ENGINEERING NOTES
on
Radio Shack Color Computers

October 1985
Vol. 2 No. 9

$1.95

DYNAMIC COLOR NEWS

* BASIC + REVIEW
* WORD PROCESSING PROGRAM
* BAR GRAPH PROGRAM
* USING PAGE -1
* ADD A SECOND PORT

* LARGE MEMORY PROGRAMS (Part 9)
* WRITING PROGRAMS (Part 8)
* OPERATING HINTS
* COMPUTER GRAPHICS (Part 9)
* NEW PRODUCTS
* PRODUCT REVIEWS

DYNAMIC COLOR NEWS is published monthly by DYNAMIC ELECTRONICS, INC., P.O. Box 896, Hartselle, AL 35640, phone (205) 773-2758. Bill Chapple, President; Alene Chapple, Sec. & Treas.; John Pearson, Ph. D. Consultant; Bob Morgan, Ph. D., Consultant.

 Subscriptions are $15/yr for U.S.A. & Canada, $30 other foreign.

The purpose of this newsletter is to provide instruction on Basic & Machine Language programming, Computer theory, operating techniques, computer expansion, plus provide answers to questions from our subscribers.

The submission of questions, operating hints, and solutions to problems to be published in this newsletter are encouraged. All submissions become the property of Dynamic Electronics if the material is used. We reserve the right to edit all material used and not to use material which we determine is unsuited for publication.

We encourage the submission of Basic and Machine Language Programs as well as articles. All Programs must be well documented so the readers can understand how the program works. We will pay for programs and articles based upon their value to the newsletter. Material sent will not be returned unless return postage is included. Basic & ML programs should be sent on a tape or disk & comments should be sent as a DAT or BIN file.

****
* *
* DYNAMIC COLOR NEWS *
* *
* October 1985 *
* *
* Editor and Publisher *
* Bill Chapple *
* *
* Secretary *
* Deanne Hill *
* *
****

CONTENTS

WRITING PROGRAMS
(PART 8)

In this series we have been defining basic commands and showing how to use them to develop basic programs. Last month we looked at what would be required to store characters in memory. Since computers only store numbers, a number representing the character is required. The American Standard Code for Information Interchange (ASCII) gives a number for each keyboard character. These numbers can be stored in memory. Let's review the commands for accomplishing this.

```
10 A$=INKEY$: IF A$="" THEN 10
20 A=ASC(A$)
30 B$=CHR$(A)
```

For the preceeding program A$ represents the string for the pressed key. A is the ASCII value of the key and it can be stored or operated on like any other numerical variable. To print the character to a printer or the screen we need a string. Statement 30 converts the ASCII value of A to the string B$. This is the operation we would need if we took the value of A from a memory location. Statement 30 converts this to a string which can be printed. IF we print A we will get the numerical or ASCII value of A.

PEEKS & POKES

These are the commands for storing and recovering numbers from memory. The formats are as follows:

```
10 POKE M,V
20 X=PEEK(M)
```

The POKE command is for storing or putting values into memory. For statement 10 the M is for memory and the V is for value. The values can be from "0" to "255". As an example to store 96 in memory 32000 we would enter:

```
POKE 32000, 96
```

The PEEK command is used to recover values from memory. Notice that the POKE command has a comma between the variables. The PEEK command has parentheses around the memory. You can keep these straight by thinking of the PEEK command as looking into memory with your eyes shielded by your hands. For statement 20, X is the value stored in memory M.

WORD PROCESSOR DEVELOPMENT

We want to develop a simple word processor using the principles covered. Our word processor will do the following:

1. Be able to store characters in memory.
2. Retrieve the characters from memory.
3. Add to our file.

There are many other features that would be required for a quality word processor. For example we would want to edit our text, right justify, move blocks, delete blocks, insert blocks, etc. Our objective now will be to store characters in memory, and to recover them.

STORING CHARACTERS

We are using the term "storing characters". As previously discussed we mean "storing the ASCII values of the characters". We need to reserve some RAM for pointers to show where our text begins and ends. Let's reserve the following:

```
500 MSB of beginning
501 LSB of beginning
502 MSB of ending
503 LSB of ending
```

Continuing Text

To add to our previous text, we just let the memory pointer start at the end of text vector in location 502-3. Then we can go to the part of the program we wrote for storing characters in memory.

