91
. 4F19  A2 03     LDX #$03
. 4F1B  BD 36 4E  LDA $4E36,X
. 4F1E  95 D5     STA $D5,X
. 4F20  CA        DEX
. 4F21  10 F8     BPL $4F1B
. 4F23  20 00 4E  JSR $4E00
. 4F26  A2 03     LDX #$03
. 4F28  B5 D5     LDA $D5,X
. 4F2A  9D 36 4E  STA $4E36,X
. 4F2D  CA        DEX
. 4F2E  10 F8     BPL $4F28
. 4F30  A9 30     LDA #$30
. 4F32  8D 91 4E  STA $4E91
. 4F35  4C C6 41  JMP $41C6
```

CONT
NEXT
LSH

```
2 REM ******************
3 REM * DAVID BOWIE    *
4 REM * HI-RES PICTURE *
5 REM ******************
6 REM * BY KEVIN WHEALS*
7 REM *                *
8 REM * (C)1986(P)1990 *
9 REM ******************
10 COLOR0,2:COLOR1,3,0:COLOR4,2
20 POKE55,0:POKE56,60:CLR:POKE1177,62
30 FORI=0TO1023:POKE15360+I,PEEK(53248+I):NEXTI
40 POKE1177,63:POKE65299,60:POKE65298,192:SCNCLR
50 FORL=0TO107:CX=0:FORD=0TO7:READCH:CX=CX+CH:POKE15360+L*8+D,CH:NEXTD
60 READCH:IFCH<>CXTHENPOKE65299,208:POKE65298,196:PRINT"ERROR IN LINE";1000+(L*1
0):STOP
70 NEXTL
100 PRINT"(DOWN)(DOWN)(DOWN)(DOWN)"
110 PRINTTAB(14);:PRINT"@ABCDEF"
120 PRINTTAB(13);:PRINT"GHIJKLMN"
130 PRINTTAB(13);:PRINT"OPQRSTUVW"
140 PRINTTAB(13);:PRINT"XYZ[£]↑←!"
150 PRINTTAB(13);:PRINT"#$%&'()*"
160 PRINTTAB(13);:PRINT"+,-./012"
170 PRINTTAB(13);:PRINT"3456789:"
180 PRINTTAB(14);:PRINT";<=> ?─":REM SHIFT *  , IS THE LAST CHAR ON THIS LINE
190 PRINTTAB(14);:PRINT"♠│─▔ ▔─":REM USE SHIFT A B C D, NORMAL SPACE, SHIFT E F
200 PRINTTAB(14);:PRINT"│ ╮╲ ╰└╲":REM USE SHIFT G H I J K L M
210 PRINTTAB(13);:PRINT"╱┌┐●─♥│ ╱":REM USE SHIFT N O P Q R S T U
220 PRINTTAB(13);:PRINT"×○♣┼│♦┼":REM USE SHIFT V W X Y Z +
230 PRINTTAB(11);:PRINT"▒ │π◥$ ×F ▌▬":REM USE CBM - SHIFT - CBM = *
235 REM FIRST SPACE IS NORMAL, SHIFT V, AFTER THE F ITS SHIFT SPACE, CBM K I
240 PRINTTAB(14);:PRINT"▔▁ │ ▒ ▙ ◤":REM USE CBM T @, NORMAL SPACE, CBM G + M £
245 REM NORMAL SPACE, SHIFT £
250 PRINTTAB(17);:PRINT" │├":REM USE CBM N Q
300 GOTO300
500 REM *** UDG DATA FOLLOWS ***
1000 DATA 000,000,031,000,056,006,019,047, 0159
1010 DATA 036,018,009,138,095,190,248,224, 0958
1020 DATA 052,071,127,255,255,063,079,142, 1044
1030 DATA 000,061,255,255,245,243,227,070, 1356
1040 DATA 000,048,252,255,191,087,073,150, 1056
1050 DATA 000,000,000,000,192,240,254,127, 0813
1060 DATA 000,000,000,000,000,000,000,128, 0128
1070 DATA 000,000,001,003,003,015,015,031, 0068
1080 DATA 015,031,254,255,254,253,229,240, 1531
1090 DATA 004,008,036,066,068,036,136,203, 0557
1100 DATA 132,016,048,032,050,067,001,129, 0475
1110 DATA 128,000,000,048,036,064,128,000, 0404
1120 DATA 000,001,018,032,034,068,074,088, 0315
1130 DATA 207,031,031,039,065,038,072,193, 0676
1140 DATA 000,000,128,192,224,240,248,062, 1094
1150 DATA 127,127,127,127,175,111,055,047, 0896
1160 DATA 232,196,226,128,073,164,080,008, 1107
1170 DATA 025,060,250,255,255,254,126,127, 1352
1180 DATA 003,131,067,001,135,015,017,004, 0373
1190 DATA 016,008,005,003,131,225,007,003, 0398
1200 DATA 082,102,076,089,210,244,245,244, 1292
```

```
1210 DATA 084,136,005,035,040,080,098,100,  0578
1220 DATA 124,222,160,047,035,199,015,175,  0977
1230 DATA 000,000,000,128,192,240,136,128,  0824
1240 DATA 023,015,015,007,007,015,007,015,  0104
1250 DATA 160,144,232,242,253,240,192,128,  1591
1260 DATA 094,129,012,002,241,008,007,000,  0493
1270 DATA 002,033,147,015,227,255,252,072,  1003
1280 DATA 049,248,255,255,255,255,254,000,  1571
1290 DATA 244,244,244,202,132,059,124,252,  1501
1300 DATA 204,222,220,152,035,076,209,059,  1177
1310 DATA 127,255,255,254,252,252,252,252,  1899
1320 DATA 000,000,000,000,000,000,000,000,  0000
1330 DATA 128,128,000,000,000,000,000,000,  0256
1340 DATA 000,000,000,000,000,000,000,000,  0000
1350 DATA 015,007,007,003,003,007,015,015,  0072
1360 DATA 128,128,128,128,128,128,128,128,  1024
1370 DATA 000,000,001,001,002,004,009,009,  0026
1380 DATA 079,159,031,127,127,255,255,255,  1288
1390 DATA 255,255,255,255,255,255,254,252,  2036
1400 DATA 248,240,224,192,128,000,000,000,  1032
1410 DATA 007,007,007,007,007,007,007,007,  0056
1420 DATA 248,248,248,248,248,252,252,252,  1996
1430 DATA 015,015,015,007,006,004,004,002,  0068
1440 DATA 128,128,128,128,128,128,128,129,  1025
1450 DATA 019,023,039,047,079,159,191,000,  0557
1460 DATA 255,255,255,255,255,255,254,253,  2037
1470 DATA 248,240,224,193,134,028,120,240,  1427
1480 DATA 000,000,128,000,000,000,000,000,  0128
1490 DATA 007,007,007,007,007,003,003,003,  0044
1500 DATA 252,252,252,254,249,243,229,253,  1984
1510 DATA 011,007,003,005,000,000,000,000,  0026
1520 DATA 128,131,128,141,149,084,104,111,  0976
1530 DATA 254,001,000,003,156,243,013,243,  0913
1540 DATA 123,063,191,223,062,252,248,248,  1410
1550 DATA 231,216,224,000,156,143,138,128,  1236
1560 DATA 128,112,008,000,048,240,168,000,  0704
1570 DATA 003,003,003,001,001,002,003,004,  0020
1580 DATA 249,254,254,135,012,203,057,140,  1304
1590 DATA 111,127,094,056,032,032,032,016,  0500
1600 DATA 231,151,023,047,047,047,095,095,  0736
1610 DATA 248,248,232,200,136,008,008,048,  1128
1620 DATA 128,128,128,128,128,128,064,032,  0864
1630 DATA 004,004,004,004,009,018,016,016,  0075
1640 DATA 122,018,020,148,080,080,080,160,  0708
1650 DATA 016,016,016,016,017,017,057,058,  0213
1660 DATA 094,190,188,184,120,112,112,224,  1224
1670 DATA 096,128,152,104,020,002,001,004,  0507
1680 DATA 024,004,052,072,176,000,000,064,  0392
1690 DATA 008,008,016,024,031,031,063,063,  0244
1700 DATA 032,032,064,192,192,192,192,192,  1088
1710 DATA 062,062,126,126,255,255,255,255,  1396
1720 DATA 224,224,192,128,129,007,131,065,  1100
1730 DATA 004,000,028,255,255,208,255,255,  1260
1740 DATA 064,000,112,252,254,039,255,254,  1230
1750 DATA 000,000,000,000,000,128,001,002,  0131
1760 DATA 127,127,127,255,255,255,255,255,  1656
1770 DATA 192,224,240,248,248,248,240,240,  1880
1780 DATA 001,001,001,003,007,007,007,000,  0027
1790 DATA 254,253,251,246,244,204,048,000,  1500
```

