Dynamic Color News

Radio Shack Color Computer
Magazine

$1.95

APRIL 1988
ISSUE # 48

Easter

PROGRAMS

OS-9

FORTH

HAM RADIO

DYNAMIC COLOR NEWS is published monthly by DYNAMIC ELECTRONICS, INC., P.O. Box 896, Hartselle, AL 35640, phone (205) 773-2758. Bill Chapple, BA, BSE President; Dean Chapple, Sec. & Treas. ; John Pearson, Ph. D. Consultant.
 Subscriptions are $18/yr for U.S.A. $20 Canada & Mexico, $30 other foreign.

The purpose of this magazine 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 magazine 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 magazine. 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 TXT file.

DYNAMIC COLOR NEWS

April 1988

Editor and Publisher
Bill Chapple W4GQC

Secretary
Dean Chapple

CONTENTS

* Included in DCN on Disk or Tape.

OS-9 Programs included in DCN on DISK.

The Marriage of Machine Language & Basic

by
John Galus

Basic is an excellent language to use for most computer applications but, sometimes it is just to slow for our programming needs. This is when Machine Language (ML) comes into play. Although, as many of us have discovered, Assembly language is difficult to learn, even if a programmer has a modest knowledge of Assembly language. It's possible to take advantage of the speed of ML by creating useful subroutines and linking them to our Basic programs. The hard part of using this method is understanding how to link the two programs together and executing them using the Basic commands that give us this capacity.

There are two ways that we can execute a ML routine from a Basic program. One is using the EXEC command and the other is the USR command. We use the EXEC when we want to Execute a ML routine or program directly. The routine may return to the calling program or not, depending on the type of program we are running. Usually, after loading a ML program we execute it causing the ML program to take control of the computer. We can also use the EXEC command to "call" a stand alone ML subroutine from Basic that does not require parameters from the calling program.

```
EXEC 44539
```

By executing the above ROM routine the computer will wait for a keypress. This call can take the place of the Basic line.

```
10 IF INKEY$="" THEN 10
```

A parameter can be thought of as data or information needed by or obtained from a subroutine. The number, or numbers, needed from a calling program by a subroutine to perform its function will be called the Input parameters and the data that we obtain from this subroutine at the exit of this routine will be known as the Output parameter. The Calling program is simply the Basic program in which the subroutine was called or executed.

The other method of calling ML subroutines in Basic is using the USR instruction. The USR command allows us to have up to ten seperate ML routines in our Basic program. The first thing we must do after creating a ML routine, which we wish to call from our Basic program, is to decide where we want it to reside in memory. Usually we place the subroutine high in memory. Make sure that the routine will fit in memory by testing it using the in memory function of your Editor/Assembler. Once we have a ML routine and have decided where to place it

in memory, we need to reserve a area of memory for it using the CLEAR command so that the routine does not get written over by our Basic program. If our routine were located starting at HEX $7F00 then we could do the following using the CLEAR command to protect the program:

```
1 CLEAR 200,&H7EFF
```

Remember to clear at least one memory position less than the address of the start of the ML subroutine. In the above example, the CLEAR specifies that the highest Basic address will be &H7EFF.

A USR function ML routine can be loaded from cassette or disk using the CLOADM or LOADM commands. We could also place the subroutine into memory by putting the routine into DATA statements and Poking it into memory (See Basic Listing 1). Once the program is in memory we must define its starting location by using the DEFUSR or Define User function. Since we have ten possible subroutines to specify, we place the number, 0-9 after the DEFUSR statement to define the entry address of the subroutine we want to use. For example, if we wished to place a subroutine for the USR function zero in memory starting at HEX $7F00 we would write the DEFUSR instruction as follows.

```
DEFUSR0=&H7000
```

Now whenever the USR command is found in our Basic program the ML routine starting at the address specified by the corresponding DEFUSR instruction will be executed. If we wish to return to Basic after executing this routine called by the USR function the subroutine MUST end with a RTS (Return) or jump to a routine that ends with this instruction. The USR function which calls a ML routine takes the following format.

```
Z = USR0(0)

OR Z$ = USR0("A")
```

The number after the USR can be any number from 0 to 9, corresponding to the routine that you are executing. The value held in the ( ) can be either a number or string and is called the "argument". This argument is passed to the ML routine. The result on the exit of this routine is assigned to the variable placed before the equal sign. A numeric argument must be assigned to a numeric variable and a string argument must be assigned to a string variable. If the variable is not the same, a "type mismatch" error will occur. If the Input or Output parameter is not required, then these parameters in the USR function are termed "dummy" arguments, since they are required for the syntax of the USR function but, not needed by the ML. We usually just place a zero in the USR argument when it is not needed (see Listing 1).

Some ML routines requires that one or more parameters are to be passed to it from the Basic calling program. One two byte parameter or argument can be passed with the USR function. This argument can be a numeric or string argument. If the argument is numeric, when a USR function is executed the X register is pointed to the Floating Point Accumulator (FPAC) that holds the numeric argument. This FP (Floating Point) value can be used to perform an arithmetic function, as in Assembly Listing 1/Basic Listing 2. Since an Assembly language routine cannot deal directly with FP numbers a numeric argument can be accessed by a routine by calling the ROM subroutine at $B3ED named INTCNV (see Assembly listing 2). This routine converts the number in the FPAC to an integer and places it in the D register. The numeric argu-

"The WIZARD'S CASTLE" is a very special 'TANDY' 'Color Computer' magazine. We devote our entire magazine to the 'CoCo family'. Our articles include columns like: "Wizard's Corner", "Letters to the Editor", "Questions for the Wizard", "Pencil-Pals","Wizard's Castle Scoreboard","Word Search", "Post-It-Notes", "Programmers Corner", "Software Reviews", "Hardware Reviews", "Doctor CoCo", "Hardware Modifications", "Adventure Hints", and "BBS UpDates". If you have been looking for a smaller more 'PERSONAL' version of a CoCo 'MAG' then we're 'EXACTLY' what you've been looking for. Remember we're exclusively for owners of any of Tandy's Color Computers. We support CoCo's 1, 2, and 3.

```
**********************************************
*  Please send me 12 POWER PACKED issues     *
*   of:  "The WIZARD'S CASTLE" magazine      *
*                                            *
*..... USA ONLY!!   for only $10.00 per yr   *
*                                            *
*..... Canada ONLY! for only $15.00 per yr   *
*                                            *
*..... Overseas     for only $30.00 per yr   *
**********************************************

Name:.....

Address:.....

City:.............St:.....Zip:.....

Phone:.(.....).....-.....
```

```
  FOR     **************************
          * Please mail orders to:  *
 RUSH!    *                        *
          * "The WIZARD'S CASTLE"   *
 ORDER    *       Dept. #18        *
          *  1737 Farmville Road   *
  SEE     *   Shelby, NC  28150    *
          **************************
 BELOW
```

```
#####################################
# We now have a 'BBS' devoted to    #
# 'Uploading' articles into our     #
# offices.  You can place orders    #
# here also.  BBS# (704)434-2629    #
#####################################
```

Are you always running out of disks???? Do you dread paying $1.50 or more per disk? Now, the answer to all of your disk problems can be answered here at the 'CASTLE'. We are offering only premium quality DSDD diskettes at under $1.00 each. These are 'TOP QUALITY' guaranteed disks. If you ever get a bad one, just send it back for replacement. We will sell these disks in quantities of 10, 25, 50, or even 100 per pk. Prices each will be cheaper depending upon amount purchased. This brand of disks is one all of our members have used for over four years. We hardly ever find '1' bad disk for every purchase we make. If you are tired of paying top dollar for disks then try just one order from us. You don't have anything to lose and only $$MONEY$$ to save.

```
**********************************************
*           Please send me:                  *
*                                            *
*..... 10 DSDD Disks for only $ 7.50 total   *
*                                            *
*..... 25 DSDD Disks for only $15.00 total   *
*                                            *
*..... 50 DSDD Disks for only $25.00 total   *
*                                            *
*.....100 DSDD Disks for only $45.00 total   *
**********************************************

Name:.....

Address:.....

City:.............St:.....Zip:.....

Phone:.(.....).....-.....
```

```
  SAVE!
          **************************
$$MONEY$$ *                        *
          * Please mail orders to:  *
 ORDER    *                        *
          * "The WIZARD'S CASTLE"   *
  THE     *    Disks Dept. #18     *
          *  1737 Farmville Road   *
 DISKS    *   Shelby, NC  28150    *
          *                        *
 TODAY!!  **************************

 NOTICE!
#####################################
# We now have a 'BBS' devoted to    #
# 'Uploading' articles into our     #
# offices.  You can place orders    #
# here also.  BBS# (704)434-2629    #
#####################################
```

ment in the Basic USR command can be a number itself or a Basic variable.

```
10 Z=USR0(100)

10 Z=USR0(A)
```

Using the INTCNV ROM routine a numeric argument can only be an integer number from +32,767 to -32768. If it is a string, the argument the X register will point to is a five byte string descriptor block, where the first byte contains the length of the string, the third and fourth byte contains the address that points to the start of the string in memory, and the second and fifth byte are unused.

```
Z$=USR0("DYNAMIC COLOR NEWS")
```

In Assembly Listing 2, we use the information in the string descriptor block to print a space after each letter of the string sent to the ML routine, so that it is printed on the screen as follows (See Assembly listing 3/Basic listing 3):

```
D Y N A M I C  C O L O R
```

You cannot use the INTCNV routine if the argument is a string. If we do we will get a type mismatch error.