WORD PROCESSOR PROGRAM

The following program is discussed in the preceeding editorial. Comments are included to show what each section does.

```
 10 PRINT"WORD PROCESSOR
    DEVELOPMENT PROGRAM
 20 PRINT"COPYRIGHT (c) 1985
 30 PRINT"dYNAMIC eLECTRONICS
    iNC.
 40 PRINT"THIS IS PGM 10-1-85
 50 '500-1 IS BEGINNING OF FILE
     AND 502-3 IS THE ENDING OF
     FILE.
 60 'SET UP MENU
 70 PRINT"W-WRITE CHARACTERS TO
     MEMORY
 80 CLS:PRINT"R-RETRIEVE
    CHARACTERS FROM MEMORY
 90 PRINT"ENTER W OR R
100 A$=INKEY$:IF A$="" THEN 100
110 'CHECK TO SEE WHICH KEY IS
     PRESSED AND BRANCH
     ACCORDINGLY
120 IF A$="W" THEN 150 ELSE IF
     A$="R" THEN 560
130 'IF THE KEY IS NOT A W OR R
     THEN TRY AGAIN
140 GO TO 70
150 CLS:PRINT"THIS WRITES
    CHARACTERS TO MEMORY
160 'SET UP MENU
170 PRINT"C-CONTINUE OLD FILE
180 PRINT"N-NEW FILE
190 PRINT"ENTER C OR N
200 X$=INKEY$:IF X$="" THEN 200
210 IF X$="C" THEN 240 ELSE IF
    X$="N" THEN 280
220 'IF THE KEY WAS NOT A C OR
     N THEN TRY AGAIN
230 GO TO 150
240 CLS:PRINT"THIS CONTINUES
    THE FILE
250 'LET M =THE END OF FILE
     POINTER WHICH IS IN 503
260 Y=503:GO SUB 550:M=V
270 GO TO 370
280 CLS:PRINT"THIS CREATES A
    NEW FILE"
290 Y=500:GO SUB 550: PRINT
    "START OF MEMORY IS"V
300 'PRESS ENTER TO CONTINUE
     WITH OLD BEGINNING MEMORY
     OR ENTER NEW MEMORY
310 INPUT"ENTER MEMORY LOCATION
    FOR START OF FILE OR PRESS
    ENTER TO CONTINUE";Y
320 IF Y=0 THEN Y=V
330 GO SUB 530
340 POKE 500,MS:POKE501,LS
350 POKE 502,MS:POKE503,LS
360 M=Y
370 PRINT"START ENTERING YOUR
    CHARACTERS. USE THE UP
    ARROW KEY TO EXIT.
380 A$=INKEY$:IF A$="" THEN 380
390 PRINTA$;:A=ASC(A$)
400 'CHECK FOR BACKSPACE
410 IF A$=CHR$(8) THEN M=M-1:GO
    TO 380
420 'CHECK UP ARROW TO END
430 IF A=94 THEN 460
440 'STORE THE VALUE IN MEMORY
     AND INCREASE MEMORY POINT-
     ER M AND GO AND GET THE
     NEXT CHARACTER
450 POKE M,A:M=M+1: GO TO 380
460 CLS:PRINT"THIS TERMINATES
    WRITING CHARACTERS
470 Y=M:GO SUB 530
480 'SAVE THE END OF TEXT
     VECTOR IN 503-4
490 POKE 503, MS: POKE 504,LS
500 'GO TO THE BEGINNING
510 GO TO 10
520 'SUBROUTINE BREAKS A VECTOR
     Y INTO ITS MS AND LS
     COMPONENTS
530 MS=INT(Y/256): LS=Y-256 *
    MS:RETURN
540 'SUBROUTINE RETURNS A VALUE
     OF A VECTOR IN MEMORY
550 V= 256*PEEK(Y) + PEEK(Y+1)
    :RETURN
560 CLS:PRINT"THIS PRINTS
    CHARACTERS IN MEMORY
570 'SET UP PRINTER
580 'LINK MULTIPLE IF-THEN
     STATEMENTS WITH ELSE
590 P=PEEK(505):IF P>0 THEN
```

```
    P$="PRINTER IS ON" ELSE IF
    P=0 THEN P$="PRINTER IS
    OFF"
600 PRINTP$
610 PRINT"ENTER N FOR NO
    PRINTER AND Y FO PRINTER
620 PRINT"PRESS ENTER TO
    CONTINUE"
630 X$=INKEY$:IF X$="" THEN 630
640 CLS
650 IF X$="Y" THEN POKE 505,1 :
    GO TO 560
660 IF X$="N" THEN POKE 505,0 :
    GO TO 560
670 'PULL THE BEGINNING VECTOR
     M AND THE ENDING VECTOR E
     FROM MEMORY
680 Y=500:GO SUB 550:M=V
690 Y=503: GO SUB 550:E=V
700 X$=INKEY$
710 IF X$=CHR$(94) THEN 10
720 'CHECK TO SEE IF THIS IS
     FINISHED
730 IF M=E THEN 770
740 A=PEEK(M):A$=CHR$(A)
750 PRINTA$;:IF P>0 THEN
    PRINT #-2,A$;
760 M=M+1: GO TO 700
770 PRINT
780 PRINT"THIS IS THE END OF
    THE FILE.
790 INPUT"PRESS ENTER TO
    CONTINUE";P
800 GO TO 10
```

COMPUTER GRAPHICS
(Part 9)

Last month we discussed drawing lines, boxes, and bar graphs. Previously we had developed a character generator using high resolution graphics. This month we want to continue with our bar graphs and combine the character generator so that we can label our graphs.