```
1800 DATA 224,080,080,072,072,068,066,065, 0727
1810 DATA 255,031,000,000,000,000,000,000, 0286
1820 DATA 252,224,000,000,000,000,000,001, 0477
1830 DATA 004,008,016,016,032,032,064,128, 0300
1840 DATA 255,255,255,255,255,255,255,254, 2039
1850 DATA 240,240,248,252,252,216,128,000, 1576
1860 DATA 000,000,000,000,000,000,000,001, 0001
1870 DATA 000,001,002,005,010,018,242,004, 0282
1880 DATA 193,063,032,032,016,016,016,008, 0376
1890 DATA 007,248,001,002,000,000,000,000, 0258
1900 DATA 000,128,000,000,000,000,000,000, 0128
1910 DATA 254,254,253,198,184,144,140,130, 1557
1920 DATA 000,000,000,000,063,064,128,000, 0255
1930 DATA 000,000,032,095,128,000,000,000, 0255
1940 DATA 002,004,008,190,004,002,001,000, 0211
1950 DATA 004,004,008,016,016,000,128,064, 0240
1960 DATA 129,064,000,000,000,000,000,000, 0193
1970 DATA 000,128,064,032,016,008,015,016, 0279
1980 DATA 000,000,000,000,000,000,254,001, 0255
1990 DATA 032,024,007,000,000,000,000,000, 0063
2000 DATA 128,000,224,000,000,000,000,000, 0352
2010 DATA 003,014,020,016,124,002,002,004, 0185
2020 DATA 224,063,000,000,063,064,064,064, 0542
2030 DATA 000,227,020,024,224,000,000,000, 0495
2040 DATA 224,000,000,000,000,000,000,000, 0224
2050 DATA 128,064,000,000,000,000,000,000, 0192
2060 DATA 004,003,000,000,000,000,000,000, 0007
2070 DATA 064,128,000,000,000,000,000,000, 0192
2080 REM *** END OF DATA ***
```

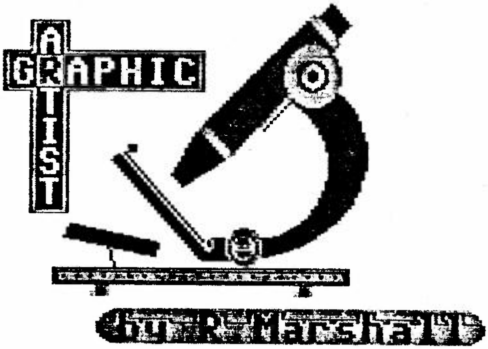
Help Wanted

Dear Roy

Could you please explain how to 'CURSET' on the C16, as the ORIC uses this command a lot, thanks.

Peter Appleby, NOTTS.

Well Peter here goes, I take it CURSET standards for CURSOR SET, ie, puting the cursor to a LOCATIon of the screen, well I'm not 100% sure, but I think you can use the LOCATE column,row, ie, LOCATE 10,10 locates the pixel position of 10 columns across and 10 rows down. As I say I'm not sure thats correct, so if any member knows how to help with this problem then please write in and HELP!!!


GRAPHIC
ARTIST
by R Marshall

Graphic Editor.

The purpose of this program is to allow you to alter any picture on the graphic screen and see every byte in detail, including the color of every pair of dots on the Graphic 3, multicolor screen.

Typing it in:

Certain graphics may give you some problems so I have listed here what you need to type.

Lines 230 and 240: L%=0 graphic is COMMODORE key with T
L%=1 COMMODORE key with Y
L%=2 COMMODORE key with U
L%=3 Shift key with D
L%=4 Shift key with F
L%=5 COMMODORE key with O
L%=6 COMMODORE key with P
L%=7 COMMODORE key with @
Line 360 graphic used is Shift with M
Line 370 Shift wth Q
Line 380 Shift with W
Lines 410 and 630 Shift with X
Line 460 Shift and V

It is sometimes vital that the number of spaces used within quotes are correct.
Line 250 contains two spaces together
Line 800 one
Line 850 two
Line 720 has three spaces then two spaces
Lines 940, 950, 970, 980, 1370, 1380, 1400 and 1410 all have eight spaces in a row.

Beating computer system problems.

The program does not use the drawing programs in the computer so it was necessasry to determine where work is being done on the screen. Line 250 puts the information on the screen in place of a portion of the screen graphic. This section of graphic is first saved using SSHAPE. The same thing happens when large dot representations of the graphics byte appear when V or M are used ('value' and 'memory').

If you have ever been confused by Color 2 and Color 0 then the way this program keeps a register of the colors in use on the screen will interest you. In next month's magazine there will another program to use with 'Graphic Editor' which will solidly lock into the computer Colors 0,1 and 2, but at the moment just let me say that the background Color 0 on the Graphic 3 screen will not necessarily be the background color on Graphic 1. Color 2 on Graphic 3 will, however, definitely be the background color if you switch to Graphic 1.

The next point concerns RCLR. In Immediate Mode, if you make Color 0 light green with luminance of say 5 and then type PRINTRCLR(0). What answer does the computer try to palm you off with? - 0? - exactly, when the answer should be 16. That is why you see +(16ANDRCLR(0)=0) etc in the listing.

```
10 REM ************************************
20 REM *        GRAPHIC  EDITOR           *
30 REM *      BASIC BY ROB MARSHALL       *
40 REM *  MACHINE CODE BY PETER CRACK     *
50 REM *            19.8.90               *
60 REM ************************************
70 KEY1,"(CLR)(WHT)COLOR0,3,0:GRAPHIC0"+CHR$(13):KEY4,"MONITOR"+CHR$(13)
80 DEFFNY(A)=INT((SN%)/320)
90 DEFFNX(A)=INT((SN%-(FNY(A)*320))/8)
100 FORI=0TO58:READA:POKE1600+I,A:NEXTI
110 FORI=0TO30:READA:POKE1664+I,A:NEXTI
120 CD$="(CLR)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)
(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)":GRAPHIC0:IFPEEK(765)=1THEN160
130 POKE760,RCLR(0)+(16ANDRCLR(0)=0):POKE761,RLUM(0)
140 POKE758,RCLR(1)+(16ANDRCLR(1)=0):POKE759,RLUM(1)
150 POKE763,RCLR(0)+(16ANDRCLR(0)=0):POKE764,RLUM(0):POKE765,1
160 COLOR0,3,0:COLOR1,2:PRINT"(CLR)(DOWN)(RGHT)(RGHT)(RGHT)(RGHT)WHEN ON THE GRA
PHIC SCREEN,"
170 PRINT"(RGHT)(RGHT)(RGHT)(RGHT)PRESS I FOR INSTRUCTIONS"
180 DO:INPUT"(DOWN)(DOWN)(RGHT)(RGHT)(RGHT)(RGHT)GRAPHIC 1 OR 3";G%:LOOPUNTILG%=
1ORG%=3
190 DO:INPUT"(DOWN)(RGHT)(RGHT)(RGHT)(RGHT)POSITION OF DATA (0-32)";D%:LOOPUNTIL
D%>-1ANDD%<33
200 TRAP1360:GRAPHICG%:GOSUB1300
210 GD%=8/(INT(G%/2)+1):DT%=D%*GD%:DB%=(D%+8)*GD%:SSHAPES1$,DT%,0,DB%,7
220 S%=10752+L%+R%:I%=PEEK(S%):SN%=S%-8192
230 IFL%=0THENL$="‾":ELSEIFL%=1THENL$="¯":ELSEIFL%=2THENL$="━":ELSEIFL%=3THENL$=
"—"
240 IFL%=4THENL$="–":ELSEIFL%=5THENL$="▬":ELSEIFL%=6THENL$="_":ELSEIFL%=7THENL$=
"_"
250 H$=RIGHT$(STR$(FNX(A)),2)+RIGHT$(STR$(FNY(A)),2):C$=MID$(STR$(I%)+"  ",2,3)
260 C%=1:IFRLUM(0)<5ORRCLR(0)=1THENC%=2
270 IFI%<>V%ANDWB%=1THENGOSUB930:WB%=0
280 COLOR1,C%:CHAR,D%,0,H$+L$+C$:GETB$:IFB$=""THEN280
290 IFB$="*"THENPOKES%,0
300 IFB$=CHR$(20)THENI%=I%-1:GOTO910
310 IFB$=CHR$(13)THENI%=I%+1:GOTO910
320 IFB$=CHR$(29)ANDFNX(A)<39THENT%=8:GOTO520:REM CURSOR RIGHT
330 IFB$=CHR$(157)ANDFNX(A)>0THENT%=-8:GOTO520:REM CURSOR LEFT
340 IFB$=CHR$(145)THENT%=-320:GOTO520:REM CURSOR UP
350 IFB$=CHR$(17)THENT%=320:GOTO520:REM CURSOR DOWN
360 IFB$="╲"THENM%=PEEK(S%):GOTO710:REM LOAD MEMORY
370 IFB$="●"THENPRINTCD$:GRAPHICG%+1:GOSUB990
380 IFB$="O"THENPRINTCD$:GRAPHICG%+1:GOSUB1010
390 IFB$="M"THENPOKES%,M%:GOTO220:REM MEMORY
400 IFB$="*"THENPOKES%,170
410 IFB$="♣"ANDG%=3THEN610:REM LOAD EXCHANGE A COLOR
420 IFB$="X"THENE%=1:GOTO600:REM EXCHANGE A COLOR
430 IFB$="—"ANDG%=3THENGOSUB860:REM CHANGE COLOR 0 OR 3 OR 4
440 IFB$="R"THENI%=255-I%:POKES%,I%:REM REVERSE FIELD
450 IFB$="*"THENPOKES%,85
460 IFB$="╳"THEN650:REM LOAD VALUE
470 IFB$="V"THENPOKES%,V%:GOTO220:REM VALUE
480 IFB$="I"THEN1030
490 IFB$="*"THENPOKES%,255
500 IFB$="+"THENL%=L%+1:IFL%>7THENL%=0:T%=320
```