To return a numeric argument from a machine language routine to Basic, we load the 16 bit D register with the value we want to pass and then call the ROM subroutine located at $B4F4 called GIVABF. This routine converts the value in the D register to FP format and places it in the FPAC, which is stored into the variable in the USR function before the equals sign (see Basic listing 4, Assembly listing 3).

Sometimes we might need to pass a parameter to a subroutine that is greater then +32,767. We do this by tricking the USR function by placing the following lines in our Basic program before the USR call.

```
100 A=B
110 IF B>32767 THEN A=A-65536
120 Z=USR0(A)
```

Parameters can also be sent to a calling routine by poking the values into a memory area reserved for them.

If we need to find the location of a Basic numeric or string variable we use the Extended Basic VARPTR function.

For example:

```
A=VARPTR(B) A=VARPTR(A$)
```

In the above, the variable A will contain the location of the variable B and the location of the string descriptor block for the string A$. This location can be sent by the USR and used directly by calling the ROM routine INTCNV at $B3ED. A string variable can be located in the Basic program itself or the string stack. It's important to remember that VARPTR locations can move in memory so, it must be used just before the USR function to work correctly.

We can use this VARPTR function to pass arguments placed in a string. Since any character in a string can be of a value of 0 to 255, using the CHR$ command we place arguments into a dummy string and then pass the strings descriptor block location to the ML which accesses the parameters in the string. I created a GET/PUT like routine that passes four parameters within a string in Assembly Listing 4. In the Basic program, Listing 5, I use this routine to animate a simple figure. We can place the ML routine itself within a string and execute it using the VARPTR command. Using this method to place ML routines in strings can save us time and memory since, the program lines that contain the data and required to poke the routine into memory can be

eliminated after it has been placed in the string (see Basic Listing 6 & 7).

There are a few restrictions when using this method you should keep in mind.

1.) ROUTINES MUST BE UNDER 256-BYTES LONG.
2.) ROUTINE MUST BE ABLE TO RESIDE ANYWHERE IN MEMORY, RELOCATABLE.
3.) ROUTINE MUST NOT USE ZERO, WHICH SIGNIFIES END OF BASIC STATEMENT, 34 QUOTES OR 13 CARRIAGE RETURN.

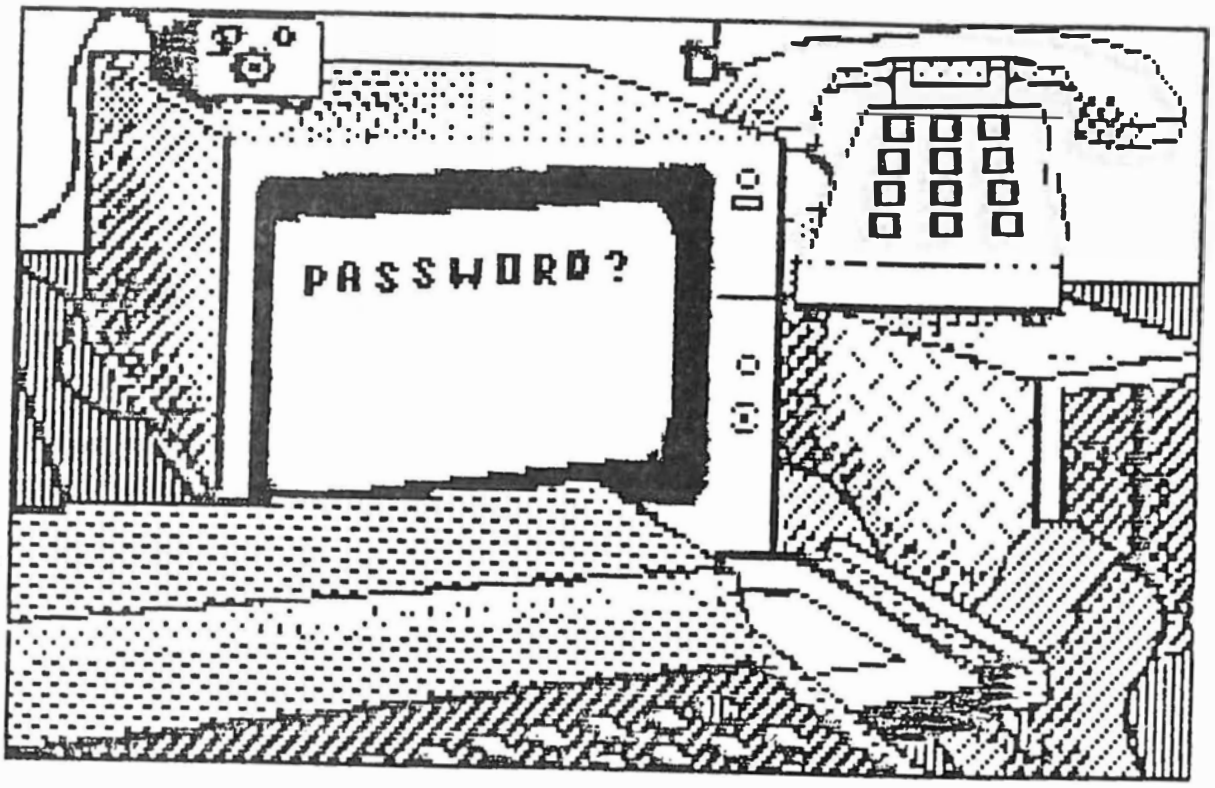
Be sure to place enough characters in your string to hold the routine.

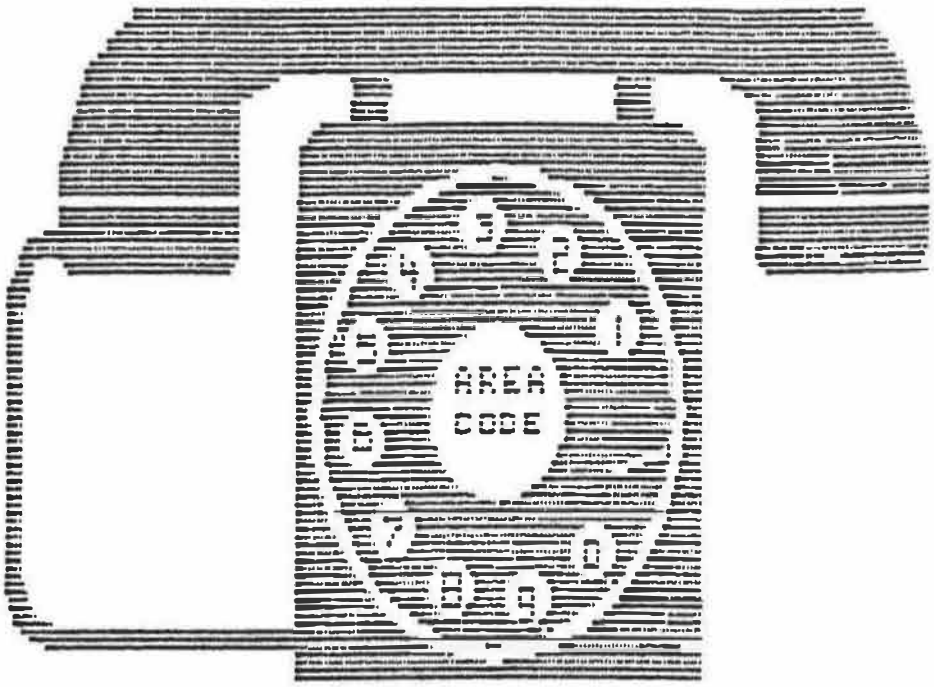
Another technique we can use is to place the routine in a REM statement within the first line of your Basic program (see Basic Listing 8). The memory locations 25 and 26 hold the address of the first Basic line in the program. We use this number and add six to skip the line number and the REM instruction. After stuffing a routine into a line using this method you can get rid of the extra lines as we did in the last example.