Graphs are very useful for presenting data in a visual form. For example if we use 12 bars with one representing each month in a year, then we can present monthly sales, electriciy used, values of stock, etc. With a bar graph we can quickly determine where the highs and lows occurred. This is much easier than looking over data and having to mentally decide which is lower and which is higher.

A graph is composed of two axes which are X and Y. The X axis is horizontal and the Y axis is vertical. On our X axis we will want the first bar to represent January, the second February, etc. Of course we can define them to be anything we want. For the Y axis we want values such as the amount of sales. These values will have to be normalized. Our maximum value will be 180. In our program we did not include a normalizing routine but you can do this before you enter your data. For example suppose your maximum value is 2230. Then you should multiply each data element by 180/2230 and take the nearest integer. If the result is 135.2 when multiplied by the multiplier, then the data value would be 135. If it were 135.5 then the data value would be 136. For decimals greater than .5 increase the value by 1 and for data values less than .5 just drop the decimal. So our data will consist of numbers from 0 to 180.

So that data is not lost everytime the program is run, we store it in memory. The RUN command erases all variables, so we get our variables from memory. Also we store our characters in memory. The data is stored in 510 to 522 and the characters are stored in 530 up. We exit the character generator with the up arrow which has an ASCII value of 94. So as we read our characters from memory, we know to quit when a 94 is found.

You can save your data and label by saving it as a machine language program. Use the following:

```
(C)SAVEM "DATA",510,650,510
```

This allows 120 characters for the label which should be adequate. This could be incorporated into the program or can be entered from the keyboard.