```
 510 IFB$="-"THENL%=L%-1:IFL%<0THENL%=7:T%=-320
 520 IFS%+T%>16191THENL%=7:T%=0:GOTO220
 530 IFS%+T%-L%<8185THENL%=0:T%=0:GOTO220
 540 IFB$="*"THENPOKES%,I%
 550 IFB$=CHR$(27)ANDG%=3THENGOSUB930:GOTO610
 560 R%=R%+T%:T%=0:GOTO220
 570 GETKEYA$:IFA$=CHR$(27)ANDG%=3THEN610
 580 COLOR0,3,0:GRAPHIC0:PRINT"(WHT)(CLR)":IFA$="R"THEN160
 590 END
 600 POKE1016,DEC(LEFT$(HEX$(S%),2)):POKE1015,DEC(RIGHT$(HEX$(S%),2)):SYS1600:GOT
0220
 610 PRINTCD$:GRAPHIC4:DO:INPUT"COLOR TO BE CHANGED";FC%:LOOPUNTILFC%>-1ANDFC%<4
 620 DO:INPUT"NEW COLOR";SC%:LOOPUNTILSC%>-1ANDSC%<4
 630 GRAPHIC3:POKE1017,FC%:POKE1018,SC%:IFB$="♣"THEN220
 640 SYS1664:GOTO570
 650 COLOR1,RCLR(0)+(16ANDRCLR(0)=0),RLUM(0):CHAR,D%+5,0,"    ",1:COLOR1,C%:V$=""
 660 IFWB%=0THENSSHAPES2$,DT%,8,DB%,15
 670 FORX=0TO2:X$="":GETKEYX$:IFPEEK(198)=1THEN690
 680 V$=LEFT$(V$+X$,3):CHAR,D%+5,0,V$:NEXTX
 690 V%=VAL(V$):IFV%<0ORV%>255THEN650
 700 POKES%,V%:Z%=1:E%=3:GOTO730
 710 IFWA%=0THENSSHAPES3$,DT%,184,DB%,191:SSHAPES4$,DT%,192,DB%,199
 720 CHAR,D%,23,"    "+C$+"   ":Z%=24:E%=2:WA%=1
 730 A%=PEEK(S%):F%=INT(1/G%):FORX=0TO6+F%STEP2-F%
 740 B1%=SGN(A%AND2↑X):IFG%=1THENGOSUB790:GOTO770
 750 B2%=SGN(A%AND2↑(X+1))
 760 IFE%=2ORE%=3THENGOSUB810:REM VALUE OR MEM
 770 NEXTX:IFE%=3THENWB%=1
 780 GOTO220
 790 COLOR1,PEEK(760),PEEK(761):IFB1%=1THENCOLOR1,PEEK(758),PEEK(759)
 800 CHAR,D%+(7-X),Z%," ",1:RETURN
 810 IFB1%=0ANDB2%=0THENCOLOR1,RCLR(0)+(16ANDRCLR(0)=0),RLUM(0)
 820 IFB1%=1ANDB2%=0THENCOLOR1,PEEK(758),PEEK(759)
 830 IFB1%=0ANDB2%=1THENCOLOR1,PEEK(760),PEEK(761)
 840 IFB1%=1ANDB2%=1THENCOLOR1,RCLR(3)+(16ANDRCLR(3)=0),RLUM(3)
 850 CHAR,D%+(6-X),Z%,"  ",1:RETURN
 860 PRINTCD$:GRAPHIC4:DO:INPUT"COLOR 0,3 OR 4";CS%:LOOPUNTILCS%=0ORCS%=3ORCS%=4
 870 DO:INPUT"COLOR (1 TO 16)";K%:LOOPUNTILK%>0ANDK%<17
 880 DO:INPUT"LUMINANCE (0 TO 7)";LU%:LOOPUNTILLU%>-1ANDLU%<8
 890 COLORCS%,K%,LU%:IFCS%=0THENPOKE763,K%:POKE764,LU%
 900 GRAPHIC3:RETURN
 910 IFI%<0THENI%=256+I%:ELSEIFI%>255THENI%=I%-256
 920 POKES%,I%:GOTO220
 930 COLOR1,PEEK(758),PEEK(759):COLORG%-1,PEEK(760),PEEK(761)
 940 IFWB%=1THENCHAR,D%,1,"            ",1:GSHAPES2$,DT%,8:WB%=0
 950 IFB$=CHR$(27)THENCHAR,D%,0,"            ",1:GSHAPES1$,DT%,0
 960 IFWA%=0THENRETURN
 970 CHAR,D%,23,"            ",1:GSHAPES3$,DT%,184
 980 CHAR,D%,24,"            ",1:GSHAPES4$,DT%,192:WA%=0:RETURN
 990 INPUT"(DOWN)WHICH DOT TO REMOVE (0-7)";DP%:IFDP%<0ORDP%>7THEN990
1000 POKES%,I%-(I%AND2↑DP%):GRAPHICG%:RETURN
1010 INPUT"(DOWN)WHICH DOT TO ADD (0-7)";DP%:IFDP%<0ORDP%>7THEN1010
1020 POKES%,I%OR2↑DP%:GRAPHICG%:RETURN
1030 PRINT"(CLR)":COLOR0,3,0:GRAPHIC0:PRINT"(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)
(RGHT)(RGHT)(RGHT)(WHT)MEANING OF KEY PRESSES"
1040 PRINT"(LBLU)(DOWN)(RGHT)(RVS)CURSOR KEYS(OFF)(YEL)(RGHT)ALLOW MOVEMENT BETW
```

```
EEN"
1050 PRINT"(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)CHARACTER POSITIONS"
1060 PRINT"(LBLU)(RGHT)(RVS) + (OFF)(ORNG)(RGHT)MOVES DOWN INDIVIDUAL LINES"
1070 PRINT"(LBLU)(RGHT)(RVS) - (OFF)(YEL)(RGHT)MOVES UP INDIVIDUAL LINES"
1080 PRINT"(LBLU)(RGHT)(RVS)SHIFT M (OFF)(ORNG)(RGHT)MEMORIZES THE VALUE OF THE(
RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)CURRENT LOCATION"
1090 PRINT"(LBLU)(RGHT)(RVS) M (OFF)(YEL)(RGHT)POKES THE MEMORIZED VALUE INTO TH
E(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)CURRENT LOCATION"
1100 PRINT"(LBLU)(RGHT)(RVS)SHIFT C (OFF)(ORNG)(RGHT)ALLOWS A COLOR CHANGE OF"
1110 PRINT"(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)COLOR0,3 OR 4 ON GRAPHIC3 SCREEN"
1120 PRINT"(LBLU)(RGHT)(RVS)SHIFT X (OFF)(YEL)(RGHT)PROGRAMS A COLOR EXCHANGE ON
(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)GRAPHIC3 SCREEN"
1130 PRINT"(LBLU)(RGHT)(RVS) X (OFF)(ORNG)(RGHT)CAUSES A ONE BYTE PRE-PROGRAMMED
"