Examine how these techniques work and see if you can come up with some interesting routines using this method by yourself.

```
00010 ***************************
00020 *
00030 * ASSEMBLY
00040 * LISTING #1
00050 *
00060 * BY JOHN GALUS
00070 *
00080 *    FOR
00090 * DYNAMIC COLOR
00100 * NEWS MAGAZINE
00110 *
00120 ***************************
00130 *
00140 * SQUARE ROUTINE
00150 * X POINTS TO FPAC THAT
00160 * HOLD NUMBER TO MULTIPLY
00170 * THIS ROUTINE USED ROM
00180 * MATH ROUTINES
00190 * FPAC1*FPAC2
00200 * RESULT IS LEFT IN FPAC1
00210 *
00220 MATH JSR $BC5F ;FP1 - FP2
00230  JSR $BACC ;FPAC1*FPAC2
00240  RTS  ;RETURN
00250  END MATH
00260 ***************************
00270 *
00280 * ASSEMBLY
00290 * LISTING #2
00300 *
00310 * BY JOHN GALUS
00320 *
00330 *    FOR
00340 * DYNAMIC COLOR
00350 * NEWS MAGAZINE
00360 *
00370 ***************************
00380 *
00390 * THIS ROUTINE EXPANDS
00400 * A STRING PASSED BY USR
00410 * FUNCTION
00420 *
00430 STR LDB ,X ;GET LEN STR$
00440  TSTB ;TEST FOR ZERO LEN
00450  BEQ FIN ;IF 0 LEN RETURN
00460  LDY 2,X ;POINT TO STRING
00470 LOOP LDA ,Y+ ;GET A CHAR
00480  JSR $A282  ;PRT IT
00490  LDA #32 ;LOAD A SPACE
00500  JSR $A282 ;PRINT IT
00510  DECB  ;LEN=LEN-1
00520  BNE LOOP ;NOT DONE LOOP
00530 FIN RTS  ;RETURN
00540  END STR
00550 ***************************
00560 *
00570 * ASSEMBLY
00580 * LISTING #3
00590 *
00610 *
00620 *    FOR
00630 * DYNAMIC COLOR
00640 *  MAGAZINE
00650 *
00660 ***************************
00670 *
00680 * FIND SQUARE ROOT
00690 * USING ROM ROUTINES
00700 * TO PASS ARGUMENTS
00710 *
00720 INTCNV EQU $B3ED ;# TO D
00730 GIVABF EQU $B4F4 ; # BACK
00740 SQR JSR INTCNV ;GET ARG
00750  LDX #-1 ;INITIAL SQR
00760  LDU #1 ;START ODD NUMBER
00770  PSHS U ;SAVE IT ON STACK
00780 LOOP LEAX 1,X ;+1 TO SQR
00790  LDY ,S ;GET # FROM STACK
00800  LEAY -2,Y ;SUBTRACT TWO
00810  STY ,S ;BACK TO STACK
00820  ADDD ,S ;SUBTRACT FROM D
00830  BCS LOOP ;NOT MINUS LOOP
00840  TFR X,D ;PUT SQR IN D
00850  PULS U ;GET # ON STACK
00860  JMP GIVABF ;RETURN ARG
00870  END SQR
00880 ***************************
00890 *
00900 * LISTING #4
00910 *
00920 * GET/PUT
00930 * ROUTINES
00940 *
00950 * BY JOHN GALUS
00960 *
00970 ***************************
00980 *
00990 * GET/PUT ROUTINE
01000 *
01010 * 1ST BYTE X POSITION
01020 * 2ND BYTE Y POSITION
01030 * 3RD BYTE PLAYER NUMBER
01040 * 4TH BYTE GET/PUT FLAG
01050 *
01060  ORG $7000
01070 PAR EQU $B3ED ;GET VALUE
01080 POS EQU $9298 ;CAL X/YPOS
01090 GET JSR PAR ;GET STR$ #
01100  TFR D,Y ;PUT IN Y
01110  LDB ,Y ;GET LENGTH
01120  CMPB #4 ;FOUR PARAMETERS
01130  BNE FIN;BAD SYNTAX
01140  LDU 2,Y ;GET LOCATION
01150 CHAR LDA ,U+ ;GET X/Y POS
01160  STA $BE ;X POS
01170  LDA ,U+ ;Y POS
01180  STA $C0
01190  PSHS U ;SAVE U
```

```
01200  JSR POS ;CAL X/Y POS
01210  PULS U
01220  TFR X,Y ;PUT IN Y
01230  LDX #SAVE ;TO SAVE AREA
01240  LDA ,U+ ;GET PLAYER #
01250  LDB #12 ;12 BYTES HIGH
01260  MUL
01270  ABX  ;ADD OFFSET
01280  LDB #12 ;DO 12 TIMES
01290  LDA ,U ;GET OR PUT
01300  CMPA #1 ;DET=0/PUT=1
01310  BNE LOOP
01320  BRA PUT ;GOTO PUT
01330 LOOP LDA ,Y ;GET A BYTE
01340  STA ,X+ ;SAVE IN MEMORY
01350  LEAY 32,Y
01360  DECB  ;ONE LESS
01370  BNE LOOP
01380 FIN RTS  ;RETURN IF END
01390 PUT LDA ,X+ ;GET A BYTE
01400  STA ,Y ;PUT IT ON SCREEN
01410  LEAY 32,Y ;UP ONE LINE
01420  DECB  ;ONE LESS
01430  BNE PUT
01440  RTS
01450 SAVE RMB 100 ;SAVE AREA
01460  END GET
```





