ENGINEERING NOTES
on
Radio Shack Color Computers

April 1984 $1.95
Vol. 1 No. 3

DYNAMIC COLOR NEWS

* ASCII PART I

* COMPUTER THEORY
* BASIC PROGRAMMING
* OPERATING HINTS
* MACHINE LANGUAGE PROGRAMMING
* QUESTIONS & ANSWERS

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.

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 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.

All paid subscribers are entitled to discounts of 10% on hardware. and 20% on software manufactured or produced by Dynamic Electronics Inc. plus "specials" mentioned in the newsletter. To receive these discounts use your DCN number which is at the right of your name on the address label. DCN subscribers may obtain a personalized reply to specific computer problems or advice on purchasing equipment. The charge for this service is $10.

```
*******************************
*                             *
*   DYNAMIC    COLOR    NEWS  *
*                             *
*         March  1984         *
*                             *
*    Editor and Publisher     *
*        Bill Chapple         *
*                             *
*          Secretary          *
*        Belinda Parker       *
*                             *
*******************************
```

CONTENTS

EDITOR'S COMMENTS

Here in North Alabama Spring is arriving. We didn't have much snow this year although we did have a cold December. We have received some letters from you and greatly do appreciate them as well as the phone calls. The comments we have received are very encouraging and it looks like we are covering the subjects in which you are interested.

This month we are starting a series on ASCII. This is the means for transmitting characters over telephone lines, from the computer to printer, and from one terminal to another. I don't know how many months we will devote to this, but we will continue until we think we have adequately covered the subject.

Another subject we are considering covering is Computer Electronics. The amount of heat generated in the computer can cause components to fail. How many accessories can be safely added without damaging the computer? Memory upgrades, character generators, video reversers, disk controllers, etc. all require power and cause additional heat. The 5 volt voltage regulator is not efficient and generates a lot of heat. I have designed and built regulators that do not get hot. If these are incorporated into the computer then more accessories can be added.

We have made good progress in the last month by putting circuits in modules that mount under the chips. The chip mounts in the socket on the module. This gives a neat installation and allows more circuits to be installed inside the computer. I am very pleased with our 96KX-M. It consists of two modules that mount under the microprocessor chip (6809E) and the extended basic chip. A resistor has to be cut and two wires connected to the ends of the break. I like this because the software is always available and I don't have to find something to plug in or load. I use the software more for trouble shooting programs than for using the other memory page in my 128K computer. In developing the 96KX-M modules I destroyed 2 SAM chips. These are the most expensive chips in the computer so I decided to do something about it. I built a module for the SAM chip with an 8 channel amplifier. The SAM chip supplies the clock signals to the microprocessor plus several other signals. I ran all of these signals through an amplifier. I have had no trouble or failures since. If you short out one of these signals either inside the computer or on the expansion port, the SAM chip will probably fail. With the amplifier the chances are very small that it would fail and I haven't had any trouble since the amplifier was installed.

What effect does adding accessories have on the performance of a computer? Some people seem to think that if extended basic is added or more memory then the performance of the computer is degraded. This is not true. When you add more memory you don't loose any of your previous features. Adding extended basic causes the computer to clear several graphic pages resulting in less memory being printed when you ?MEM. You can still POKE 25, 6: NEW and then ?MEM and you will get the same as without the extended basic chip. When you add a disk drive then you loose about 2000 bytes because the drive has to use some of the computer's memory. With our 128K expanders you will have everything you would have with any 64K computer except you would now have the equivalent of two 64K computers. You can switch banks and run Basic or a Machine Laguage program in the other bank. Our 96KX uses the upper 8K of ROM and does not require any of your computer's RAM for its use and allows you to use all of the 64K RAM plus retain the Basic and Extended Basic ROMS. There are a couple of software programs that will allow you to access the other 32K memory bank but they have to be loaded into your RAM in both banks reducing your usable memory.