BAR GRAPH PGM
with
CHARACTER GENERATOR

```
 10 PRINT"BAR GRAPH PGM
 20 PRINT"COPYRIGHT (c) 1985
 30 PRINT"dYNAMIC eLECTRONCS
    iNC.
 40 PRINT"PGM 10-2-85
 50 PRINT"EXTENDED BASIC IS
    REQUIRED
 60 INPUT"ENTER 1 FOR DISK
    DRIVE";D
 70 'SET UP ARRAYS
 80 PCLEAR8
 90 DIM X(5),X$(5),Y(15),K$(100)
100 INPUT"ENTER 1 TO LABEL
    GRAPH";W
110 IF W=1 THEN 530
120 GO TO 150
130 C$=INKEY$:IF C$=""THEN 130
140 RETURN
150 'THIS DRAWS BAR GRAPHS
160 'CLEAR GRAPHICS AND SCREEN
170 PCLS:CLS
180 PRINT"THIS DRAWS BAR GRAPHS
190 'DISPLAY BAR GRAPH VALUES
     FROM MEMORY
200 FOR K=1 TO 12
210 Y(K)=PEEK(510+K)
220 IF Y(K)>160 THEN PRINT
    "VALUE GREATER THAN 160
230 'TAKE CARE OF THE A,B,& C
    OPTIONS
240 IF K=10 THEN PRINT"A";: GO
    TO 290
250 IF K=11 THEN PRINT"B";:GO
    TO 290
260 IF K=12 THEN PRINT"C";:GO
    TO 290
270 'LABEL AND PRINT THE VALUES
280 PRINTK;
290 PRINTY(K)
300 NEXT K
310 PRINT"ENTER CHARACTER TO
     CHANGE OR PRESS ENTER KEY
     TO DRAW GRAPH.
320 GO SUB 130:N=VAL(C$)
330 'DEFINE N FOR C$>9
340 IF C$="A" THEN N=10 ELSE IF
    C$="B" THEN N=11 ELSE IF
    C$="C" THEN N=12
350 'BRANCH IF VALUE TOO LARGE
     OR TOO SMALL
360 IF N>12 THEN 150 ELSE IF
    N=0 THEN 400
370 PRINT"ENTER NEW VALUE FOR
    "C$
380 INPUT X:POKE 510 +N,X:
    GO TO 150
390 'ENTER GRAPHICS MODE
400 PMODE 3,1:SCREEN 1,0
410 LINE (20,160)-(20,0),PSET
420 LINE (20,160)-(255,160),
    PSET
430 'DRAW BAR GRAPH
440 FOR Q=2 TO 13
450 'DEFINE Y2 SO GRAPH STARTS
     FROM THE BOTTOM UP
460 Y2=160-Y(Q-1)
470 X1=18*Q:X2=X1+2
480 LINE (X1,160) - (X2,Y2),
    PSET,BF
490 NEXT Q
500 GO SUB 570
510 GO SUB 130
520 GO TO 150
530 '
540 'SET UP A CHARACTER ARRAY
550 'LET M=5120 FOR DISK DRIVE
560 GO SUB 1500:GO TO 150
570 CS=1:G=2:PP=4
580 'SKIP 180 LINES TO START
     LABEL
590 LD=180:M=3072+32*LD
600 IF D=1 THEN M=5120+32*LD
610 'SET UP GRAPHICS MODE
620 K$(42)="04211404142104
630 K$(43)="00040431040400
640 K$(44)="00000000040408
650 K$(45)="00000031000000
660 K$(46)="00000000000004
670 K$(48)="14171921251714
680 K$(49)="04120404040414
690 K$(50)="14170106081631
700 K$(51)="31010206011714
710 K$(52)="02061218310202
720 K$(53)="31163001011714
730 'ASCII 54="6"
740 K$(54)="07081630171714
750 K$(55)="31010204080808
760 K$(56)="14171714171714
770 K$(57)="14171715010228
780 K$(58)="00000400040000
790 K$(59)="00000400040408
800 K$(60)="02040816080402
810 K$(61)="00003100310000
820 K$(62)="08040201020408
```

```
830 K$(63)="14170204040004
840 K$(65)="04101717311717
850 K$(68)="30171717171730
860 K$(69)="31161630161631
870 K$(70)="31161630161616
880 K$(71)="15161616191715
890 K$(72)="17171731171717
900 'ASCII 73="I"
910 K$(73)="14040404040414
920 K$(74)="01010101011714
930 K$(75)="17182024201817
940 K$(76)="16161616161631
950 K$(77)="17272121171717
960 K$(78)="17172521191717
970 K$(79)="14171717171714
980 K$(80)="30171730161616
990 K$(81)="30171717211813
1000 'ASCII 82="R"
1010 K$(82)="30171730201817
1020 K$(83)="14171614011714
1030 K$(84)="31040404040404
1040 K$(85)="17171717171714
1050 K$(86)="17171717171004
1060 K$(87)="17171721212717
1070 'ASCII 88="X"
1080 K$(88)="17171004101717
1090 K$(89)="17171004040404
1100 K$(90)="31010204081631
1110 'H IS THE NUMBER OF
      CHARACTERS ON A LINE
1120 X=0:H=16*G
1130 M1=530
1140 A=PEEK(M1):A$=CHR$(A)
1150 IF A=94 THEN RETURN
1160 PMODE 3,1: SCREEN 1,1
1170 'CONVERT THE STRING TO A
1180 'VALUE
1190 N=M
1200 '
1210 'BREAK THE STRING K$ DOWN
1220 FOR J=1 TO 7
1230 Q=2 * J-1:X$=MID$ (K$ (A),
     Q, 2) :Y=VAL (X$): X=Y
1240 POKE N,X:N=N+32
1250 NEXT J
1260 'LEAVE SPACE BELOW CH
1270 POKE N,0
1280 M=M+1:C=C+1
1290 'CHECK FOR END OF LINE
1300 IF C=H THEN GO SUB 1440
1310 M1=M1+1:GO TO 1140
1320 FOR K=1 TO 5:AA=K-1:Z=2aA
1330 X(K)=Z AND Y: IF X(K)>1
     THEN X(K)=1
1340 NEXT K
1350 B1=3 * X(5)
1360 B2=3 * X(1) + 12 * X(2) +
     48 * X(3) + 192 * X(4)
1370 '255 MEANS ALL BRIGHT
1380 B1=255-B1
1390 B2=255-B2
1400 POKE N,B1: POKE N+1,B2
1410 POKE N+16,B1:POKE N+17,B2
1420 N=N+32
1430 RETURN
1440 M=N+1:C=0
1450 'LEAVE SPACE BETWEEN ROWS
1460 FOR F=1 TO 6*H
1470 POKE M,0:M=M+1:NEXT F
1480 RETURN
1490 END
1500 PRINT"THIS STORES THE
     GRAPH LABEL IN  MEMORY.
     ENTER LABEL & EXIT WITH UP
     ARROW.
1510 M1=530
1520 P$=INKEY$:IF P$="" THEN
     1520
1530 PRINTP$;
1540 P=ASC(P$)
1550 IF P=8 THEN M1=M1-1: GO TO
     1520
1560 POKE M1,P
1570 IF P=94 THEN RETURN
1580 M1=M1+1: GO TO 1520
```

Second Port
(Hardware Project)

The expansion port on color computers is very useful. Disk drives, cartridges, plus other accessories use this port. The disadvantage of having only one port is that devices have to be plugged in and disconnected. If you have a multiport expander then you can select the devices by a switch or software.