AREA CODE

This telephone application program can be used to find area codes all over the United States and Canada. A state can be typed in and the major cities with their area codes will be displayed. Also if the 3 digit area code is entered then the state for the code will be printed. This program is provided as a courtesy of T & D Subscription Software (See their ad on page 8) and is used by permission.

```
1 'AREACODE (C)1988 T&D SOFTWARE
2 CLS:PRINT"WHEN SEARCHING BY ST
   ATE/AREA,  INPUT EITHER THE
   WHOLE NAME OF ASTATE OR CANAD
   IAN PROVENCE AND  THE PROGRAM
    WILL BREAK IT DOWN  INTO SPE
   CIFIC AREAS.
3 PRINT"IF YOU WANT TO SEARCH BY
    AREA  CODE, SIMPLY INPUT TH
   E 3-DIGIT  NUMBER AND THE PRO
   GRAM WILL TELLYOU WHICH AREA
   IS COVERED.
4 PRINT@484,"HIT ANY KEY TO CONT
   INUE"::EXEC44539
5 AA$="BR3U3ERFDNL3D2
6 CC$="BR4REGLHU2ERFBD3
7 DD$="BR3U4R2FD2GL2BR3
8 EE$="BR3NR3U2NR2U2R3BD4
9 OO$="BR3BUU2ERFD2GLNHBR
10 RR$="BR3U4R2FGL2RF2
11 PMODE4,1:PCLS5:SCREEN1,1:COLO
   R0,1
12 DRAW"BM70,160U100E5R8U9R9D9R5
   7U9R9D9R8F5D100L100BM65,45R11
   0FRFRFRFRF2D17NR42BD3NR42D10F
   2R38E2U10BU3NL42U5HU2HU2HU2HU
   2HU2HU2HULHULHULHULHULHULHUL160B
   M65,45GLGLGLGLG2D17NL42BD3NL4
   2D10G2L38H2U10BU3NR42U5EU2EU2
   EU2EU2EU2EU2EUREUREUREUREUREU
   R40
13 CIRCLE(120,110),44,0:CIRCLE(1
   50,94),7,0:CIRCLE(135,80),7,0
   :CIRCLE(116,75),7,0:CIRCLE(98
   ,83),7,0:CIRCLE(87,97),7,0:CI
   RCLE(86,115),7,0:CIRCLE(94,13
   2),7,0:CIRCLE(109,143),7,0:CI
   RCLE(128,145),7,0:CIRCLE(144,
   136),7,0:CIRCLE(120,110),18,0

14 DRAW"BM107,106"+AA$+RR$+EE$+A
   A$:DRAW"BM107,114"+CC$+OO$+DD
   $+EE$
```

```
 15 DRAW"BM162,115G2L3GL3GL3GDFR3
    ER3ER3ER2":PAINT(65,40),0,0:P
    AINT(40,80),0,0:PAINT(220,80)
    ,0,0
 16 CIRCLE(120,110),48,0:DRAW"BM1
    50,92D5BM134,78R3D2L3D3R3BM11
    5,73R3D2NL2D3L3BM96,81D3R4LU2
    D4BM85,95NR4D2R4D3L4BM84,112N
    R3D5R4U3L4BM92,130R4DG3DBM107
    ,140R4D3NL4D3L4U6BM126,143ND3
    R3D3NL3D3L3BM142,134R3D5L3U5"
    :PAINT(120,85),0,0:PAINT(72,1
    58),0,0
 17 DRAW"BM12,80L3GLGD2GD2GD2GD50
    FDFDFDF2RFRFR58D3L59LHLHLHLH2
    UHUH2UHU55EUEUEUEUERERER4":PA
    INT(10,78),0,0
 18 FOR X=1TO2000:NEXTX
 19 CLS:PRINT@66,"AREA CODE - LOC
    ATION FINDER
 20 PRINT@135,"FROM T&D SOFTWARE
 21 PRINT@169,"(616)399-9648
 22 PRINT@202,"^^^
 23 R$="                          H
    IT ANY KEY TO CONTINUE
       ":FOR Y=1 TO LEN(R$)-32:PR
    INT@484,MID$(R$,Y,23);:EXEC 4
    3345:NEXT Y:FOR U=1508 TO 153
    0:POKE U,PEEK(U)-64:EXEC 4334
    5:NEXT U:EXEC 44539
 24 GOSUB 63
 25 CLS:INPUT"STATE OR AREA:";ST$
 26 PRINT@66,"area":POKE1094,32:P
    RINT@71,"codes":PRINT@79,"are
    as":POKE1108,32:PRINT@85,"cov
    ered"
 27 PRINT"";
 28 FOR Z=1 TO 150
 29 READ AC$,RM$,PL$
 30 IF AC$="*"THEN 34
 31 AC=VAL(AC$)
 32 IF ST$=PL$ THEN SOUND 191,1:P
    RINT TAB(8)AC;"- ";TAB(15)RM$
 33 NEXT Z
 34 RESTORE:SOUND 50,2
 35 GOSUB 63
 36 CLS:INPUT"AREA CODE:";AA$
 37 PRINT@64,"area":POKE1092,32:P
    RINT@69,"covered":PRINT@82,"s
    pecific":POKE1114,32:PRINT@91
    ,"area
 38 PRINT"";
 39 FOR Z=1 TO 150
 40 READ AC$,RM$,PL$
 41 IF PL$="*"THEN 44
 42 IF AA$=AC$ THEN SOUND 191,1:P
    RINT PL$;TAB(18)RM$
 43 NEXT Z
 44 RESTORE:SOUND 50,2
 45 GOSUB 63
 46 DATA 201,NEWARK,NEW JERSEY,20
    2,,DISTRICT OF COLOMBIA,203,A
    LL,CONNECTICUT,204,ALL,MANITO
    BA,205,ALL,ALABAMA,206,SEATTL
    E,WASHINGTON,207,ALL,MAINE,20
    8,ALL,IDAHO,209,FRESNO,CALIFO
    RNIA,212,NEW YORK CITY,NEW YO
    RK,213,LOS ANGELES,CALIFORNIA

 47 DATA 214,DALLAS,TEXAS,215,PHI
    LADELPHIA,PENNSYLVANIA,216,CL
    EVELAND,OHIO,217,SPRINGFIELD,
    ILLINOIS,218,DULUTH,MINNESOTA
    ,219,SOUTH BEND,INDIANA
 48 DATA 301,ALL,MARYLAND,302,ALL
    ,DELAWARE,303,ALL,COLORADO,30
    4,ALL,WEST VIRGINIA,305,MIAMI
    ,FLORIDA,306,ALL,SASKATCHEWAN
    ,307,ALL,WYOMING,308,NORTH PL
    ATTE,NEBRASKA,309,PEORIA,ILLI
    NOIS,312,CHICAGO,ILLINOIS,313
    ,DETROIT,MICHIGAN,314,JEFFERS
    ON CITY,MISSOU
 49 DATA 315,SYRACUSE,NEW YORK,31
    6,WICHITA,KANSAS,317,INDIANAP
    OLIS,INDIANA,318,SHREVEPORT,L
    OUISIANA,319,DUBUQUE,IOWA,401
    ,ALL,RHODE ISLAND
 50 DATA 402,LINCOLN,NEBRASKA,403
    ,ALL,ALBERTA,403,ALL,NORTHWES
    T TERRITORIES,403,ALL,YUKON,4
    04,ATLANTA,GEORGIA,405,OKLAHO
    MA CITY,OKLAHOMA,406,ALL,MONT
    ANA,408,SAN JOSE,CALIFORNIA,4
    09,BEAUMONT,TEXAS,412,PITTSBU
    RGH,PENNSYLVANIA,413,SPRINGFI
    ELD,MASSACHUSE
 51 DATA 414,MILWAUKEE,WISCONSIN,
    415,SAN FRANSISCO,CALIFORNIA,
    416,TORONTO,ONTARIO,417,SPRIN
    GFIELD,MISSOURI,418,QUEBEC,QU
    EBEC,419,TOLEDO,OHIO
 52 DATA 501,ALL,ARKANSAS,502,LOU
    ISVILLE,KENTUCKY,503,ALL,OREG
    ON,504,BATON ROUGE,LOUISIANA,
    505,ALL,NEW MEXICO,506,ALL,NE
    W BRUNSWICK,507,ROCHESTER,MIN
    NESOTA,509,SPOKANE,WASHINGTON
    ,512,SAN ANTONIO,TEXAS,513,CI
    NCINNATI,OHIO,514,MONTREAL,QU
    EBEC
 53 DATA 515,DES MOINES,IOWA,516,
    HEMPSTEAD,NEW YORK,517,LANSIN
    G,MICHIGAN,518,ALBANY,NEW YOR
    K,519,LONDON,ONTARIO,601,ALL,
    MISSISSIPPI,602,ALL,ARIZONA
 54 DATA 603,ALL,NEW HAMPSHIRE,60
    4,ALL,BRITISH COLOMBIA,605,AL
    L,SOUTH DAKOTA,606,COVINGTON,
    KENTUCKY,607,BINGHAMPTON,NEW
    YORK,608,MADISON,WISCONSIN,60
    9,TRENTON,NEW JERSEY,612,MINN
    EAPOLIS,MINNESOTA,613,OTTAWA,
```

```
   ONTARIO,614,COLUMBUS,OHIO
55 DATA 615,NASHVILLE,TENNESSEE,
   616,GRAND RAPIDS,MICHIGAN,617
   ,BOSTON,MASSACHUSETTS,618,WES
   T FRANKFORT,ILLINOIS,619,SAN
   DIEGO,CALIFORNIA,701,ALL,NORT
   H DAKOTA
56 DATA 702,ALL,NEVADA,703,ROANO
   KE,VIRGINIA,704,CHARLOTTE,NOR
   TH CAROLINA,705,NORTH BAY,ONT
   ARIO,707,EUREKA,CALIFORNIA,70
   9,ALL,NEWFOUNDLAND,712,COUNCI
   L BLUFFS,IOWA,713,HOUSTON,TEX
   AS,714,RIVERSIDE,CALIFORNIA,7
   15,EAU CLAIRE,WISCONSIN,716,B
   UFFALO,NEW YOR
57 DATA 717,HARRISBURG,PENNSYLVA
   NIA,718,NEW YORK CITY,NEW YOR
   K,800,ALL,TOLL FREE SERVICE,8
   01,ALL,UTAH,802,ALL,VERMONT,8
   03,ALL,SOUTH CAROLINA
58 DATA 804,RICHMOND,VIRGINIA,80
   5,BAKERSFIELD,CALIFORNIA,806,
   AMARILLO,TEXAS,807,THUNDER BA
   Y,ONTARIO,808,ALL,HAWAII,809,
   ALL,BAHAMAS,809,ALL,PUERTO RI
   CO,809,ALL,VIRGIN ISLANDS,812
   ,EVANSVILLE,INDIANA,813,TAMPA
   ,FLORIDA,814,ERIE,PENNSYLVANI
   A
59 DATA 815,ROCKFORD,ILLINOIS,81
   6,KANSAS CITY,MISSOURI,817,FT
   .WORTH,TEXAS,818,PASADENA,CAL
   IFORNIA,819,SHERBROOKE,QUEBEC
   ,900,ALL,SPECIAL SERVICES
60 DATA 901,MEMPHIS,TENNESSEE,90
   2,ALL,NOVA SCOTIA,902,ALL,PRI
   NCE EDWARD I.,903,ALL,NORTHWE
   ST MEXICO,904,JACKSONVILLE,FL
   ORIDA,905,ALL,MEXICO CITY,906
   ,ESCANABA,MICHIGAN,907,ALL,AL
   ASKA,912,SAVANNAH,GEORGIA
61 DATA 913,TOPEKA,KANSAS,914,WH
   ITE PLAINS,NEW YORK,915,ABILE
   NE,TEXAS,916,SACRAMENTO,CALIF
   ORNIA,918,TULSA,OKLAHOMA,919,
   RALEIGH,NORTH CAROLINA
62 DATA *,*,*
63 PRINT@480,"SEARCH: sTATE/AREA
    OR aREA CODE";
64 A$=INKEY$:IF A$=""THEN 64
65 IF A$="S"THEN 25
66 IF A$="A"THEN 36
67 GOTO 64
68 RETURN
```

OS-9 & BASIC09

An operating system is designed to make operations easier on the computer. The basic that comes with color computers is an operating system that is dedicated to using basic commands. It is "burned" into programmable read only chips or PROMS and the computer automatically configures itself for basic when power is applied. Only the commands that are contained within the PROM can be executed by the computer when basic is loaded.