1140 PRINT"(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)COLOR EXCHANGE ON GRAPHIC3 SCREEN"
1150 PRINT"(LBLU)(RGHT)(RVS)SHIFT V (OFF)(YEL),GIVE THE CURRENT LOCATION A"
1160 PRINT"(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)VALUE USING THE KEYBOARD NUMBERS"
1170 PRINT"(WHT)(DOWN)(DOWN)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)PRES
S A KEY FOR NEXT PAGE":GETKEYA$
1180 PRINT"(LBLU)(CLR)(DOWN)(DOWN)(RGHT)(RVS)SHIFT Q (OFF)(ORNG),ELIMINATE A DOT
 BY ITS 0 TO 7(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)POSITION IN THE BYTE"
1190 PRINT"(LBLU)(RGHT)(RVS)SHIFT W (OFF)(YEL),ADD A DOT BY ITS 0 TO 7(RGHT)(RGH
T)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)POSITION IN THE BY
TE"
1200 PRINT"(LBLU)(RGHT)(RVS)RETURN(OFF)(ORNG)(RGHT)INCREASES THE VALUE AT THE(RG
HT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)CURRENT LOCATION
BY 1"
1210 PRINT"(LBLU)(RGHT)(RVS)INST DEL(OFF)(YEL)(RGHT)REDUCES THE VALUE AT THE(RGH
T)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)CURRENT LOCATION B
Y 1"
1220 PRINT"(LBLU)(RGHT)(RVS) * (OFF)(ORNG)(RGHT)FINDS YOUR PLACE"
1230 PRINT"(LBLU)(RGHT)(RVS)ESC(OFF)(YEL)(RGHT)STOPS THE PROGRAM AND CHANGES WHO
LE"
1240 PRINT"(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)SCREEN COLOR DOT PATTERNS"
1250 PRINT"(LBLU)(RGHT)(RVS)FUNCTION KEY 1(OFF)(ORNG)(RGHT)CHANGES THE SCREEN (R
GHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)TO THE TEXT SCREEN"
1260 PRINT"(LBLU)(RGHT)(RVS)FUNCTION KEY 4(OFF)(YEL)(RGHT)MOVES TO MONITOR"
1270 PRINT"(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)TO ALLOW ARTWORK TO BE SAVED"
1280 PRINT"(WHT)(DOWN)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)PRES
S A KEY TO RETURN"
1290 PRINT"(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)(RGHT)TO
GRAPHIC SCREEN":GETKEYA$:GOSUB1300:GRAPHICG%:GOTO220
1300 F%-INT(G%-(2/G%))+1:COLOR0,PEEK(760+F%),PEEK(761+F%):RETURN
1310 DATA36,131,48,1,96,120,160,0,173,248,3,174,247,3,32,84,6,88,96,234,133,35
1320 DATA134,34,162,3,177,34,72,41,3,205,249,3,208,3,173,250,3,74
1330 DATA110,251,3,74,110,251,3,104,74,74,202,16,231,173,251,3,145,34,96
1340 DATA36,131,48,1,96,120,162,32,160,0,134,35,132,34,138,72,32,88,6,200
1350 DATA208,250,230,35,104,170,202,208,241,88,96
1360 GRAPHICG%:COLOR1,PEEK(758),PEEK(759):COLORG%-1,PEEK(760),PEEK(761)
1370 IFWB%-1THENCHAR,D%,1,"        ",1:GSHAPES2$,DT%,8:WB%-0
1380 CHAR,D%,0,"        ",1:GSHAPES1$,DT%,0
1390 IFWA%-0THEN1420
1400 CHAR,D%,23,"        ",1:GSHAPES3$,DT%,184
1410 CHAR,D%,24,"        ",1:GSHAPES4$,DT%,192:WA%-0
1420 GETKEYA$:GOTO580
```

Using GRAPHIC EDITOR by Rob Marshall

The program, under your direction, decides which memory location on the graphic screen is to be communicated with. This I call the 'current location'. The pixel cursor is not the same thing and functions using pixel cursor parameters are not useable in this program.

There are many features to the program which are accessed by pressing the appropriate keys on the keyboard:-

* flashes the current location for you to find your place.

I produces two pages of instructions while you are using the program.

The cursor keys behave normally.

Shift and M memorizes the value of the current location

M puts the value previously stored with shift and M into the current location.

Shift and V allows you to enter, via the keyboard numbers, the value you want the currect location to be and memorizes it.

V puts the value previously stored using shift and V into the current location.

- moves us up the screen by one dot.

+ moves us down the screen by one dot.

Return increases the value in the current location by 1.

Inst Del reduces the value in the current location by 1.

Shift and Q is the gateway to deleting one chosen dot using the numbers 0 to 7. 0 is the right most dot and 7 is the left most dot in the current location.

Similarly, shift and W adds a single dot on the screen.

If you are using the Graphic 3 screen then you can change dot patterns which ofcourse means that the colors would change. Any location can be treated to a color change using shift and X to program the change then press x and wherever you are on the screen, that change you have ordered will be activated. To issue new orders simply press shift and X and reprogram.

Also on the Graphic 3 screen you can choose the actual values of color and luminance used for Color 0 and Color 3. Press shift and C to reprogram these colors.

Whole screen color changes can be achieved using Esc. This is usually used to eliminate colors (make a color the same as Color 0) to prepare artwork for printing. Never use this without first saving your original artwork.

To stop the program press Stop and watch the result. The data is removed from the screen leaving only your artwork ..... assuming you have written the program correctly !!!

There are some other little facets to this program and you can always modify it to your own design, but get it going first and write to me with your suggested alterations.

Now you are left with the problem of what to do, what to draw. Well, as the saying has it, I wouldn't have started from here! The best thing is to firstly write a drawing program. Save the program. Run the program. Save the picture as a block of memory (you will be in MONITOR and save the picture with S'My picture 1',8,2000,3F40). This can the be loaded back from disk either as a BASIC program or as a block of memory. Once you use GRAPHIC EDITOR, all subsequent savings will only be as blocks of memory. Tape users, buy a disk drive, I recommend the 1581 3.5 inch disk drive.

Mr Roy Robinson,
112 Cliff Road,
Hornsea,
N. Humberside, HU18 1JE.

89 Old Road,
East Peckham
Tonbridge,
Kent, TN12 5EN.

29th October 1990

Dear Roy,

I am pleased to be able to make available to any club member the 'Programmer's Reference Guide for the Commodore Plus/4' by Cyndie Merten & Sarah Meyer for £7.00 including postage.

I have been using this reference book for two years and have found it invaluable and now it is available from one of my electronic components suppliers.

I would like to suggest and addition to the book for any one who is interested. On page 189 there is a table of data about the color keys. I suggest adding two columns to the table indicating the normal values of the keys:-

	Location	Value
Black	275	0
White	276	113
Red	277	50
Cyan	278	99
Purple	279	68
Dark green	280	53
Blue	281	70
Yellow	282	119
Orange	283	72
Dark brown	284	41
Yellow-green	285	90
Pink	286	107
Blue-green	287	92
Light blue	288	109
Dark blue	289	46
Light green	290	95

Any color key can be made to be any color and shade. For example, to make the 'black' key to be grey (= white, luminance level 4) POKE 275,66. Every time that the black key is used in print statements after using poke 275,66 the result will be in grey instead of black. If you wanted to, you could have all eight levels of one colour distributed to eight of the colour keys.

On the subject of books and pokes, if you have 'Using the Commodore 16' by Peter Gerrard, he talks on page 14 about using POKE 806,103 to disable the stop key. Disabling the stop key is essential to prevent accidentally interrupting a data base program which would loose all the work you are currently doing but POKE806,103 has serious consequences such as file may not be saved so some research has produced POKE 806,120:POKE 807,227. This redirects the stop routine to turn the screen on - which it probably already was any way instead of stopping the program.

One other useful tip I have concerns using custom characters (your own typeface) on the graphic screen using PRINT or CHAR commands. Location 740 holds the key to the character set used. 740 normally contains 208 which means $D000 (53248) the upper case / graphics set. If you parked your own character set at $6000 (24576) and then put POKE 740,96 your character set would be in use for PRINT and CHAR.

From the same source as I can get the Reference Guide I can also get parts for Commodore machines. The print head for the MPS801 was recently asked for in the magazine and I could provide it for £50.

Yours sincerely

Rob Marshall

Rob

For Sale:

COMMODORE PLUS/4 COMPUTER
COMMODORE GREEN SCREEN MONITOR
COMMODORE DATASETTE
COMMODORE PLUS/4 PSU
TROJAN LIGHT PEN
2 x CHEETAH STAR PROBE JOYSTICKS (WITH ADAPTORS FOR BOTH)
80+ TOP SOFTWARE TITLES

£149.99 (separately all items would cost over £350)

Everything is in perfect working order with all leads. All 99% of software has original instructions.

If you would like to buy all/anything (apart from software), get software list, ask anything or anything you want write to:

Matthew Newton-Lewis,
24, Jubilee Road,
Chichester,
West Sussex,
PO19 1XJ.

Or ring: Chichester (0243) 782176 after 7pm.

All offers are welcome, don't be afraid to ask!

NEW

Mr T.M.SEXTON
17,Grove Road North
Portsmouth
HANTS
PO5 1JJ
0705-823470

13/9/90

Membership No:-

Dear Roy,

Sorry I have'nt been in touch for a while but as you can see from above I have been moving house (thank god its over) anyway the good part about it is now ive got a room to myself so that the trusty +4 is set up all the time now, plus one or two more 'toys' I have aquired so along with the disk drive I now have an MPS801 Printer a Datachat Modem and a subscription to Prestel and Micronet so as you can imagine I am kept quite busy (the Wife is'nt too pleased though she hardly ever sees me).

Right down to business, Have you heard about the new Commodore Magazine thats being published ? its called 'COMMODORE FORMAT' and its published by FUTURE PUBLISHING should hit the shops today (13/9) from what I can gather its aimed at the C64 but Ive written to the Editor Steve Jarret and asked him whether there will be any features etc for the +4. also EMAP PUBLISHING are ~~are~~ bringing out a qaurterly magazine called 'THE COMPLETE GUIDE TO THE 64' which will be written by ex Zzap journalists so perhaps there will be somthing of interest there.

I was on the phone to Mark Everingham the other day (he of YOUR COMMODORE fame) about his 'PSYCLONE' tape backup disc which I have just recieved (which was reviewed in your magazine (v2 iss2) by David Cambell I belive. He said that he had'nt forgotten about +4 users but as he now has an 'AMIGA!!' that came first but he is half way through an arts package for the +4 that he will try to get finished, so perhaps a letter or two from members would convince him that we are still very much alive and kicking out here.

Down to Personel matters, can I place an advert in the mag please Roy

'WANTED'
A SCRIPT-PLUS CARTRIDGE FOR MY +4.
ALSO ANY UTILITIES (ADD-ONS ETC)
Please write with prices etc to:-
TONY SEXTON, 17,Grove Road North
Portsmouth
HANTS
or Mailbox me on Prestel 705823470
In fact if you have access to Prestel
Mailbox me anyway and we'll have a chat.

Cheers for that Roy, well I think thats all, keep up the good work, thanks as always for supporting our needs if I here anything else Ill let you know, All the best for now

Yours faithfully

TONY SEXTON

P.S. Could you send the mag to the above address in future - ta ta

THIS MAGAZINE NEEDS YOUR ARTICLES!!

PROGRAMS

REVIEWS

SO SEND THEM TO:-

ROY ROBINSON, 112 CLIFF ROAD, HORNSEA N. HUMBERSIDE, HU18 1JE.

TELEPHONE: 0964-534611 after 6pm but not on Friday Night.

OR ELSE!!!

IN FACT ANYTHING TO DO WITH THE C16/C116/+4 IS MOST WELCOME..

COMMODORE 16

BASEMENT BIN

COMMODORE 16 POKES

FINGERS MALONE

KNOCK-OUT

WHO DARES WINS II

PANIK

AIRWOLF II

SPACE FREEKS

MANIC DEATH CHASE

OUT ON A LIMB (PLUS 4 ONLY)

Further Details and Prices

Philip Saunders
90 Oakbank
Fieldway
New Addington
Croydon

SURREY
CR0 9EF

Cheques & PO's made payable to
MR PHILIP SAUNDERS.

IMPORTANT NOTICE

ON BASEMENT BIN

TO ALL MEMBERS CAN YOU PLEASE NOTE THE DISCOUNT SCHEME DOES NOT APPLY TO THE BASEMENT BIN.

PRICES ON COMMODORE 16 POKES

16 Pence Each

8 Pokes for £1.10

16 Pokes for £2.00

PRICES ON COMPUTER POSTERS

20 Pence Each

10 Posters for £1.50

20 Posters for £2.00

30 Posters for £2.50

Or Finally The Whole Collection for £3.00

PLEASE ADD ANOTHER 20% TO HELP COVER THE COST OF POSTAGE & PACKING

COMMODORE 16
BASEMENT BIN

COMPUTER POSTERS

PLAY IT AGAIN SAM 5

REPTON THRU TIME

MAGNETRON

TEST DRIVE

SINCLAIR 128

ROBBERY

MARIES CHRISTMAS BOX

BMX SIMULATOR PLUS

OPERTION H

PLAYERS PREMIER

SKY

ACE 1

RING WARS

CASCANDIA

ACE II

IMPLOSION

UMS

JINXTER

XENON

BOBSLEIGH

TOMHAWKER

MORPHEUS

DAN DARE II

JET SKI SIMULATOR

COMMODORE 16
BASEMENT BIN

COMPUTER POSTERS

MOTOR MADNESS

ENDZONE

TRAP DOOR

EAGLES NEST

SUMMER EVENTS

EXPLODING FIST

ECHELON

THE GAMES

BBC MIRCO

STAR GLIDER

IRIDIS

JOE BLADE II

WHIRLIGG

BIGGLES

EARTHLIGHT

DYNAMIC DUO

REPTON INFINITY

REPTON III

ZARCH

CONQUEROR

PLAY IT AGAIN SAM 6

BY FAIR OR FOUL

MAGNIFICENT SEVEN

LAST NINJA

GAME REVIEW

TITLE: SIR KNIGHT
PUBLISHER: TYNESOFT
PRICE: £3.99 CASSETTE

Sir Knight is one of these rare 'PLUS/4 only games that are available and I would say its the worst'.

You are 'Sir Knight' who has to rescue a Princess imprisoned in a castle (sounds familiar does'nt it?)

Its a platform game that resembles 'Punchy' - an old commodore game that was included with the C16 when it was first launched (remember those 'C16 Starter Pack Adverts?), (Yep I've even got the same BOX, ED).

Where punchy was colourful, playable and addictive - Sir knight is dull, difficult and boring.

The graphics are poor with a dull choice of colours and chunky characters - they are nowhere near the stand of other PLUS/4 games or the majority of 16K ones. The sound is slightly better but still well below average.

As I've mentioned before; Sir Knight is boring and difficult which really lets it downm its also too repetitive.

Published by Tynesoft - famous for Phantom, SPY VS SPY (I've got hold of Spy Vs Spy 3, done by some Hungarian, German Hackers, its good, ED) etc, Sir Knight is a huge disappointment, I find it hard to believe its a 'PLUS/4 only' game. Definitely NOT recommended!

Ratings:

GRAPHICS:	2
SOUND:	3
PLAYABILITY:	2
VALUE:	2
OVERALL:	2

Reviewer: Andy Tang, LONDON

WANTED (URGENTLY!!!)

SCRIPT PLUS, GUNSLINGER and FIRE ANT.

Will pay very good prices...

Any offers? If YES the Write to:-

Matthew Newton_lewis
24 Jubilee Road
Chichester
West Sussex
PO19 1XJ

Or Phone 0243 782176 between 7-9 pm and ask for Matt.

Two Basic Programs
By Wayne Kenzitt

HI PROGRAM