In this article we will explain how to wire a second port. We suggest that if you are not reasonably experienced with soldering you do not attempt this project. There are 40 pins on the expansion connector. All of the pins are not needed for running cartridge programs. However a second port can be added on the right side of the computer near the keyboard. If you want all of the capabilities of the present port then you will have to wire all 40 pins. You will need a switch to select the ports and provide for disabling the unselected port. A miniature single pole double throw switch is ideal for this purpose.

Disabling the Port

Pin 32 of the port is the enable line. It comes from pin 12 of the 74LS138. You will need two 1/4 watt resistors of any value from 4.7 to 10K. The resistors can be glued to the top of the 74LS138. Do the following to provide for port selection:

1. Cut pin 12 of the 74LS138. If it is in a socket just bend the pin out.
2. Run a wire from the chip side of the cut to the center of the switch.
3. Connect one side of both resistors to 5 volts. Pin 16 of the 74LS138 is 5 volts.
4. Connect the free side of one

resistor to the circuit board side of the cut and to one side of the switch.

5. Connect the free side of the other resistor to pin 32 of the new socket and the other side of the switch.

The switch now will select either port. Next we need to wire the second port. As stated earlier we can run a wire to each pin of the main socket but the inside pins are not easily accessible.

For just running cartridge programs in the second port we can obtain the required signals from either the basic or extended basic chips. The following are the connections from the second port to the basic or EB chip:

Function	Basic or Ext Basic	Second Port
A0	8	19
A1	7	20
A2	6	21
A3	5	22
A4	4	23
A5	3	24
A6	2	25
A7	1	26
A8	23	27
A9	22	28
A10	19	29
A11	18	30
A12	21	31
D0	9	10
D1	10	11
D2	11	12
D3	13	13
D4	14	14
D5	15	15
D6	16	16
D7	17	17
Ground	12	33 or 34
+5 volts	24	9

Wiring to the ROM

If you have plug in ROMS then the easiest way is to purchase a 24 pin header and a 24 pin socket. First make the connections to the header and then to the

second port. If you remove the extended basic ROM and plug the header into the socket, you can try out the second port and your wiring. Plug cartridges into each port and you should be able to select them with the switch. Before doing this you might want to mount the second port to the computer and also mount the switch at some convenient location. If there are no wiring errors either cartridge program should run when selected.

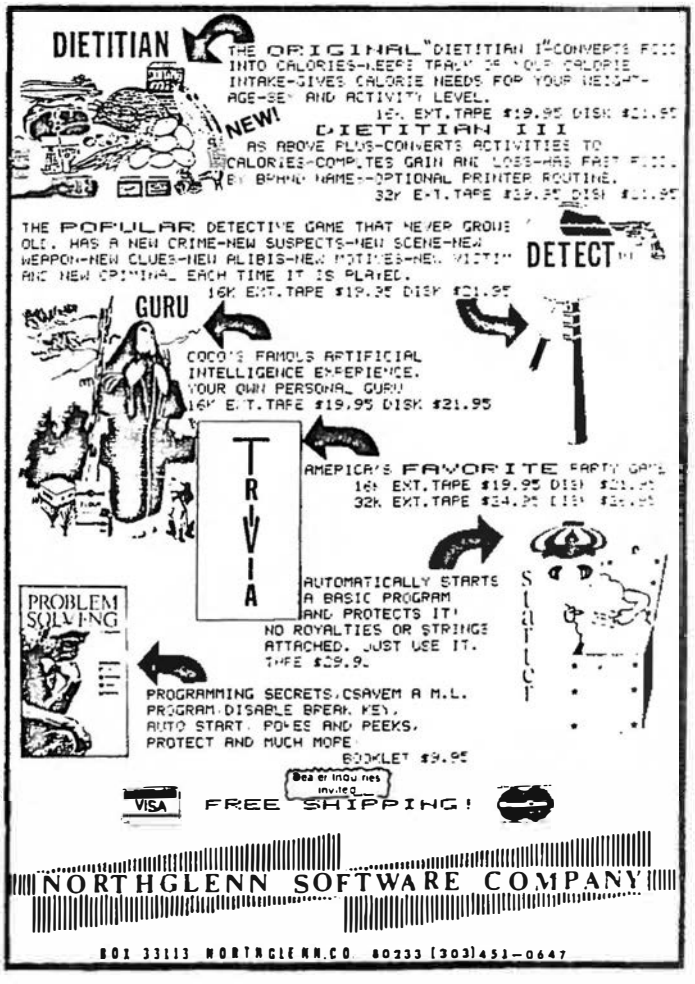
If this works OK you are ready to solder the socket onto the header, plug it into the extended or basic ROM socket and plug the ROM into the socket on the header. You can now turn the computer on and either port should run cartridge programs. If there are problems make sure you have good soldering between the socket and header.

PAGE -1?