If more flexibility is desired a separate operating system can be loaded into the computer's memory. The computer can then be instructed to take the commands given by the operating system. IBM compatible computers do not have an operating system. When the computer is turned on, an operating system has to be loaded from a disk. This takes more time but adds the flexibility of using a number of different operating systems or programs with their own instructions.

There have been several operating systems for color computers. A very good and popular operating system is FLEX which was marketed by Frank Hogg Labs. It has been discontinued. OS-9 is the operating system marketed by Radio Shack and it is gaining popularity. An operating system generally handles files and input/ output tasks. However a computer's power is in its calculating ability. OS-9 does not allow calculations and a version of basic or machine language programs must be used with OS-9 for calculations. Unfortunately the version of basic that is in the PROMS is not recognized by OS-9. Because of this BASIC 09 was developed for color computers to be used with OS-9. The marriage of OS-9 and BASIC 09 makes a very good combination. As instructions are written using BASIC.09, they are compilled into machine language codes. This gives improved speed over Microsoft Basic.

80 COLUMN WINDOW

Each month we want to look at some commands for the OS-9 operating system and then look at BASIC 09. We are using OS-9 level 2 for the color computer 3. The color computer 3 has many advantages over the earlier versions. An 80 column display is one of the nicest features. When OS-9 is booted up a 32 character screen appears which is not very pleasing to the eyes. It would be nice to modify the operating system so that an 80 column reversed screen is available. To do this the startup file on the Basic 09 disk needs to be changed. To change a file we will need to do the following:

1. Rename the present file.
2. Create a new file.
3. Merge the new file.

Window 7 gives the 80 column display. To change the background color to black type:

```
display 1b 33 02
```

33 is the code for the background and 02 is the code for black.

To change the text color to white type:

```
display 1b 32 00
```

32 is the code for text and 00 is the code for white.

To create a black border type:

```
display 1b 34 02
```

34 is the code for the border and 02 is the code for black.

Now let's write the instructions.

```
rename startup aa ‹ENTER›
```

Now delete startup as follows:

```
del startup ‹ENTER›
```

aa will be our temporary file. Now build the new file and call it "xx".

```
build xx
iniz w7 ‹ENTER›
shell i=/w7 ‹ENTER›
display 1b 32 00 1b 32 02 1b 34
     02 0c › /w7 ‹ENTER›
‹ENTER›
```

Now merge aa with xx and call the new file startup.

```
merge aa xx › startup ‹ENTER›
```

List startup to make sure the additions are added. Now when DOS is typed from basic the 80 column window will be created. Press the clear key to access this window. If you are using a television then a 32 column window will probably be better. Use the same procedure to customize your window.

EDITING FILES

The OS-9 editor is different from the Microsoft basic editor. To enter the editor type

```
OS9: edit xx
```

where xx is the name of the file to be edited. The symbols "E:" indicates the editor mode is activated.

LISTING LINES

The lines of the file can be listed by pressing the "l" key and the number of lines to list. Note this is the lowercase "L" key and not a one. To list 5 lines press l5. To move to the end of the buffer press the "/" key. To move to the top of the buffer press the "-*" keys. To move backwards 5 lines enter -5. To move n characters to the right on the edit line type <n.

Inserting characters in a line is a little confusing. The change command has to be used for this. Suppose we have the following line:

```
This is atest program.
```

It is obvious that a space needs to be added between a and t. After the editor is brought up and the editor pointer is pointing to the line to be edited, type in the following:

```
c/at/a t/
```

To move n characters to the left on the edit line type >n. Let's list the most useful commands notice these involve the "*":

```
-* moves to top of buffer
+* moves to end of buffer
l* lists all lines
i  inserts new line at pointer
   or skip the first space and
   type in the line
c  allows character string to
   be changed. c/str1/str2/.
q  ends edit mode
```

BASIC 09

To get the 80 column reversed screen modify startup on the OS-9 disk as discussed in the previous section. For a one drive system the following file will create a path for a single drive and load basic09.

```
chx /d0/cmds
chd /d0
load basic09
```

A good name for the loader file is "go". This can be copied to the basic09 disk. To load basic09 just enter go. Then to run basic09 type Basic09 and the B: prompt will quickly appear. Programs written using basic09 are called procedures. As a command line is entered, it is compiled, and if there are any errors, they are displayed.

When writing a procedure it is not necessary to enter numbers for procedure lines. However if it is necessary to do any branching such as GO TO or GO SUB then numbers are required. It would be impossible to GO TO 10 if line 10 were not defined.

Basic09 is not as friendly as Microsoft's color basic. It falls between color basic and assembly. Each line is assembled as it is written. This makes it easy to spot errors. Also the error messages can be brought up from the error files. For a single disk drive enter the following:

```
OS9: load /d0/cmds/error
```

Now if an error number appears then enter error xx where xx is the error number. If you are in basic09 then enter $error xx. Remember the $ will link the command back to OS-9.

MEMORY PEEK PROGRAM

This month we wrote a memory peek program to demonstrate using the GOTO command. The procedure is called mem. To start the editing procedure enter:

```
B:edit mem
```

Next enter the lines as follows:

```
E: 10 ?"enter memory"
E: input m
E: a=peek(m)
E: a$=chr$(a)
E: ?"memory=";
E: ?m
E: ?"a="; a
E: ? a$
E: goto 10
E:q
```

After entering the lines they can be listed by typing list from the B: prompt. Also the program can be run from the B: prompt. Notice that the first space is skipped because it is reserved for commands. We used the command "q" to end the procedure. The listing of the program is as follows:

```
0000 10  PRINT "enter memory"
0013     INPUT m
0018     a=PEEK(m)
0023     a$=CHR$(a)
002D     PRINT "memory=";
0039     PRINT m
003E     PRINT "a="; a
0048     PRINT a$
004D     GOTO 10 Ready B:
```

The program only has one label which is 10. It allows memory to be looked at and displays the value in memory and the ASCII character of the value if it is a character that can be displayed on the screen. Remember that the main purpose of the program is to demonstrate the use of GOTO and the label or number 10 for the first line. This program is included on the back side of our DCN on disk.

Next month we will continue with more commands and programs. Since Basic09 complies each line as it is entered, its operation is similar to an assembler. However it uses basic commands which makes it a very useful programming language.

Editor's Comments

It is good to see Spring arrive. I am glad to see the warmer weather as we have had enough cold weather here in North Alabama.

Let me give some suggestions to beginners. A computer can be very discouraging if you are not familiar with how it operates. When a command is given, the computer has to know what instructions are associated with the command. It searches its memory and if it finds the command, then it executes the instructions associated with that command. If not, it will give an error message. This is especially confusing with OS-9 because all commands have to be loaded into memory. With extended basic and disk basic, commands are contained within a read only memory (ROM) chip and are available at all times. When an operating system has to be loaded into memory, it may not be feasible to load all commands into memory. This is true with IBM compatible computers using MS-DOS as well as Radio Shack Color computers using OS-9. Os-9 is not for beginners but can be very enjoyable for those wanting a challenge. I did not like OS-9 until I tried Basic 09. With this combination, basic programs can be written and managed with the OS-9 operating system.

Take things one at a time. A disk drive, printer, and modem all involve special commands. A disk drive has commands that are not required with cassette operation. Disks have to be formatted for the computer in use. Did you know that IBM XT disks will work on color computers? I use the same disks for both. There are also commands for copying files, creating files, backing up a disk, etc. A printer has special commands too. You can select italics, double size, double strike, emphasized modes, plus graphics. It takes time to learn how to use each of these. For modem use, the baud rate, parity, word lenght, and number of stop bits have to be selected.