You have to make a trade-off decision between software and hardware accessories. For example should you install a lowercase hardware character generator or purchase a software program. Once a hardware item is installed it is always available and doesn't require loading. However it requires power, occupies space and causes additional heat to be generated. There are always decisions to be made and it is helpful if you know what options are available. I can visualize a 256K color computer with internal battery to keep the memory chips refreshed, a bank of read only memories (ROMS) for storing a word processor, accounting program, assembler - disassembler, plus file manager programs. I have been using a small 2" television with my computer for about a year now. It is easy to read and could be permanently mounted on the top of the computer. Also I have been using a micro-cassette recorder. It too could easily be mounted to the computer giving a small powerful portable computer. I looked at the Radio Shack Model 100 and other portable self contained computers but saw that it would cost too much to expand their memory because they use static RAMS. Therefore I think that a 128K or 256K portable self contained color computer would be the way to go.

We have had many requests for a video reverser that will work in the graphics mode with word processors such as the Telewriter 64. This is the word processor we are using with an Epson MX-80 printer. I am going to look into this and see if we can develop a plug in module for this application.

ASCII - PART 1

This subject is very important when it comes to computers. In fact it is about the only thing that is common between computers. It defines how characters are stored in memory, how they are transferred to a printer, or sent over telephone lines through a modem. ASCII is also used to define printer control functions. Before we continue let's define ASCII. ASCII stands for the American Standard Code for Information Interchange. It is pronounced as ask-eee or ask-key by pronouncing only one "k".

ASCII generally means that information is sent out one bit at a time. For this discussion we are going to call the device that sends the characters the "sender" and the device that receives the characters the "receiver". It is much harder to send information in strings of bits than it is to send a byte at a time. Remember a byte contains 8 bits. Before a character can be sent the receiver has to be ready. The receiver also has to be prepared to receive at the rate the information is being sent. For example if you send information too fast to a printer it will print garbage. For sending information over telephone lines two tones are used to indicate a 1 or a 0. You might ask "well what is the advantage of using ASCII" ? The answer is the simplicity of the hardware interfaces. It would be very involved if we tried to send an 8 bit byte over the telephone line at one time. We would have to have 8 tones for the bits or we would need 8 telephone lines. ASCII is used to transmit information over radio channels and this reduces channel width and allows more stations to operate than could be allowed if a byte at a time were transferred.

Information can be transferred at very fast rates using ASCII.

BAUD RATE

The rate at which informated is transferred by ASCII is called BAUD. The higher the number the faster the rate. Most bulletin boards and modems used with home computers use 300 baud while some use 1200 baud. Some commercial modems transfer information at baud rates up to 9600. Everyone is familiar with loading and storing information from a cassette recorder. The baud rate for this is around 1500. So this should give you an idea of baud rate.

7 or 8 BIT WORDS

Now let's look at the structure of an ASCII word. For computers a byte consists of 8 bits. However for sending characters only 7 bits are required. With 7 bits we can have up to 127 different characters. This includes all capital letters of the alphabet, all lower case letters, all numbers and punctuation, plus machine codes for printers, etc. Now if we are going to send Basic programs using ASCII then we will need 8 bit words since we will need to transfer all the bits of a byte. So ASCII can send either 7 or 8 bit words.

Start & Stop Pulses

The rate at which information is sent is selected by storing (poking) a value into a memory location or selecting the value with a switch. As mentioned earlier both sender and receiver have to be initialized for the same rate. The sender sends a logical "1" to indicate that it is on line. The receiver responds with a ready signal. Everything has to be properly timed. The receiver must know when information is beginning. As soon as it senses a change from the logical "1" to a "0", it begins its timing sequence. The first thing it sees is the "start pulse". After timing the start pulse the receiver times the first bit and remembers whether it is a 1 or a 0. It then in turn removes the remaining bits until it has received 7 or 8 bits. It then times the stop pulse which is a "1". If there are supposed to be 2 stop pulses it times the second one also. The character is completed and printed or stored depending on the application. The receiver is now ready for the next character. The chart below will show how an ASCII character is sent.