```
10 GRAPHIC 3, 1
20 COLOR 4, 13, 0: COLOR 0, 13, 0
30 COLOR 1, 9, 4
30 BOX 1, 5, 5, 25, 190
50 PAINT 1, 10, 10
60 BOX 1, 50, 5, 70, 190
70 PAINT 1, 55, 10
80 BOX 1, 25, 80, 50, 100
90 PAINT 1, 35, 90
100 BOX 1, 80, 5, 100, 190
110 PAINT 1, 90, 10
120 DRAW 1, 120, 5 TO 150, 5
130 DRAW 1 TO 135, 135
140 DRAW 1 TO 120, 5
150 PAINT 1, 130, 10
160 CIRCLE 1, 135, 165, 12
170 PAINT 1, 135, 165
180 GETKEY A$
190 GRAPHIC 0
200 COLOR 4, 7: COLOR 0, 2
210 COLOR 1, 1
```

COLOUR PYRAMID PROGRAM

```
10 SCNCLR
20 XE=39: Y=22
30 COLOR 4, 1: COLOR 0, 1
40 FOR XS=0 TO 20
50 FOR B=XS TO XE
60 S=INT(RND(1)*4)+3
70 C=INT(RND(1)*15)+2
80 COLOR 1, C, S
90 PRINT CHR$(18);
100 CHAR 1, B, Y, CHR$(160)
110 NEXT B
120 XE=XE-1: Y=Y-1
130 NEXT
```

Send me all you BASIC programs, I don't care what they do just send them.

```
2 REM ****************************
3 REM * GREAT BRITAIN & EIRE MAP *
4 REM * IDEA ADAPTED FROM A      *
5 REM * VIC-20 PROGRAM           *
6 REM ****************************
7 REM * BY KEVIN WHEALS          *
8 REM * (C)1990                  *
9 REM ****************************
10 COLOR0,14:COLOR4,14:COLOR1,6,6
20 POKE55,0:POKE56,60:CLR:POKE1177,62
30 FORI=0TO1023:POKE15360+I,PEEK(53248+I):NEXTI
40 POKE1177,63:POKE65299,60:POKE65298,192:SCNCLR
50 FORL=0TO61:CX=0:FORD=0TO7:READCH:CX=CX+CH:POKE15640+L*8+D,CH:NEXTD
60 READCH:IFCH<>CXTHENPOKE65299,208:POKE65298,196:PRINT"ERROR IN LINE";1000+(L*
0):STOP
70 NEXTL
100 PRINT"(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)"
110 PRINTTAB(19);:PRINT"#$%"
120 PRINTTAB(19);:PRINT"&'()"
130 PRINTTAB(19);:PRINT"*++,"
140 PRINTTAB(19);:PRINT"-+."
150 PRINTTAB(17);:PRINT"/01++2"
160 PRINTTAB(15);:PRINT"345+67+8"
170 PRINTTAB(15);:PRINT"9++:; <+=":REM USE NORMAL SPACE
180 PRINTTAB(15);:PRINT">?+─ ♠│+──":REM USE SHIFT * A B C D  NORMAL SPACE
190 PRINTTAB(15);:PRINT"‾─+│ │╲++╲╱└":REM USE SHIFT E F G H I J K L
200 PRINTTAB(15);:PRINT"╲╱┌ ┐●─++♥│ ":REM USE SHIFT M N O P Q R S T  NORMAL SPAC
210 PRINTTAB(19);:PRINT"╱╳○++♣│":REM USE SHIFT U V W X Y
220 PRINTTAB(18);:PRINT"♦+▒│π◥ ":REM USE SHIFT Z +  CBM -  SHIFT -  CBM = *
221 REM SHIFTED SPACE
300 GOTO300
500 REM *** MAP DATA FOLLOWS ***
1000 DATA 000,003,003,001,001,003,003,007, 0021
1010 DATA 000,131,255,255,255,255,255,255, 1661
1020 DATA 056,248,248,240,240,224,192,000, 1448
1030 DATA 003,015,031,031,015,031,031,007, 0164
1040 DATA 254,252,254,254,252,255,255,255, 2031
1050 DATA 000,000,000,000,119,255,255,255, 0884
1060 DATA 000,000,000,000,000,240,240,224, 0704
1070 DATA 007,015,031,031,127,127,095,031, 0464
1080 DATA 255,255,255,255,255,255,255,255, 2040
1090 DATA 192,192,192,128,128,128,000,000, 0960
1100 DATA 031,015,015,015,031,127,095,031, 0360
1110 DATA 254,224,252,252,240,224,246,255, 1947
1120 DATA 000,000,000,000,000,007,127,255, 0389
1130 DATA 000,000,000,000,000,128,060,254, 0442
1140 DATA 025,049,032,032,032,096,003,003, 0272
1150 DATA 128,224,240,240,248,248,248,248, 1824
1160 DATA 000,000,000,000,000,000,000,007, 0007
1170 DATA 000,000,000,000,000,001,003,255, 0259
1180 DATA 127,255,255,063,063,255,255,255, 1528
1190 DATA 007,007,135,130,066,192,192,192, 0921
1200 DATA 255,255,255,248,064,000,000,000, 1077
1210 DATA 248,252,252,252,254,254,254,255, 2021
1220 DATA 003,007,000,000,003,007,031,015, 0066
1230 DATA 255,255,254,254,248,248,248,248, 2010
1240 DATA 192,128,000,000,000,000,000,000, 0320
```