Some people want to learn everything at the same time which could lead to frustration. I have the same problem because I am involved with many different things. I can only concentrate on one thing at a time and try to shut out other problems until I have finished the one I am addressing.

I have some very bad news for those of you who are planning to increase your computer's memory. The price of memory chips has trippled within the last few months. A couple of years ago American semiconductor manufacturers filed a suit against Japanese semiconductor manufacturers. From what I have read, the Japanese were dumping memory chips on the U.S. market at prices below their cost. This had the effect of driving most U.S. manufacturers out of the memory chip business. Because of the suit, a tarriff was placed on these chips and the result is excessively high priced memory chips for U.S. consumers.

I just purchased chips for a 512K upgrade for a color computer 3 and paid $8.40 for each one. With 16 chips being required the order was in the $130 range. We had been selling the complete upgrade for $89.95. You can expect to pay in the range of $200 for a 512K upgrade if the trend continues.

Prices will of course drop again but not until this price spiral runs out. Of course the higher prices will slow demand and with United States manufacturers gearing up for production again maybe this crisis will ease soon.

We occasionally sell an IBM clone and have been offering 640K units. Now we will be offering 256K units unless our customers are willing to pay for 640K units. Of course 256K units can be upgraded later and will run most programs. A local church is interested in purchasing a computer. They have an Apple but are interested in an IBM compatible computer. It would be hard to beat a color computer for handling their records, letters, and accounting. However many churches use IBM compatible computers and the ability to exchange programs is a great asset. There are numerous public domain programs available for IBM compatible computers which makes them attractive from that standpoint. However there are also numerous public domain programs for the color computers. We have a growing collection and there are many others. This is a good and inexpensive way to increase your software capability. Sometimes public domain software is not easy to use because there are no instructions. Instructions are included with some packages as "DOC", "TXT", or "DAT" files. These can be read into a word processor and the instructions printed on a printer. Our "LOADER/BAS" program included with each DCN on disk or tape will read these files and print them on the screen or to a printer.

We still need names of potential subscribers. I want to thank those of you who have sent in names or requested copies to pass out at a club. If you can help with this please let us know. Also if you are a member of a computer club, own or have access to a bulletin board, or want a pen-pal, please fill out the information on the tear out sheet and send it back to us.

Taking Control
(Basic Programming Part 7)

This is a series on basic programming. Each month we cover a few programming commands and then give example programs for using them. Last month we discussed using edit commands. These commands are useful for correcting basic statements with errors or for adding additional commands to a statement. Commands are seperated in a line by using a colon (:). This approach saves memory because only one byte is required to seperate commands. If a seperate line number is used for each command, then 5 bytes are required. A line can contain many statements and if an error is detected then the edit commands can be used to correct the line eliminating the need to retype the line.

Let's suppose we want to edit line 325. Then we would type:

EDIT 325 ‹ENTER›

Line 325 will appear on the screen with the cursor on the first character. Forward movement of the cursor is accomplished by pressing the space bar or a number and then the space bar. For example to move forward 25 spaces enter:

25 sp where sp means to press the space bar.

The cursor is moved backwards by pressing a number and the left arrow. By just pressing the left arrow key the cursor moves back one character.

To delete a character just press the "D" key for each character to delete. This is generally easier than counting the characters, entering a number and pressing the "D" key.

To insert characters, move the cursor to the location for the insertion and press the "I" key. Characters can then be inserted. To terminate this procedure hold down the shift key and press the up arrow key. Then press the "L" key to list the line.

To extend a line press the "X" key. This moves the cursor to the end of the line and allows characters to be inserted. To exit and remain in the edit mode, hold down the shift key and press the up arrow. Then press the "L" key to return to the first character and list the line.

Being able to edit basic programs is necessary if a program has a problem. Some people are not interested in programming but a knowlege of editing com-

mands can be used by them to correct programs with errors.

An easy way to write or modify basic programs is to use a word processor that can handle ASCII files. The program to be edited should be loaded into the computer and saved as an ASCII file. To do this add a ",A" after the program name.

```
SAVE "FIRST",A
```

The word processor can then read in the program as an ASCII file. Modifications can be made and the program can the be saved with the /BAS extension. Before saving the program make sure that each line number is the beginning of a line. For long lines, the word processor will place commands on the next text line. Basic will only recognize numbers at the beginning of a line. You can usually move from one line to the next with the arrow keys depending upon the word processor. The word processor can also be used to write basic programs. This is a quick way to spot errors.

COMMANDS

The sound command is easy to use and can be used for many purposes. For converting data, a sound can be made to alert the operator when the computer is finished. It can also be used as sound an alarm or to add excitement to games. The sound command requires two parameters which are the tone and duration. An example is SOUND 150,5. The tone is 150 and the duration is 5. The format is:

```
SOUND tone, duration
```

Notice that a comma seperates the arguments. The frequency of the sound increases as the first number increases. SOUND 200,1 is higher in pitch than SOUND 50,1. The tone and duration can vary from 1 to 255. Fortunately sound is a quantity that we can physically hear if we are not deaf. Let's write a simple sound demonstration program.

```
5 'CONT-1
10 ?"SOUND DEMONSTRATION PGM
20 ?"THIS GENERATES A SOUND AND
   DISPLAYS THE NUMBER FOR THE T
   ONE
30 ?"A FOR-NEXT LOOP WILL BE USE
   D TO CHANGE THE TONE VALUES.
32 ?"THE STEPS CAN BE CHANGED
35 INPUT"ENTER STEPS";S
37 ?"INCREASING THE TONE
40 FOR J=1 TO 255 STEP S
50 SOUND J,1
60 ?"TONE NUMBER IS "J
70 NEXT J
75 ?"DECREASING THE TONE
80 FOR J=255 TO 1 STEP -S
90 SOUND J,1
100 ?"TONE NUMBER IS "J
110 NEXT J
120 ?"NOW INPUT TONE NUMBERS"
130 INPUT"TONE NUMBER";N
140 SOUND N,1
150 GOTO 130
160 '
```

USING PEEKS AND POKES

These powerful commands allow us to store values in memory and recover them. The lower memory locations contain vectors or pointers that point to other memory locations. For example the values in 150 and 151 determine the baud rate for a printer. The values in 136 and 137 determine the location of the cursor on the screen. A memory location contains a byte which can only represent a value from 0 to 255. Two bytes are required for numbers greater than 255. The lower byte is called the most significant and its value is multiplied by 256. The upper byte is called the least significant. This value is added to the most significant to determine the value of the two bytes. Let's look at a demonstration program for

DOES THIS NEWS JUST TICKLE YOUR EAR?

REMUSIC 1.0 ♪♪ is here! ♪♪

REMUSIC is a versatile, programmable music synthesis software package requiring only a 16K disk-based CoCo(32K recommended). Enter musical data as BASIC program REM statements -- no graphic interface is used. Once loaded, REMUSIC keeps a low profile as you roll merrily along in BASIC. Here are a few of REMUSIC's powerful features:

*Four independent voice tracks *Definable and dynamically assignable tone waves and amplitude envelopes, allowing such effects as decay, vibrato, and tremolo *Music plays through TV or monitor speaker *Nestable multiple music repeats with the option of different endings *Key signature and two different transposition modes *Ability to mathematically link repeat iterations with such functions as transposition, octave, tempo, waveform assignment and subordinate repeats and endings *Ability to convert music to independent machine-language programs for use in your own programs *Since music source files are actually BASIC programs, they can be edited and merged as other BASIC programs can.

Package includes program diskette, sample music files, and documentation.

Send check or money order for $25.00* to:

CODIS ENTERPRISES
2301-C CENTRAL DR., STE. 684
BEDFORD, TX 76021

*Texas residents add $1.75 sales tax.

RAINBOW CERTIFICATION SEAL

Get a REMUSIC demo for $2.00

Since the benefits of REMUSIC 1.0 must be heard to be appreciated, CODIS ENTERPRISES offers a diskette containing a sampling of the quality of music of which REMUSIC is capable. Best of all, it requires no additional hardware. To receive this audition, send $2.00 to CODIS ENTERPRISES for the demo disk and a coupon entitling you to a two-dollar discount on REMUSIC 1.0.

Name_____

Address_____

City_____State_____Zip_____

Mail to:

CODIS ENTERPRISES
2301-C CENTRAL DR.
STE. 684
BEDFORD, TX 76021

Please send me ☐ the REMUSIC demo disk for $2.00*.
☐ REMUSIC 1.0 for $25.00*.

Enclosed is a check or money order for $_____.