Last month we promised to show how you could operate in page -1 and have over 32K of memory available. You have probably heard of page 0. To operate in page 0 do the following:

```
POKE 25,6: NEW for cassette
           operation
POKE 25,14: POKE 14 * 256,0:
      NEW for disk operation
```

Let's briefly look at the memory map. The video display is from 1024 to 1535. For disk operation the area from 1536 to about 3300 is used by the disk for RAM. Therefore programs must start above 3300. For a cassette the disk RAM can be used and programs can start at 1536. This gives vector values of 6,1. Using this notation, the number before the comma is multiplied by 256 and added to the number after the comma. The beginning of basic vectors are in 25 and 26.



The keyboard buffer is directly below the video RAM or from 733 to 988. However there are about 250 bytes below this buffer which are not generally used. So we can load a basic program into this area as long as it is a short program. We have found it very useful to load a utility program that will let us do memory peeks and pokes. These programs can be written in a short space.

We will define page -1 to occupy the area from 512 to 733. To use this area POKE 25,2 : NEW. Then enter or load your program. The program can be left resident in this area and used by poking 25,2. To return to your previous program POKE 25, X where X is the value for the start of the program.

There are some precautions that must be observed when operating in page -1. As stated earlier the program must be less than 220 bytes. In fact it should be much less thann 220 bytes to allow room for variables. Use 200 bytes as a rule and this will take care of most situations.

Other programs can not use this area. If a program uses this area for storing vectors and variables then this will wipe out your special program.

You can find the length of your program by looking at the values in 27 and 28. Actually the value in 28 is the length of your program and should not be greater than about 200. The value in 27 should stay at 2. You can write your program in this area and save it.

We have found that page -1 operation is very handy for short utility programs. It can be accessed with only one memory poke, and you can return to the original program with another memory poke.

LARGE MEMORY PROGRAMS
(Part 9)

In this series we have been explaining how to use the extra memory in your computer, especially the second 32K bank in 64K computers. We have given programs that will configure the computer for the all RAM mode and allow full use of the second 32K bank for the normal mode of operation.

A couple of months ago we started looking at writing a program that would be a directory for programs stored in the second 32K bank. Last month we talked about statement vectors and gave a program for correcting them when a program is moved from one memory area to another. Fortunately extended color basic has a command that will do this for us. It is the PCLEAR command which is used to reserve a number of graphics pages. The PCLEAR command actually moves the program from one memory area to another, and corrects the statement vectors for the new location.

We have been looking at what would be required for our directory. There is one more thing that we want to consider. Since we will only have 32K of memory for programs, it will be beneficial if we can pack them as tightly as possible. If we delete a program, then we want to move programs together to leave more free space. This means that we need to do block moves in the second bank since our control program will be in the first bank. To make this possible we wrote a machine language subroutine that will allow us to move data in the second bank.

Previously our machine language links were in the memory area from 4000-4200. This is not a very desireable location so we decided to move it to 32000 so it would be out

of the way. So this month we are including a complete set of machine language subroutines for all of our requirements. We will give the values so you can poke them into memory and save them as a machine language program.

BANK 2 Data Relocation

This is new material so we will show how this is done. The following vectors are in bank 1 and are for the memory in bank 2.

474 - Beginning of data
476 - Ending of data
478 - New beginning of data

This subroutine will start at 32140 and our assembly language listing is as follows:

```
32140 LDX E 474 'Put the
          beginning vector in X
32143 LDY E 478 'Put the ending
          vector in 478
32147 STA E  65493 'switch to
          bank 2
32150 LDA X R+ 'Load A register
          with memory indicated
          by X and increment X
32152 STA Y R+ 'Store A in the
          memory indicated by Y
          and increment Y
32154 STA E 65492 'Go back to
          the first bank
32157 CMPX E 476 'Is X=value in
          476?
32160 BLS 32147 'Branch if true
          to 32147
32162 RTS
```

ML Subroutines

The following are the updated machine language subroutines. They reside in memory from 32015 to 32162.

1. Transfer ML subroutines to upper memory (32015-32038)

```
32015 - 142, 125, 44, 16, 142
32020 - 253, 44, 166, 128, 183
32025 - 255, 214, 183, 255, 223
32030 - 167, 160, 183, 255, 222
32035 - 140, 125, 200, 45, 238
32040 - 57, 18, 18, 18
```

2. Exchange banks (32044-32067)

```
32044 - 79, 95, 31, 1, 166, 132
32050 - 183, 255, 213, 230, 132
32055 - 167, 132, 183, 255, 212
32060 - 231, 128, 140, 127, 255
32065 - 35, 237, 57, 18, 18
```

3. Copy Bank 1 into Bank 2 (32070-32089)

```
32070 - 79, 95, 31, 1, 166
32075 - 132, 183, 255, 213, 167
32080 - 128, 183, 255, 212, 140
32085 - 127, 255, 35, 241, 57
```