```
1250 DATA 255,255,255,207,015,031,063,063, 1144
1260 DATA 192,240,248,248,252,252,252,247, 1931
1270 DATA 015,003,000,000,001,001,001,003, 0024
1280 DATA 255,255,063,255,255,255,255,255, 1848
1290 DATA 252,252,252,248,252,252,252,248, 2008
1300 DATA 000,000,192,224,207,191,063,255, 1132
1310 DATA 063,127,127,031,031,191,255,255, 1080
1320 DATA 242,248,254,255,255,255,255,255, 2019
1330 DATA 000,000,000,000,000,128,128,000, 0256
1340 DATA 014,000,015,015,255,031,063,127, 0520
1350 DATA 031,255,255,255,255,255,255,255, 1816
1360 DATA 248,240,240,224,224,224,192,000, 1592
1370 DATA 001,003,000,000,000,000,000,000, 0004
1380 DATA 255,063,063,031,031,063,063,127, 0696
1390 DATA 254,254,255,255,255,255,255,255, 2038
1400 DATA 024,063,191,255,255,255,255,255, 1553
1410 DATA 000,000,192,192,224,192,224,192, 1216
1420 DATA 255,127,095,015,031,001,000,000, 0524
1430 DATA 255,255,255,250,224,000,000,000, 1239
1440 DATA 240,192,128,000,000,000,000,000, 0560
1450 DATA 000,007,031,127,031,062,008,000, 0266
1460 DATA 127,255,255,255,255,255,127,111, 1640
1470 DATA 255,255,255,255,255,255,255,207, 1992
1480 DATA 255,255,255,255,255,254,248,248, 2025
1490 DATA 224,224,192,000,000,000,000,000, 0640
1500 DATA 000,000,000,000,000,003,003,003, 0009
1510 DATA 015,007,000,048,127,255,255,255, 0962
1520 DATA 031,063,063,255,255,255,255,255, 1432
1530 DATA 240,224,000,227,255,252,248,240, 1686
1540 DATA 000,000,000,128,128,000,000,000, 0256
1550 DATA 000,000,000,000,000,000,007,006, 0013
1560 DATA 007,015,063,063,062,240,224,192, 0866
1570 DATA 255,255,255,248,120,048,000,000, 1181
1580 DATA 255,159,007,000,000,000,000,000, 0421
1590 DATA 248,224,142,015,026,000,000,000, 0655
1600 DATA 255,047,000,000,000,000,000,000, 0302
1610 DATA 192,128,000,000,000,000,000,000, 0320
1620 REM *** END OF DATA ***
```

* C16 ONLY ** C16 + 4 ONLY *** +4 ONLY

*	AARDVARK	1.50	**	DIZZY DICE	1.50
*	DARK TOWER	1.50	**	DROID ONE	1.50
*	CUTHBERT ENTERS TOMB OF DOOM	1.00	**	FURY	1.50
**	3D GLOPPER	1.50	**	FIENDS	1.50
**	HUSTLER	1.50	**	GWNN	1.50
**	KANE	1.50	**	HARBOUR ATTACK	1.00
**	LAKTUS	1.00	***	ICICLE WORKS	1.00
**	JACK ATTACK	1.00	**	LEAPER	1.50
**	OBLIDEO	1.50	**	FIRE ANT	1.00
*	OLYMPIC SKIER	1.00	**	PHEENIX	1.50
**	PIRATE ADVENTURE	1.00	*	ROLLER KONG	1.00
*	SHARK	1.50	**	SPORE	1.50
**	SKRAMBLE	1.50	*	SQUIRM	1.50
**	STAR COMMANDER	1.50	**	SOLO	1.50
*	WIZARD AND PRINCESS	1.00	**	STRANG OYDSSEY	2.50
**	ZOLYX	1.50	*	WILLIAMSBURG ADV 3	1.00
**	ZODIAC	1.50	**	XADIUM	1.50
**	MYRIAD	1.50	**	VIDZULLES	2.50

EDUCATIONAL

**	HIDE & SEEK(4-10 YRS)	3.50	***	NUMBER CHASER(5-12 YRS)	3.50

COMPILATIONS

** WINTER EVENTS (BIATHLON,BOBSLED,SKI-JUMP, SLALOM, DOWNHILL, SPEED SKATING) 5.50

*** SIX OF THE BEST (TERRA NOVA,OUT ON A LIMB,KARATE KING, BRIDGEHEAD,JUMP JET,STRIP POKER) 5.50

PLEASE GIVE ALTERNATIVES TO GAMES ORDERED AS STOCKS ARE VERY LOW.

BOOKS

PLUS 4 MAGIC FOR BEGINNERS	(1 ONLY PRICE)	5.00
100 PROGRAMS FOR THE C16	(1 ONLY PRICE)	5.00

C16 + 4 REFERENCE BOOK (SPECIAL PRICE) 3.00

This book lays special emphasis on the graphics, sound, music and machine code programming. The graphics cover multi colour, high resolution bit mapping, sc scrolling and raster interrupt. Its coverage of machine code is extensive and gives full information regarding KERNAL routines and how to use them in your own programs. The book gives complete memory map 7501 processor commands CBMC C16 comparison chart to enable conversion and all the important KERNAL jump table. The subject under discussion is explained lucidly and illustrated by example of line by line explanation.

JOYSTICK - Which includes adaptor

COMPETITION PRO 500	16.50	CRUISER	11.50
QUICK SHOT TURBO	11.50	NAVIGATOR	16.50
KONIX SPEED KING	16.50	EXTERMINATOR	8.00

HARDWARE

C16 DUST COVER	2.50	LEAD-COMPUTER TO 1701 MON.	2.00
PLUS 4 DUST COVER	1.00		
CASS. PLAYER DUST COVER	2.00		
LEAD TO DISC DRIVE	3.00	JOYSTICK ADAPTERS	3.00
MPS 801 RIBBON	4.50	MPS 803 RIBBON	4.50

ALL ITEMS SUBJECT TO AVAILABILTY

PAYMENT

UNITED KINGDOM: By Cheque, Postal Order, Visa, Access or Trust card.
OUTSIDE UK: By cheque drawn on a British Bank, Euro cheque, Giro, Master or Visa card.

POSTAGE

UNITED KINGDOM:	Orders under 5.00	-	50p postage & packing charge
	Orders over 5.00	-	Free
	Books	-	50p postage & packing charge
EUROPE:	Each item despatched	-	1.00
	Books	-	1.50
OUTSIDE EUROPE:	Each item despatched	-	2.50
	Books by Air	-	5.00
	Books by Sea	-	3.00

NAME AND ADDRESS:-

ANCO MARKETING LTD
UNIT 10
BURNHAM TRADING ESTATE
LAWSON ROAD
DARTFORD
KENT
DA1 5BH

TELEPHONE:- DARTFORD (0322) 92513 & 92518

HOURS:- 9.30am - 4pm Monday to Friday

THIS PRICE LIST SUPERCEDES ALL PREVIOUS LISTS