*Texas residents add 7% sales tax.

determining the value of a two byte pointer.

```
1 MS=PEEK(M): LS=256*PEEK(M+1)
2 V=256*MS+LS
```

To convert a value into the two components it is necessary to do the following:

```
1 MS=INT(V/256):LS=V-256*MS
```

The INT command takes the integer part of the quotient and drops the decimal. The INT(9/2) is 4. Actually 9/2 is 4.5 but the decimal is discarded with the INT command.

There are some very important pointers that should be remembered. A few are as follows:

```
25,26 - Beginning of basic.
27,28 - Ending of basic.
136,137 - COCO 2 cursor.
150,151 - Printer baud rate.
```

The cursor can be moved by changing the values in 136 and 137. The first top left hand location is 1024 and the bottom right hand location is 1534. Let's write a program to display these pointers.

```
10 PRINT"CONT-2
20 PRINT"POINTER DEMO PROGRAM
30 PRINT"THE MOST SIGNIFICANT IS
    IN THE
40 PRINT"LOWER BYTE AND THE LEAS
   T SIG-
50 PRINT"NIFICANT IS THE UPPER B
   YTE. THE
60 PRINT"MOST SIGNIFICANT IS MUL
   TIPLIED
70 PRINT"BY 256 AND ADDED TO THE
    LEAST
80 PRINT"SIGNIFICANT TO FORM THE
    VALUE
90 PRINT"OF THE POINTER.
100 INPUT"PRESS ENTER TO CONTINU
   E";X:CLS
110 BE=256*PEEK(25)+PEEK(26)
120 EN=256*PEEK(27)+PEEK(28)
130 PR=256*PEEK(150)+PEEK(151)
140 PRINT"THIS PGM BEGINS AT"BE
150 PRINT"AND ENDS AT"EN
160 PRINT"PRINTER VALUE="PR
170 CU=256*PEEK(136)+PEEK(137)
180 PRINT"CURSOR VALUE="CU
190 '
```

These are just a few examples of using memory peeks and pokes. Locations from about 20 to 400 in lower memory are reserved mainly for vectors that direct various operations to different memory locations. Machine language subroutines can be placed into specified memory areas by using pokes. The values can be read from data statements and poked into memory. They can be poked directly into memory from a program similar to the following:

```
10 CLS:PRINT"CONT-3
20 PRINT"MEMORY PEEK AND POKE PG
   M"
30 INPUT"ENTER MEMORY";M
40 PRINT"1 PEEK MEMORY
50 PRINT"2 POKE VALUES INTO MEMO
   RY
60 PRINT"3 POKE CHARACTERS INTO
   MEMORY
70 INPUT"ENTER NUMBER ";X
80 ON X GO TO 100,150,190
90 GOTO 10
100 PRINT"THIS PEEKS MEMORY
110 A=PEEK(M):A$=CHR$(A)
120 PRINTM;A;A$
130 M=M+1:GOTO 110
140 '
150 PRINT"THIS POKES VALUES INTO
    MEMORY
160 PRINT"MEMORY="M;:INPUT"ENTER
    VALUE";V
170 POKE M,V:M=M+1:GOTO160
180 '
190 PRINT"THIS POKES CHARACTERS
   INTO MEMORY
200 A=PEEK(136):B=PEEK(137) 'HOL
   D CURSOR VALUES
210 PRINT@0,"MEMORY="M:POKE 136,
   A:POKE137,B
220 X$=INKEY$:IF X$="" THEN 220
230 X=ASC(X$);POKE M,X:PRINTX$;
240 M=M+1:GOTO 200
250 '
```

Next month we will continue with more on PEEKS and POKES.

1st 2nd 3rd 4th

By Tim Tillman

Welcome Back! This is article two in my introductory series on the language FORTH. FORTH is a high level language like PASCAL, BASIC, C and many others. We will be using Dynamic Electronic's PD-10 Color Computer FORTH. You can get your copy on tape or disk for only $6.00. Last month we saw several simple FORTH words, the stack, stack notation, single length numbers, and several math words that operate on them. In this month's article we'll wrap up single length numbers and go on to unsigned and double length numbers. We will also introduce the math words associated with these new types of numbers. Next, we'll look at a series of words that manipulate the order of numbers on the stack. Then, we'll look at two decision word constructions - DO ... LOOP and IF ... ELSE.

In last month's article we saw the words + - * and / , the four basic math operators used with single length numbers (-32768 to +32767). These words add, subtract, multiply, or divide the two top numbers on the stack and return the results to the top of the stack. There are four related words that can be used when speed is of the essence. These words are listed below:

WORD	STACK NOTATION	PRONUNCIATION
1+	(n1 -- n1+1)	one-plus
1-	(n1 -- n1-1)	one-minus
2*	(n1 -- n1*2)	two-times
2/	(n1 -- n1/2)	two-slash

Editor's note: In the charts for this article, continuations for a column are in the next line in the same column. Notice in the previous chart STACK NOTATION is the heading for the second column.

These words are excellent for incrementing counters or loops. It is faster to use these words than to write them out because they are part of FORTH's vocabulary defined in machine code. Their operation is simple, but here is an example to get you into FORTH gear.

```
: COUNTS 4 0 DO I 1+ DUP CR
. LOOP ;
```

Let's review the above word for the elements of a FORTH definition. The first word in the definition, : (COLON) tell's FORTH that a new definitiion follows. The next word COUNTS is the name of the new word. The next two numbers are elements of the DO ... LOOP construction; we'll discuss the particulars of the DO ... LOOP later in this article. The word I pushes the number of the loop onto the stack. 1+ adds one to the top number on the stack, and DUP duplicates the number on top of

the stack. CR prints a carriage return, and . prints the top number on the stack to the screen. LOOP ends the DO ... LOOP construction, and ; closes the new definition.

This word, COUNT5, prints the numbers 1 through 5 . To execute COUNT5, type in the word followed by an ‹ENTER›. Don't worry about the new words, we'll get to them later

There are two final single length words remaining to discuss. These words are */ (STAR-SLASH) and */MOD (STAR-SLASH-MOD). Here are the new words and their stack notations:

WORD	STACK NOTATION	PRONUNCIATION
*/	(n1 n2 n3 -- n4)	STAR-SLASH
*/MOD	(n1 n2 n3 - - n4 n5)	STAR-SLASH-MOD

The difference between */ and */MOD is the difference between / and /MOD. */MOD and /MOD both return the result and remainder, while */ and / return only the result. Another important fact about these two words is that they employ a double length number during the step (n1*n2). We will discuss double length numbers later in this article. For now, it is enough to say that we can now exceed our previous range limits of (-32768 to +32767). A good use for */ is to calculate percent. To see */ in action, let's define a word called % (PERCENT). The definition of % might look like this:

```
: % 100 */ . ;   (3Starting FORTH4, Leo Brodie 1981)
```

Now we can solve for percent using our new word, % . Try a few examples, such as:

```
1000 30 % . ‹ENTER›
2937 63 % . ‹ENTER›
10500 11 % . ‹ENTER›
```

*/ and */MOD are fairly easy to understand; but to get a firm grip on these two words, let's look at unsigned and double length numbers.

As I mentioned in last month's article, Forth recognizes various types of numbers. We have already seen the first type, single length numbers. As you know by now, the range of single length numbers is -32768 to +32767. But, I have not explained why. Well, here goes nothing.

Most people using computers know that on its most basic level the computer operates in binary code, ones and zeros. This is also the way that numbers are stored. We also know that each one or zero is refered to as a bit. With only one bit, the largest number that we can express is one and the smallest is zero. With two bits, and both bits on (or one), the largest number is three. With three bits, the largest number is seven. Refer to the following chart for numbers containing up to 16 bits:

3BIT #4	3POWER OF TWO4	3VALUE4
1	0	1
2	1	2
3	2	4
4	3	8
5	4	16
6	5	32
7	6	64
8	7	128
9	8	256
10	9	512
11	10	1024
12	11	1048
13	12	4096

14	13	8192
15	14	16384
16	15	+ 32768
		--------
	if all on -	65535

By adding the values of the first fifteen bits, we arrive at a figure of 32767. This is the largest number that we have known so far. The sixteenth bit is called the sign bit. If the sign bit is one, the number is negative. If the sign bit is zero, the number is positive.

Of course, a computer would not be very useful if its largest number were only 32767. This brings us to the next type of number, unsigned single length numbers. Unsigned implies that the number is positive. Therefore the sixteenth bit is not used to represent the sign of the number. If you add the values of the fifteenth and sixteenth bits, the result will be 65535. We now see, that the range of unsigned single length numbers is 0 - 65535.