	OL	ST	B1	B2	B3	B4	B5	B6	B7	B8	SP	SP
VALUE	1	0	0,1	0,1	0,1	0,1	0,1	0,1	0,1	0,1	1	1
BIT #		1	2	3	4	5	6	7	8	9	10	11

OL represents the on line or ready to receive state, ST represents the start pulse which is always a 0, B1 through B8 represent the bits of the character which can be 0 or 1, and SP represents the stop pulses which are always 1. After the last stop pulse we are back to the ready to receive state. Notice that there are 11 bits required to send a character. If we want only one stop pulse then 10 bits are required.

PARITY Bit

It may be desireable to perform a check to see if the character sent was sent correctly. For text only 7 bit characters are used. The eighth bit can be used as a check bit or parity bit. Parity is determined by counting all of the ones that make up the character. If there are an even number of 1's then the parity is 0 and if there are an odd number then the parity is 1. The eighth bit is then sent as a 0 or 1 to indicate the parity of the character. The receiver determines the parity from the bits it has received. If the parity bits agree then the received character is probably correct. This is called "even parity". Odd parity is determined the same way except the complement is sent. If the parity is a 0 then a 1 is sent.

Summary

We have covered the basics of ASCII. If you use a printer or modem then you will use some of these terms. ASCII is also used in Basic Programming. Characters can not be stored in memory. Computers work with numbers and only numbers representing characters can be stored. When a character is entered from the keyboard it is assigned an ASCII value before it is stored in memory. When characters are to be retrieved from memory then the numerical value representing the character is converted into the character so it can be displayed.

BASIC PROGRAMMING THEORY

Arrays, DATA & READ Statements

Last month we introduced variables and arrays. We want to continue with this and show how information can be organized using this technique. The use of READ and DATA statements give us an easy way to enter information without having to type labels for the data. When using READ and DATA statements the information has to be perfectly ordered. If one item is out of line the the computer reads the wrong information. If you need information contained in the last few data statements then you have to start at the first and let the computer read them all including the ones you don't want. Another disadvantage is that double memory is required for the information. Carrying the data within data statements occupies space in your program. When the computer reads the data it stores it in memory so the information is handled twice. However the procedure is easy to understand so we will continue with this approach this month.

We are not covering all of the Basic programming operations. If we use something that you are not familiar with then read about it in your book that came with your computer or a Microsoft programming book. In our computer classes we used example checkbook programs to demonstrate concepts. So this month we want to show you how to design a checkbook program using DATA statements. As stated last month think of the information as if it were displayed on a chart with columns and rows. We would like to display the check number, amount, to whom the check was written, and the date it was written. What about deposits? The easiest way to handle them is to consider them as negative checks. If the check is (-) then subtracting a (-) gives a plus or deposit. We will set up the program to subtract the value of the check from the balance and give a new balance.