```
2 REM ******************
3 REM * ATOMIC ROSE     *
4 REM * HI-RES PICTURE *
5 REM ******************
6 REM * BY KEVIN WHEALS*
7 REM *                *
8 REM * (C)1990        *
9 REM ******************
10 COLOR0,2:COLOR4,2:COLOR1,1
20 POKE55,0:POKE56,60:CLR:POKE1177,62
30 FORI=0TO1023:POKE15360+I,PEEK(53248+I):NEXTI
40 POKE1177,63:POKE65299,60:POKE65298,192:SCNCLR
50 FORL=0TO61:CX=0:FORD=0TO7:READCH:CX=CX+CH:POKE15640+L*8+D,CH:NEXTD
60 READCH:IFCH<>CXTHENPOKE65299,208:POKE65298,196:PRINT"ERROR IN LINE";1000+(L*1
0):STOP
70 NEXTL
100 PRINT"(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)"
110 PRINTTAB(18);:PRINT"#$%&"
120 PRINTTAB(16);:PRINT"'()*+,-"
130 PRINTTAB(16);:PRINT"./01234"
140 PRINTTAB(16);:PRINT"56789:;<"
150 PRINTTAB(17);:PRINT"=>?─♠│─":REM USE SHIFT * A B C
160 PRINTTAB(16);:PRINT"‾‾─┤ ├╲ ◡":REM USE SHIFT D E F G H I J K
170 PRINTTAB(16);:PRINT"└╲╱┌┐●─♥":REM USE SHIFT L M N O P Q R S
180 PRINTTAB(18);:PRINT"│ ╱╳○♣":REM USE SHIFT T U V W X
190 PRINTTAB(18);:PRINT" │◆":REM USE SHIFT Y Z
200 PRINTTAB(18);:PRINT"┼▒ ":REM USE SHIFT +  CBM -
210 PRINTTAB(17);:PRINT" │π◥":REM USE SHIFT -  CBM = *
220 PRINTTAB(19);:PRINT" ":REM USE SHIFTED SPACE
300 GOTO300
500 REM *** UDG DATA FOLLOWS ***
1000 DATA 000,000,000,000,000,000,000,001, 0001
1010 DATA 001,002,004,008,031,096,128,000, 0270
1020 DATA 128,096,030,001,255,016,008,004, 0538
1030 DATA 000,000,000,000,192,056,004,002, 0254
1040 DATA 000,000,000,000,000,000,015,022, 0037
1050 DATA 003,012,016,032,016,112,192,192, 0575
1060 DATA 254,004,008,008,016,035,036,069, 0430
1070 DATA 000,003,012,048,207,020,126,159, 0575
1080 DATA 002,249,007,252,003,000,028,244, 0785
1090 DATA 001,224,016,008,004,194,033,017, 0497
1100 DATA 000,188,070,118,062,031,031,031, 0531
1110 DATA 018,016,023,023,023,019,011,011, 0144
1120 DATA 192,192,193,193,193,193,225,225, 1606
1130 DATA 074,146,018,036,036,040,040,016, 0406
1140 DATA 043,077,082,098,065,128,128,128, 0749
1150 DATA 254,255,254,126,062,141,096,029, 1217
1160 DATA 137,149,165,165,197,133,197,073, 1216
1170 DATA 031,031,031,031,031,030,030,018, 0233
1180 DATA 009,005,004,002,002,001,000,000, 0023
1190 DATA 241,176,048,248,252,252,254,063, 1534
1200 DATA 032,160,160,160,081,049,017,009, 0668
1210 DATA 128,128,128,064,160,096,016,015, 0735
1220 DATA 002,004,004,008,017,033,066,159, 0293
1230 DATA 073,145,146,146,036,071,248,019, 0884
1240 DATA 145,080,160,160,192,000,000,224, 0961
1250 DATA 000,128,064,032,016,016,016,016, 0288
```

```
1260 DATA 031,028,096,064,032,032,060,094, 0437
1270 DATA 133,067,033,034,028,002,001,000, 0298
1280 DATA 000,000,128,064,032,016,008,135, 0383
1290 DATA 038,088,032,031,000,000,015,048, 0252
1300 DATA 028,032,064,128,000,000,128,064, 0444
1310 DATA 032,032,016,016,032,032,065,131, 0356
1320 DATA 016,032,064,064,128,128,000,000, 0432
1330 DATA 000,000,000,001,001,002,004,009, 0017
1340 DATA 094,158,190,063,127,127,255,255, 1269
1350 DATA 000,000,000,000,000,000,128,128, 0256
1360 DATA 064,032,031,000,000,000,000,000, 0127
1370 DATA 192,128,128,064,063,015,015,007, 0612
1380 DATA 063,000,000,000,193,255,255,255, 1021
1390 DATA 007,015,063,127,255,255,255,253, 1230
1400 DATA 128,128,128,128,128,128,128,128, 1024
1410 DATA 023,019,009,004,002,001,000,000, 0058
1420 DATA 255,255,255,255,127,254,028,003, 1432
1430 DATA 192,224,248,252,227,007,079,191, 1420
1440 DATA 000,000,000,255,255,255,247,243, 1255
1450 DATA 007,007,006,254,206,006,062,255, 0803
1460 DATA 063,127,015,127,063,255,253,228, 1131
1470 DATA 249,233,237,239,163,191,206,124, 1642
1480 DATA 128,192,192,128,000,000,000,000, 0640
1490 DATA 007,007,005,008,017,009,009,019, 0081
1500 DATA 249,220,252,248,208,208,176,240, 1801
1510 DATA 255,127,031,015,003,000,000,000, 0431
1520 DATA 245,215,031,254,252,248,000,000, 1245
1530 DATA 248,240,224,000,000,000,000,000, 0712
1540 DATA 019,021,027,005,004,006,002,002, 0086
1550 DATA 240,240,248,248,120,248,124,124, 1592
1560 DATA 004,005,005,005,005,009,004,120, 0157
1570 DATA 244,246,246,230,252,252,244,228, 1942
1580 DATA 001,001,001,001,001,003,007,015, 0030
1590 DATA 129,061,095,223,159,129,127,128, 1051
1600 DATA 196,252,248,252,254,254,254,254, 1964
1610 DATA 030,006,002,000,000,000,000,000, 0038
1620 REM *** END OF DATA ***
```

```
1 REM *****************
2 REM * DALEK           *
3 REM * HI-RES PICTURE *
4 REM *****************
5 REM * BY KEVIN WHEALS*
6 REM *                *
7 REM *    (C) 1990     *
8 REM *****************
10 COLOR0,2:COLOR1,1:COLOR4,2
20 POKE55,0:POKE56,60:CLR:POKE1177,62
30 FORI=0TO1023:POKE15360+I,PEEK(53248+I):NEXTI
40 POKE1177,63:POKE65299,60:POKE65298,192:SCNCLR
50 FORL=0TO119:CX=0:FORD=0TO7:READCH:CX=CX+CH:POKE15360+L*8+D,CH:NEXTD
60 READCH:IFCH<>CXTHENPOKE65299,208:POKE65298,196:PRINT"ERROR IN LINE";1000+(L*
0):STOP
70 NEXTL
100 PRINT"(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)"
110 PRINTTAB(14);:PRINT"@ABCDEF"
120 PRINTTAB(14);:PRINT"GHIJKLM"
130 PRINTTAB(17);:PRINT"NOPQR"
140 PRINTTAB(14);:PRINT"S  TUVWR":REM USE 2 NORMAL SPACES
150 PRINTTAB(14);:PRINT"XYZ[£]↑←"
160 PRINTTAB(15);:PRINT"!";:PRINTCHR$(34);:PRINT"#$%&'"
170 PRINTTAB(16);:PRINT"()*+,-."
180 PRINTTAB(16);:PRINT"/012345"
190 PRINTTAB(16);:PRINT"6789:;<"
200 PRINTTAB(15);:PRINT"=>?─♠│──":REM USE SHIFT * A B C D
210 PRINTTAB(15);:PRINT"▔─▏ ▕╮ ╰╯└╲":REM USE SHIFT E F G H I J K L M
220 PRINTTAB(14);:PRINT"╱┌▔●─♥▏ ╱╳○♣":REM USE SHIFT N O P Q R S T U V W X
230 PRINTTAB(14);:PRINT" ▏♦┼▒ ▏π◥ ▌▄▔":REM USE SHIFT Y Z +  CBM -  SHIFT -
235 REM CBM = *  SHIFTED SPACE  CBM K I T
240 PRINTTAB(14);:PRINT"┘ ▒ ▙▛ ▏├▪┐":REM USE CBM @ G + M £  SHIFT £  CBM N Q D Z
 S
250 PRINTTAB(14);:PRINT"▂ ┌┴┬┤▏▌ ▛":REM USE CBM P A E R W H J L Y
300 GOTO300
500 REM *** UDG DATA FOLLOWS ***
1000 DATA 000,000,000,000,000,000,001,001, 0002
1010 DATA 000,000,000,120,132,130,050,073, 0505
1020 DATA 000,000,000,000,000,002,005,250, 0257
1030 DATA 000,000,001,000,000,160,081,143, 0385
1040 DATA 000,131,092,160,064,144,048,240, 0879
1050 DATA 000,254,113,004,002,001,001,001, 0376
1060 DATA 000,000,206,050,038,072,088,052, 0506
1070 DATA 001,001,001,000,000,000,000,000, 0003
1080 DATA 073,073,050,130,132,120,000,000, 0578
1090 DATA 010,250,005,002,000,000,000,000, 0267
1100 DATA 168,175,082,164,004,004,004,004, 0605
1110 DATA 016,240,048,050,004,200,000,000, 0558
1120 DATA 000,000,000,000,000,007,004,000, 0011
1130 DATA 130,098,034,033,045,041,073,001, 0455
1140 DATA 003,001,007,008,007,001,007,008, 0042
1150 DATA 255,191,191,000,191,191,191,000, 1210
1160 DATA 255,127,127,000,127,127,127,000, 0890
1170 DATA 254,124,127,000,127,124,127,000, 0883
1180 DATA 000,000,000,128,000,000,000,128, 0256
1190 DATA 000,000,000,000,060,126,239,255, 0680
```