4. Block move from Bank 1 to Bank 2 (32090-32112)

```
32090 - 190, 1, 218, 16, 190
32095 - 1, 222, 166, 128, 183
32100 - 255, 213, 167, 160, 183
32105 - 255, 212, 188, 1, 220
32110 - 35, 241, 57, 18, 18
```

5. Block move from Bank 1 to Bank 2 (32115-32137)

```
32115 - 190, 1, 218, 16, 190
32120 - 1, 222, 183, 255, 213
32125 - 166, 128, 183, 255, 212
32130 - 167, 160, 188, 1, 220
32135 - 35, 241, 57, 18, 18
```

6. Block move in Bank 2 (32140-32162)

```
32140 - 190, 1, 218, 16, 190
32145 - 1, 222, 183, 255, 213
32150 - 166, 128, 167, 160, 183
32155 - 255, 212, 188, 1, 220
32160 - 35, 241, 57
```

Before entering the values do the following to protect the data:

```
CLEAR 600, 32000
```

After the values are entered then they can be saved as a machine language program:

```
(C)SAVEM "ML",32000,32200,32000
```

EDITOR'S COMMENTS

Organization is a big problem for me or should I say that lack of organization is my big problem. It is easy to get involved in particulars and miss your objective. Our objective with this newsletter is to present monthly information on color computers. We have been slipping with our schedule and this edition will be mailed near the end of October.

So to help with our problems we decided to change our order of articles a little. As soon as we write an article, we will get it ready to be printed. This will save us the task of printing everything at the same time. So before this is taken to the printers, we will be printing articles for our next issue. We are shooting for a 3 week delay between this and the November issue. We will allow us two weeks and the printer one week. So we should be able to move forward a week each month.

Included are the articles we promised last month on the hardware project and operating in page -1. Next month we will give you a page -1 utility program you can use. We would like to hear from you about what you would like for us to cover. Do you want more hardware projects?

There are many new topics we could cover. What about a series on storing information? We are familar with cassettes and disk drives. How do they work? Should one expand to a disk drive if he only has a cassette? What about printers? How do you know what type to purchase? What is a hard disk? What about monitors instead of televisions? These are some questions that need answering especially if you are interested in becoming involved with these hardware items.

What about software? We have mainly been developing software in these editorials. Software is what makes your computer perform. As previously mentioned your computer can be made to do many different tasks with software.

There are many new products being developed for color computers and we will list the ones that are sent to us in our New Products section. These computers have been on the market for several years now so it is easy to develop products for them. Memory expanders are now available for 128K, 256K, 512K and up. This means that you can store a lot of information quickly in RAM for fast access. You can expect more new and sophisticated products to be available soon.

PRODUCT REVIEWS

This section is open to all producers and dealers of color computer products. We will review you product free of charge and write an editorial on the product. We do not use a rating system but will explain what the product does, and what can be expected from it. Any comments about the review from the firm submitting the product will be printed in a later issue.

* * * * * * * * * * * * * * * * *

BASIC +

Basic + consists of several utility programs that greatly enhance the operation of Color Computers. It requires a 32K Extended Color Basic Computer and a disk drive.

It has about 20 commands that can be entered by pressing only one key. Also you can scroll forward and backward through a basic program and edit the last line scrolled by pressing only one key. This gives the advantage of a full screen editor.

As a bonus, each key will auto repeat and give a click sound in the speaker.

Some useful printer features are included. For example the printer can be turned on so that everything that appears on the screen is printed. Also a large printer buffer is included so that you do not have to wait for the printer. We listed an 8K program to the printer and it took just a few seconds for the familiar "OK" to appear on the screen using the buffer. The printer buffer uses the upper 32K of memory.

Also included is a 32 character typahead keyboard. With this you can type in commands while previous commands are being executed.

Another useful feature is that commands can be generated and saved as a machine language program. These commands are dumped into the keyboard buffer and are executed as if they were typed in from the keyboard. They can be included within a program and called by using LOADM "NAME":EXEC.

The instructions are very complete and we did not have any problems in running the programs. Basic + resides in memory from 31489 (7B01) to 32767 (7fff) which is only about 1.25K of memory. This allows it to work on 32K machines and on most programs that use the total 64K of RAM in 64K computers. If the program uses the area occupied by Basic + then of course Basic + can not be used.

One Key Commands

The following are the keys and their commands:

Key	Command
!	LOAD "
"	LOADM "
#	SAVE "
$	SAVEM "
%	CLOAD "
&	CLOADM "
'	CSAVE "
(	CSAVEM "
*	DUAL TOGGLE or print to screen and printer.
:	DEL
=	? PEEK (
@	RUN
,	COPY "
.	EDIT
/	KILL "
;	DIR
+	RENUM

DOWN-ARROW scrolls down one line & lists it.

UP-ARROW scrolls up one line and lists it.

. Edits last line listed with arrows keys

: Deletes last line listed with arrow keys.