Also we want to display the balance after the check or deposit and it would be nice if we could let the program search for a check either by using the check number or entering the name of the person to whom the check was written. The following is a program with remarks about what each section does. You can eleminate the remark if you want to type and run the program.

```
5 'CHECK BOOK PROGRAM USING DATA STATEMENTS AND ARRAYS
10 'SET UP ARRAY SIZES WITH DIM STATEMENT.
12 'WE ARE SETTING UP THE ARRAYS TO HANDLE 50 CHECKS
15 DIM N(50),A(50),R$(50),D$(50),B(50)
20 'N(X) IS THE NUMBER OF THE CHECK
25 'A(X) IS THE AMOUNT OF THE CHECK
30 'R$(X) IS THE NAME OF THE RECEPIENT.
35 'D$(X) IS THE DATE THE CHECK WAS WRITTEN
37 'B(X) IS THE BALANCE AFTER THE CHECK IS SUBTRACTED OR THE
40 'DEPOSIT ADDED.  NOW READ THE BEGINNING BALANCE BB
45 READ BB
48 'WE ARE USING 500.95 FOR OUR BB
50 DATA 500.95
55 'NOW READ THE INFORMATION FOR THE CHECKS
58 B=BB
60 FOR J=1 TO 50: READ N(J), A(J), R$(J), D$(J)
61 'CALCULATE THE BALANCE AFTER EACH CHECK
62 B=B-A(J):B(J)=B
65 'TEST TO SEE IF WE ARE OUT OF DATA
70 IF N(J)=0 THEN 80
75 NEXT J
80 'SAVE THE NUMBER OF THE CHECKS READ (NC)
85 NC=J-1
90 'WRITE A MENUE OF OPTIONS
95 PRINT"WHAT DO YOU WANT TO DO?"
100 PRINT"1 DISPLAY ALL CHECKS"
105 PRINT"2 DISPLAY CHECK #----
110 PRINT"3 DISPLAY CHECKS BEGINNING WITH THE LETTER ---"
125 INPUT X
130 ON X GO TO 200, 400, 600
140 'STATEMENT 130 IS THE EASY WAY TO DO MULTIPLE BRANCHES.
150 'IF X=1 THEN WE GO TO 200.  IF X=2 THEN WE GO TO 400, ETC.
200 PRINT"THIS DISPLAYS ALL CHECKS
205 'SET UP LOOP TO DISPLAY INFORMATION
210 B=BB:PRINT:PRINT"BEGINNING BALANCE="B
215 'WHEN STARTING CALCULATIONS LET THE BALANCE BE
216 'EQUAL TO THE BEGINNING BALANCE
220 FOR J=1 TO NC
225 PRINT
230 'REMEMBER NC WAS THE NUMBER OF THE CHECKS THAT WE HAVE
235 PRINT N(J);"TO "R$(J);" ON "D$(J)
240 B=B-A(J):B(J)=B
245 PRINT"FOR"A(J),"BAL="B
250 'SLOW THE DISPLAY DOWN
255 FOR K=1 TO 200: NEXT K
260 NEXT J
290 GO TO 2000
400 ?"THIS DISPLAYS ONE CHECK
410 INPUT"CHECK #";N
415 IF N=0 THEN 100
420 PRINT
430 FOR J=1 TO NC-1
450 IF N(J)=N THEN 460 ELSE NEXT J
455 PRINT"NO SUCH NUMBER":GO TO 410
```

```
460 PRINTN;"TO ";R$(J);" ON "D$(J)
470 PRINT"FOR "A(J),"BAL="B(J)
475 PRINT
480 PRINT"DO YOU WANT TO LOOK AT ANOTHER CHECK #?"
490 A$=INKEY$: IF A$="Y" THEN 400 ELSE IF A$="" THEN 490
495 GO TO 100
600 PRINT"THIS DISPLAYS CHECKS BEGINNING WITH A LETTER
610 INPUT"LETTER";L$
620 FOR J=1 TO NC
625 'STRIP THE LEFT CHARACTER FROM THE RECEPIENT R$(J)
630 X$=LEFT$(R$(J),1)
640 IF X$=L$ THEN 660 ELSE NEXT J
650 GO TO 690
660 PRINTN(J);" TO "R$(J)" ON "D$(J)
670 PRINT"FOR $"A(J),"BAL="B(J)
675 PRINT
680 NEXT J
690 A$=INKEY$:IF A$="" THEN 690 ELSE IF A$="C" THEN 600
695 GO TO 100
700 '
705 'YOU CAN PUT DATA STATEMENTS ANYWHERE IN YOUR
710 'PROGRAM. WE ENTERED THE INFORMATION FOR ONE
720 'CHECK OR DEPOSIT AS ONE STATEMENT. HOWEVER WE
730 'COULD HAVE COMBINED THE INFORMATION FOR SEVERAL
740 'CHECKS INTO ONE STATEMENT.
750 'THE BEGINNING BALANCE IS IN STATEMENT 50
760 '
800 DATA 101,125.50,JAMES SMITH,3-10
810 DATA 102, 35.2,WATER DEPT.,3-12
820 DATA 10,-250,DEPOSIT,3-12
830 DATA 103,50.50,SEARS,3-14
840 DATA 104,110.50,ELECTRIC DEPT.,3-15
850 DATA 105,5.95,PAPER BOY,3-15
860 DATA 106,79.33,CREDIT UNION,3-17
870 DATA 12,-235.50,DEPOSIT,3-19
880 DATA 107,125.33,CHEVRON,3-21
890 DATA 109,25.50,CABLE TV,3-22
990 'STATEMENT 999 ENTERS A 0 FOR EACH VALUE INDICATING END OF DATA
999 DATA,,,,,
2000 A$=INKEY$:IF A$="" THEN 2000 ELSE 100
2010 'STATEMENT 2000 IS A SUBROUTINE THAT WAITS FOR
2020 'A KEY TO BE PRESSED.  IT ALLOWS YOU TO GO OVER
2030 'THE CHECKS BEFORE CONTINUING.
```