```
1200 DATA 007,001,007,008,007,003,007,009, 0049
1210 DATA 191,191,191,000,191,191,255,140, 1350
1220 DATA 127,127,127,000,127,127,255,099, 0989
1230 DATA 127,124,127,000,127,126,255,024, 0910
1240 DATA 215,223,239,127,126,060,000,000, 0990
1250 DATA 224,030,225,030,001,000,000,000, 0510
1260 DATA 000,000,192,060,195,060,003,000, 0510
1270 DATA 009,009,009,009,137,121,135,120, 0549
1280 DATA 140,140,140,140,140,140,255,248, 1343
1290 DATA 099,099,099,099,099,099,255,003, 0852
1300 DATA 024,024,024,024,024,024,024,255, 0423
1310 DATA 128,128,128,128,128,128,128,128, 1024
1320 DATA 000,000,000,000,000,000,000,000, 0000
1330 DATA 031,008,055,008,031,000,000,000, 0133
1340 DATA 255,000,255,000,255,001,001,000, 0767
1350 DATA 191,124,014,124,152,000,255,192, 1052
1360 DATA 007,248,007,000,000,000,255,000, 0517
1370 DATA 115,251,027,251,115,003,255,035, 1052
1380 DATA 255,255,255,255,255,255,255,255, 2040
1390 DATA 192,224,224,224,224,240,240,240, 1808
1400 DATA 000,000,001,001,003,005,004,004, 0018
1410 DATA 096,255,240,240,240,115,100,101, 1387
1420 DATA 000,255,002,002,002,194,036,036, 0527
1430 DATA 051,255,003,001,001,121,133,149, 0714
1440 DATA 255,254,255,255,255,255,173,140, 1842
1450 DATA 240,144,248,252,252,254,085,070, 1545
1460 DATA 000,000,000,000,000,000,000,128, 0128
1470 DATA 002,003,007,015,015,031,043,035, 0151
1480 DATA 100,228,195,192,192,207,144,148, 1406
1490 DATA 036,036,196,004,008,008,137,137, 0562
1500 DATA 133,133,121,001,001,241,009,041, 0680
1510 DATA 140,156,255,255,255,255,174,142, 1632
1520 DATA 102,238,255,255,255,255,181,049, 1590
1530 DATA 128,128,128,000,000,192,160,160, 0896
1540 DATA 035,019,031,063,063,127,174,142, 0654
1550 DATA 144,144,015,000,000,060,066,082, 0511
1560 DATA 137,137,016,016,016,019,036,036, 0413
1570 DATA 009,009,241,001,003,195,035,163, 0656
1580 DATA 142,158,255,255,255,255,175,143, 1638
1590 DATA 049,115,255,255,255,255,094,030, 1308
1600 DATA 160,224,192,192,224,248,180,052, 1472
1610 DATA 000,000,000,000,000,001,002,002, 0005
1620 DATA 142,078,124,252,252,252,185,057, 1342
1630 DATA 066,066,060,000,000,240,008,072, 0512
1640 DATA 036,036,067,064,064,071,072,073, 0483
1650 DATA 035,035,195,003,003,131,067,067, 0536
1660 DATA 143,159,255,253,252,253,173,141, 1629
1670 DATA 030,062,255,255,255,143,047,015, 1062
1680 DATA 052,120,248,248,254,173,141,141, 1377
1690 DATA 002,001,001,003,003,007,010,008, 0035
1700 DATA 057,057,240,240,240,241,242,242, 1559
1710 DATA 008,008,240,000,000,224,016,144, 0640
1720 DATA 136,136,135,128,128,143,144,146, 1096
1730 DATA 079,079,143,007,007,007,143,143, 0608
1740 DATA 141,156,252,252,252,254,174,142, 1623
1750 DATA 159,255,127,063,127,135,149,133, 1148
1760 DATA 159,254,255,235,227,227,103,031, 1491
1770 DATA 000,000,128,064,064,192,192,128, 0768
1780 DATA 000,000,000,000,000,001,002,004, 0007
```

```
1790 DATA 008,004,007,031,111,159,040,032, 0392
1800 DATA 226,226,193,192,192,195,132,133, 1489
1810 DATA 017,017,225,001,001,193,034,034, 0522
1820 DATA 016,016,015,000,000,030,033,037, 0147
1830 DATA 135,135,007,015,015,015,015,015, 0352
1840 DATA 142,158,254,254,254,254,174,142, 1632
1850 DATA 132,120,000,060,066,074,066,066, 0584
1860 DATA 129,128,142,149,145,081,078,064, 0916
1870 DATA 192,096,080,040,044,043,048,224, 0767
1880 DATA 000,000,000,000,000,000,192,048, 0240
1890 DATA 008,016,032,064,064,064,064,064, 0376
1900 DATA 035,019,031,015,007,003,001,000, 0111
1910 DATA 132,132,003,000,000,000,255,000, 0522
1920 DATA 034,034,194,002,004,004,255,000, 0527
1930 DATA 033,033,030,000,000,000,255,000, 0351
1940 DATA 015,015,015,015,015,015,240,000, 0330
1950 DATA 142,158,254,255,248,128,000,000, 1185
1960 DATA 060,000,031,224,000,000,000,000, 0315
1970 DATA 079,240,000,000,000,000,000,003, 0322
1980 DATA 000,000,000,000,003,012,112,128, 0255
1990 DATA 012,002,014,050,194,002,014,048, 0336
2000 DATA 100,082,073,068,067,096,120,124, 0730
2010 DATA 000,000,000,000,007,240,079,064, 0390
2020 DATA 000,000,000,000,255,000,255,000, 0510
2030 DATA 000,000,000,000,225,000,255,128, 0608
2040 DATA 000,000,000,000,224,001,254,000, 0479
2050 DATA 000,000,000,000,000,255,016,016, 0287
2060 DATA 000,000,000,003,252,000,000,000, 0255
2070 DATA 000,003,029,225,001,001,007,255, 0521
2080 DATA 060,192,000,003,015,127,255,252, 0904
2090 DATA 000,015,127,255,255,252,192,000, 1096
2100 DATA 224,192,192,192,000,000,000,000, 0800
2110 DATA 126,063,031,007,003,000,000,000, 0230
2120 DATA 064,240,255,255,255,063,000,000, 1132
2130 DATA 000,000,128,255,255,255,255,000, 1148
2140 DATA 128,128,128,255,255,255,255,000, 1404
2150 DATA 000,000,000,255,255,255,248,000, 1013
2160 DATA 016,019,255,255,255,240,000,000, 1040
2170 DATA 063,255,255,254,240,000,000,000, 1067
2180 DATA 255,254,224,000,000,000,000,000, 0733
2190 DATA 224,000,000,000,000,000,000,000, 0224
2200 REM *** END ***
```

TO: -

MR ROY ROBINSON

112 CLIFF ROAD
HORNSEA
NORTH HUMBERSIDE
HU18-1JE

Thursday 9th August 1990

Dear Sir,

I reply to your letter, which I recieved yesterday afternoon. I reply as follows : -

I will start on the BASIC PROGRAMMING COURSE sometime next week. I don't know do to it in drip & draps or all at once. Thank-You for your submission for another C16 GAME. I will forward your address again. Just in case he does what to contact you.

I've now decided to give everyone a FREE MYSTERY GIFT with every order recieved before 31st DECEMBER 1990. So if you could please inform your readers of this it would be greatful.

I look forward to hearing about the progress of the BOOK RENTAL & PUBLIC DOMAIN SOFTWARE. Which issue will the information be in ?

I also look forward to recieving your poke details.

With reference to my employment I am currently working and have been for the last 10 months for a firm of ACCOUNTANTS.
My position is COMPUTER OPERATOR / BOOKEEPER this involes installing Software & Hardware, Buying of Software, Operating of Software, also the teaching of other staff. I will not start boring you with the accounts side. If you request more information you might like to have a copy of my C.V which is avaiable on request.

Yours faithfully

MR PHILIP SAUNDERS
C16 POKES

FROM : -

C16 POKES
90 OAKBANK
FIELDWAY
NEW ADDINGTON
CROYDON
SURREY
CR0-9EF

Sunday 9th September 1990

Dear Sir,

Just a short letter to inform you that the <u>COMPUTER COURSE</u> is now drafted up. It will be typed up and posted to Mr Roy Robinson. C16 MONTHLY. 112 Cliff Road. Hornsea. North Humberside. HU18-1JE within 21 days.

We have now decided to offer our members the following services:

Photocopying
Typing Services
Computer Labelling
C.V

Further details are enclosed. We look forward to your views and comments as always. which should be send to the address at the top of this page.

Yours faithfully

MR PHILIP SAUNDERS
C16 POKES

MORE SERVICES FROM C16 POKES

PHOTOCOPYING

1 - 5 PAGES	-	5p Each + S.A.E
6 - 10 PAGES	-	4p Each + S.A.E
OVER 10 PAGES	-	3p Each + S.A.E

TYPING

75p Per Computer Page

COMPUTER LABELLING

1 - 5 Lables	-	4p Each + S.A.E
5 - 10 Labels	-	3p Each + S.A.E
OVER 10 Labels	-	2p Each + S.A.E

C.V

We will quite happily prepare your C.V for when you apply for that most important job. From special devised sheets we will produce a four page computerised C.V. Made up in to a binder to help give it that professional look. All this for a small fee of £5.00. Samples are now avaiable for the even smaller fee of £1.50 including Printing, Postage & Packing.

CHEQUES ARE MADE PAYABLE TO MR PHILIP SAUNDERS