Basic programs can be listed one line at a time by using the up and down arrows. To edit the previous line just press the "." key and you are in the edit mode for that line. Because each key auto repeats, you can quickly move to any part of the line with the space bar and left arrow key. These features are a tremendous time saver especially for writing and editing basic programs.

SUMMARY

Basic + provides about 20 of the most used basic commands with one key stroke, it provides auto repeat for each key, and has a 32K printer buffer plus allows keyboard programs to be generated and loaded as machine language programs. These are very powerful and useful programs.

Spectrum Projects, P. O. Box 21272, 93-15 86th Dr., Woodhaven, NY 11421, $29.95 + $3 S/H.

- - DCN STAFF - -

```
+ ++ + ++ + ++ ++ ++ + ++ + ++ +
+                               +
+         RENEWAL TIME?         +
+                               +
+ The date beside your name on +
+ the address label indcates   +
+ the last issue you will re-  +
+ ceive. Send in your renewal  +
+ if you want to continue re-  +
+ ceiving technical informa-   +
+ tion on Color Computers.     +
+ This is the last issue for   +
+ those with 10/85.            +
+                               +
+ + ++ + ++ + ++ ++ ++ + ++ + ++
```

NEW PRODUCTS

This section is available free for producers and dealers of color computer products. These products have not been reviewed by us but are included for our reader's information. Send a description of new products to:

New Products
Dynamic Electronics Inc.
P. O. Box 896
Hartselle, AL 35640

* * * * * * * * * * * * * * * * * *

512K RAM DISK

A 512K RAM DISK cartridge for the Radio Shack Color Computer is now available for $298. The CCRD cartridge is designed to plug into the multipak expansion bus of a CoCo running OS-9. OS-9 drivers for the device are available separately on disk for $20. Two cartridges provide a single device with 1 Megabyte of

****
* BACK ISSUES *
* Back issues of DCN are available for $1.95 each, 3 for $5, or 12 for $15 pp. *
* Foreigners other than Canada add $2 for Air Mail postage. *
****

storage. A ram-disk is especially useful for speeding up compiles of the C or Pascal compilers by moving programs, source code and libraries to the ram disk. For more information or to order contact: D. P. Johnson, 7655 Southwest Cedarcrest St., Portland, Oregon 97223. (503) 244-8152.

SPECTRUM FONT GENERATOR

The Spectrum Font Generator allows the user to print out documents in a highly detailed character set on most dot-matrix printers. Facilities are included to allow the user to create custom character sets or modify existing ones. The file must be in ASCII format and can be printed out in Italics, Old English, Futuristic, and Block. After the desired program is load, all output to the printer will be in the selected character set. This includes all PRINT#-2 and LLIST commands. For more information contact: Spectrum Projects, P. O. Box 21272, 93-15 86th Dr. , Woodhaven, NY 11421.

OS9 UTILITIES

IVA Electronics has a new series of OS9 utilities called EZ-Back and EX-Manager. EZ-Back performs file system backups while EZ-Manager allows faster and easier access to the complex hierarcal file system of OS9. For more information contact: IVA Electronics, 6117 Gerard Morisett, Montreal, QUE., Canada HIM 3J8.

CLASSIFIED ADS

1. 10 cents a word, $3 minimum.
2. Name, Address, & Telephone listed free.
3. Send payment with ad.
4. Closing date 1st of the preceeding month. Ex. Nov ad closing is Oct. 1.

X-RATED Adventure Game!

A game so real that you will know that your COCO has finally grown up. 16/32K $15 tape/ $20 disk; Y. Kraft, 164 N. Blackstone Ave., Fresno, CA 93701

* * * * * * * * * * * * * * * * *

DISPLAY ADS

Closing 1st of preceeding month.

Pages	1 time	2 times	3 times
1	$25	$22	$20
1/2	18	15	13
1/4	10	8	7

* * * * * * * * * * * * * * * * *

!!!!!! Attention Dealers !!!!!

We need announcements of your new products to print. Also we need products to review. Have you considered our low advertising rates?

****
* Please sign me up for one year for the DYNAMIC COLOR NEWS SERVICE. I *
* Want to receive instruction on programming, Computer Theory, Operat- *
* ing Techniques, Computer Expansion plus the Individual Reply to my *
* Computer problems for a special of $10 each. Also I understand that *
* therm will be no c(arge for letters printed with answers in the *
* Newsletter. Cost $15 USA & Canada, $30 foreign. *
*
* Name _____ Mail payment to *
* Address _____ Dynamic Electronics Inc *
* City _____ P. O. Box 896 *
* State & Zip _____ Hartselle, AL 35640 *
* Enclosed is a check ___ *
* charge to VISA ___ MC ___ Number _____Exp._____ *
*
****

DYNAMIC ELECTRONICS INC.
P. O. Box 896 (205) 773-2758
Hartselle, AL 35640