MACHINE LANGUAGE PROGRAMMING

DATA RELOCATION

Last month we introduced indexed addressing. This month we want to continue and give you a program that you can use to relocate data. Have you ever wished you could save the information on the screen for later use? Or have you ever wanted to print the information on the screen? Machine language subroutines are very handy for these applications. This month we will give you a subroutine to exchange the information on the screen with information stored in a designated memory area. This subroutine can be called from basic with the EXEC command as we will show.

First let us review a little. The index registers are pointers or vectors that point to memory locations. We have 2 index registers that can be used for this application. They are the X and Y registers. The microprocessor has two working registers which are A and B. We can load a register with the value in a memory location defined by one of the index registers. We can also store the value in a register referenced to one of the index registers. When we say referenced to the index registers we mean we have all of the options covered last month in the chart on page 9. The address that we actually used is the address the index register is pointing to plus the offset. Also the index register can be incremented or decremented after the operation is completed.

Branching and Conditional Tests

Before we continue with the problem we need to cover branches. We need to tell our subroutine when it is finished. In Basic we can use the IF (action) THEN (do something). For example we could have statements similar to

```
          200 IF X=1536 THEN GO TO 10
          250 IF X<1536 THEN 15
```

To do these operations with machine language we have to do 2 operations. First we have to compare the register with a value. This is the "IF X=1536" part. Then we have to put a branch command if the test is true which is the "GO TO" part.

The Compare Instruction

This is easy to use. The symbols are CMPA, CMPB, CMPX, CMPY, CMPD, CMPU, CMPS. With each compare instruction you need to designated the addressing mode, which is Indexed, Direct, Extended, or Immediate. So to do the first part of statement 200 we can enter

```
          CMPXI 1536
```

We will use the immediate mode since we will give the number to compare X with. Always use the immediate mode of addressing when you have the numerical value to use.

Branch Instructions

We will always need branch instructions when we are writing machine language programs so we will go ahead and introduce them now. The list is as follows:

BCC - Branch on carry clear
BCS - Branch on carry set
BEQ - Branch if equal to zero
BGE * Branch on greater than or equal to zero
BGT * Branch on greater
BHI - Branch if higher
BHS - Branch if higher or the same
BLE * Branch on less than or equal to zero
BLS - Branch on lower or the same
BLT * Branch on less than zero
BMI * Branch on minus
BNE - Branch if not equal to zero

```
BPL * Branch on plus
BRA - Branch always
BRN - Branch never
BSR - Branch to subroutine
BVC * Branch on overflow clear
BVS * Branch on overflow set
```

The branch instructions are similar to GO TO in basic. The limitations are the number of bytes you can branch to. Remember last month our discussion of 2's complement numbers. We want to be able to branch backwards in memory as well as forwards. So we use numbers from 0 to 127 for positive branches and 128 to 255 for negative branches with 2's complement numbers.

Also we are fortunate to have the capability of longer branching. To indicate long branching an L is placed in front of the branch instruction. For example LBPL means to long branch on plus. With long branching you can branch to locations from -32768 to 32767 from your present location.

Now back to our original problem. The video display area is from 1024 to 1535. Let's reserve 12000 for the area we are going to store our information and let's use 13000 as the beginning of our machine language program. First let's write a basic program to exchange the information in the two locations.

```
10 X=1024: Y=12000
20 A=PEEK(X): B=PEEK(Y)
30 POKE X,B: POKE Y,A
40 X=X+1: Y=Y+1
50 IF X <> 1536 THEN GO TO 20
60 END
```

For our machine language program we will use a similar approach. Let's do the equivalent of statement 10.

```
13000 LDX I   1024  (Put the value of 1024 in X)
13003 LDY I  12000  (Put the value of 12000 in Y)
```

Now the equivalent of statement 20 is

```
13007 LDA X   R + 0   (Load A indexed to X with no offset)
13009 LDB Y   R + 0   (load B indexed to Y with no offset)
```

The equivalent of statements 30 and 40 is

```
13011 STA Y   R +  (Store A in Y's location and increment Y)
13013 STB X   R +  (Store B in X's location and increment X)
```

Now we need to see if you have finished which is the equivalent of statement 50

```
13015 CMPX I  1536  (compare X with 1536)
13018 BNE   13007   (Branch to 13007 if not equal)
```

To finish the program we need to put a Return from subroutine (RTS) instruction.

```
13020 RTS
```

The numbers to poke into memory if you want to try this subroutine are as follows in decimal and hex:

13000	142	32C8	8E
13001	04	32C9	04
13002	0	32CA	00
13003	16	32CB	10
13004	142	32CC	8E
13005	46	32CD	2E
13006	224	32CE	E0
13007	166	32CF	A6
13008	132	32D0	84
13009	230	32D1	E6
13010	164	32D2	A4
13011	167	32D3	A7
13012	160	32D4	A0
13013	231	32D5	E7
13014	128	32D6	80
13015	140	32D7	8C
13016	6	32D8	06
13017	0	32D9	00
13018	38	32DA	26
13019	243	32DB	F3
13020	57	32DC	39

After you have entered the program you can use it any time from the keyboard by entering "EXEC 13000 ENTER". The display you have will be exchanged with the contents of 12000-12512. The first time you use the program you will probably get garbage on the screen. Clear the screen and execute it again and your original screen should return. The program is position independent which means you can put it anywhere in memory. It was designed to operate on a 16K machine.

OPERATING HINTS

DISABLE ROM PORT

You can disable the expansion port by poking 65314, 54. After doing this you can plug in a cartridge and it will not run. To enable the cartridge EXEC 49152.
To restore the expansion port POKE 65315, 52

HEX TO DECIMAL AND DECIMAL TO HEX

In you have extended basic you can convert hex to decimal by ?&H**** where **** are the hex characters. To convert decimal to hex ?H$(*****) where ***** are the decimal characters.

ML PROGRAM ADDRESSES

You can copy machine language programs. First load the program with (C)LOADM "NAME". Calculate the vectors.

```
   BE=256*PEEK(487)+PEEK(488)
   EN=256 * PEEK(126) + PEEK(127)-1
   EX=256*PEEK(157) + PEEK(158)
```

Then to make the copy type (C)SAVEM "NAME",BE,EN,EX <ENTER>

QUESTIONS AND ANSWERS

These are questions that we have been asked about computers. If you have a question send it to us and we will answer it here. If we can't answer it we will ask for help.

Question: Will OS-9 work if I upgrade my computer with one of your 128K upgrades?

Answer: When you add more memory you do not loose any of your capability. If the software was not designed for 128K then it might not be easy to adapt it to fully utilize the extra memory. You have to be able to poke memory locations to access the additional memory and it may be hard to modify software to allow these pokes, especially if it is in machine language. However if you have the same thing in both 64K memory banks then you can manually switch and everything works. For instance we can load our word-processor into both 64K banks and can switch at any time from one editoral to another. Also if we have Basic programs then we can switch to the other bank by software or with the switch with no problems.

We need help on the following question. If you know of a solution to the problem we would appreciate hearing from you. This question was submitted by E. J. Haas.

Gentlemen:

I am operating a 64K Color Computer with a Tandon double density double sided disk drive and have noticed frequent formatting errors on side two when running DSKINI and more so under the OS-9 FORMAT command. The make of disk does not appear to be of significance and the sectors in error are not consistant. This gives rise to two questions:

(1) Can you advise me of what adjustment to make? (2) Do you know where I may purchase a service manual for this drive.

DYNAMIC ELECTRONIC INC.

HARDWARE PRODUCTS - DCN Subscribers can deduct 10% from these prices.

96KX expanders allow you to use all 64K of your memory in 64K Computers. Transparent until accessed, it works with disk drives and multicartridge selectors. Simply EXEC 57701 from Basic to access its powerful software. In addition to managing both memory pages, the software performs many of the everyday operating and program writing requirements and aids in troubleshooting defective programs. Your 64K computer becomes two 32K computers. There is nothing to load since the 96KX software is in permanent ROM the same as the Basic and Extended Basic ROMS.

* Initilizes the second 32K memory bank so basic can be run in it.
* Transfer blocks of data within or between either bank.
* Enter data in DECIMAL, HEX, VECTOR or ASCII formats.
* Copy machine language programs from magazines in HEX.
* Allows mixed HEX and DECIMAL entries for data storage.
* Performs HEX to DEXIMAL and DECIMAL to HEX Converions.
* Displays program statement numbers and their memory locations.
* Change statement numbers one at a time.
* Gives the beginning, ending, & execute addresses of ML programs.
* Software is included for our interrupt switch for a hard reset.
* Uses top 8K of memory in the ROM memory area.

96KX-M . . . A plug in module that mounts in the extended basic socket and the extended basic chip plugs into the module. Solderless installation for D & E Computers, 2 wires to solder for later version computers. $59.95

96KX-C . . . The same as above except in a plug in cartridge. Use with cassette or disk systems with multipack expanders. $49.95

Interrupt switch . . . Allows the computer to be interrupted and run a machine language program. Mounts in 1/4 inch hole with instructions. Only $9.95

Video Reverser . . Reduces eye strain. Solderless assemblies with 3 position switch to give you (1) All characters reversed, (2) All characters reversed and displayed as capital letters & (3) the normal display. $19.95

Sam Buffer - Integrated circuit that mounts on the SAM chip and protects it against shorts due to memory upgrading. $8.95

Solderless & Reversible Memory upgrades without trace cutting. 64K upgrades require a 1.1 ROM. EXEC 41175 <ENTER> for your ROM type. State board or computer type when ordering.

ME-3 . . .	D, E, & 285 from 16K to 32K	$36.95.
ME-4F . . .	F or 285 and COCO 2 to 64K	$79.95.
ME-4 . . .	D & E Computers to 64K	$89.95.

ME-128-64 . . . If you have a 64K computer you can upgrade it to 128K with this upgrade kit. Wired assembly consists of a set of 64K chips with sockets for your 64K chips, control circuit board for hardware or software bank selection, plus a 3 position toggle switch that mounts in a 1/4 " hole for manual bank selection. $199

YOUR BASIC OR ML PROGRAMS IN A CARTRIDGE . . . We can put your programs in a cartridge and you will have all of your memory. To access these you POKE 25, 192: RUN for Basic or EXEC 49152 for ML programs. Nothing to load, faster than tape or disk. Send a cassette with 2 copies of your program on it. Don't use PCLEAR command. Up to 8K $29.95.

```
************************************************************************
*                                                                      *
* Please sign me up for one year for the  DYNAMIC COLOR NEWS SERVICE. I *
* understand I will receive a monthly news letter, Discounts on DYNAMIC *
* ELECTRONIC INC. Computer  products  plus the  Individual Reply  to my *
* Computer  problems for a  special of $10 each. Also I understand that *
* there  will  be  no  charge 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