Forth calls a group of sixteen bits (2 bytes) a cell. We can say, that single length and unsigned single length numbers both occupy one cell on the stack. Later we'll see words that manipulate the order of the cells of the stack. Of course, at times, we need larger numbers when we work with data.

The double length number has the largest range in the FORTH system. As its name implies, these numbers occupy two cells on the stack. If the double length number is signed it has a range of -1073741824. to 1073741823. If the number is unsigned its range is from 0. to 2147483648. This is certainly large enough for most CoCo applications. It is also important to note that in order for FORTH to recognize a double length number that it must be terminated with a decimal point. And, double length numbers must also be integers. There are no digits to the right of the decimal point. We will see ways of outputing decimal values in the next article. There are special words that manipulate two cells at a time, and we also have math words that operate unsigned and double length numbers. We'll see these words shortly.

But, before we go on, we'll need some additional words for sending numbers from the stack to your CoCo's Screen. Last month we used the word . to print single length numbers. This month, I will give you three more words to handle unsigned and double length numbers. The new words are listed as follows:

3WORD	PRONUNCIATION	STACK NOTATION4
U.	u-dot	(u --)
D.	d-dot	(d --)
UD.	u-d-dot	(ud --)

In the preceeding stack notations, we have three new abbreviations. The u stands for unsigned single length numbers. The d stands for double length numbers, and the ud stands for unsigned double length numbers. From the notations, we also see that these new words behave much like . They take the top number from the stack and print it on the CoCo's screen. They do not return a value to the stack. There are several other output words, but we'll learn more about them in future articles. Now let's see some more math words.

PD-10 Color Computer FORTH has a very limited set of words that operate on unsigned and double length numbers. The six words are listed as follows,

along with the output words associated with them:

3WORD	PRONUN-CIATION	OUT PUT	STACK NOTATION4
D+	d-plus	D. UD.	(d1 d2 -- d1+d2)
U*	u-star	D. UD.	(u1u2 -- ud1)
U/	u-slash	. U.	(d1 n1 -- n2 n3)
M*	m-star	D. UD.	(n1n2 -- d1)
M/	m-slash	D.	(d1 n1 -- n2 n3)
M/MOD	m-slash-mod	D. UD.	(d1 n1 -- n2 d2)

All of the words listed above behave like their single length counterparts. However, an explanation and definition involving D+ might be in order. The word D+ takes two double length words from the top of the stack and returns a double length result to the top of the stack. Since there is not a word defined to subtract double length numbers, we can define one using D+ and DMINUS :

```
: D- DMINUS D+ ;
```

The stack notation for DMINUS is (d1 -- -d1). To avoid confusion, let's call our new word, D- (d-dash). Here's what happens when D- is executed. DMINUS multiplies the top number on the stack by -1 and returns the negated number to the top of the stack. D+ then does its thing, returning a double length number to the top of the stack. Try using this example with D- :

```
200000.  100000.  D-  D.
```

100000. If you forgot a decimal or used . instead of D. or UD., you would have gotten an incorrect response. Now we can go on with stack manipulators.

Many times during the flow of a FORTH word written by a programmer, the order of numbers on the stack will not be what is required at the moment. This is why FORTH has words designed to change the order of numbers on the stack known as stack manipulators. There are two subgroups of manipulators. The first group is designed to operate with single length numbers. The second group is designed to operate on double length numbers or sets of two single length numbers. The first set is listed below:

3WORD	PRONUNCIATION	STACK NOTATION4
DUP	dupe	(n1 -- n1 n1)
-DUP	minus-dupe	(n1 -- n1 n1) if n1<>0
SWAP	swap	(n1 n2 -- n2 n1)
OVER	over	(n1 n2 -- n1 n2 n1)
ROT	rote	(n1 n2 n3 -- n2 n3 n1)
ROLL	roll	(n -- nth number to top of the stack. Extended ROT
PICK	pick	(n -- nth number copied to the top. Extended OVER
DROP	drop	(n --)

Looking at the stack notation for DUP, it is easy to see that it simply copies the top number on the stack. Dup was used in our example COUNT5 earlier in this article. PD-10 provides us with a word related to DUP called -DUP. -DUP will only copy if the top number is not equal to zero.

SWAP, OVER, and ROT are fairly self explainitory by their stack notations. So, we can go on to DROP, ROLL, and PICK. DROP takes the top number on the stack and disgards it with no further action. ROLL can be thought of as an extended ROT. ROLL requires an argument in the stack notation n. An example would help.:

```
5 ROLL ( n1 n2 n3 n4 n5 n6 -- n1 n3 n4 n5 n6 n2 )
```

In the above example, ROLL takes the fifth number from the top of the stack, and places it on top.

The phrase 3 ROLL is the same as ROT. PICK operates somewhat differently as can be seen from this example:

```
5 PICK ( n1 n2 n3 n4 n5 n6 -- n1 n2 n3 n4 n5 n6 n2 )
```

Here PICK copies the fifth number to the top of the stack. This is all of the single length stack mainpulators provided with PD-10. Now we can continue with double length manipulators.

Double length manipulators have two functions. First, they can manipulate double length numbers. Secondly, they can be used to manipulate pairs of single length numbers. PD-10 provides only one double length manipulator, 2DUP (two-dupe). I have defined four others for you: 2DROP, 2SWAP, 2OVER, and 2ROT. These are standard FORTH words explained in 3Starting FORTH4. You may try defining 2PICK and 2ROLL. I would be interested in seeing your defi-

nitions. Since their operations are easy to understand, I will simply present their stack notations below:

WORD	ON SINGLE LENGTH	ON DOUBLE LENGTH4
2DUP	(n1 n2 - - n1 n2 n1 n2)	(d1 - - d1 d1)
2DROP	(n1 n2 --)	(d1 --)
2SWAP	(n1 n2 n3 n4 - - n3 n4 n1 n2)	(d1 d2 - - d2 d1)
2OVER	(n1 n2 n3 n4 -- n1n2 n3n4 n1n2)	(d1 d2 - -d1 d2 d1)
2ROT	(d1n2 n3n4 n5n6--n3n4 n5n6 n1n2)	(d1 d2 d3 - - d2 d3 d1)

And here are the four definitions:

```
: 2DROP DROP DROP ;
: 2SWAP 4 ROLL 4 ROLL ;
: 2OVER 2SWAP 2DUP 6 ROLL 6
  ROLL 2SWAP ;
: 2ROT 6 ROLL 6 ROLL ;
```

If you can come up with different definitions for the above words I would be interested in hearing from you.

Now that we have seen the basics, let's look at two different control structures provided by PD-10 CoCo FORTH. These two structures are IF ... ENDIF and DO ... LOOP . Unlike BASIC, FORTH has no GOTO or GOSUB commands. They require line numbers, which FORTH does not have. GOTOs and GOSUBs can make a program difficult to read, update, and document. It may be tough going for a while, but soon you won't even miss them.

The first control structure, that we'll look at, is IF ... ENDIF . This phrase operates in much the same manner as the IF ... THEN Phrase described in 3Starting FORTH4. Here is an example of a word using IF ... ENDIF :

```
: ?TWENTY DUP 20 = IF ."
TWENTY " ELSE
          DUP 20 < IF ."
LESS " ELSE
                    ."
MORE "
          ENDIF ENDIF DROP
```

To execute ?TWENTY type any single length number followed by ?TWENTY and ‹ENTER›. Here's what happens upon execution. The word IF checks the preceeding condition, 20 = , and if the number on top of the stack makes the condition true, execution proceeds to the phrase following IF. If the condition is not true, then the execution proceeds to the word phrase following ELSE. The word ELSE is not required in all instances. However, each IF does require an ENDIF . If you nest IF ... ENDIF statements, it is important not to let your definition become too convoluted.

There are a few other condition words included in PD-10. These are:

WORD	STACK NOTATION	DESCRIPTION4
=	(n1 n2 -- r1)	True if n1 = n2
›	(n1 n2 -- r1)	True if n1 › n2
‹	(n1 n2 -- r1)	True if n1 ‹ n2
0=	(n1 -- r1)	True if n1 = 0
0‹	n1 -- r1)	True if n1 ‹ 0

In the previous stack notations

the r represents a logical result, if true (-1), if false (0).

The DO ... LOOP is the other type of control structure that we will discuss in this article. It is known as a definite loop, because we know exactly how many times it will repeat itself. Below, we can see the basic elements of the DO ... LOOP .

```
: A-LOOP 10 0 DO ." TESTING " CR LOOP ;
```

The first element after the new word's name is a single length number (+ or -) setting the upper limit of the loop. The next number sets the lower limit. The word DO tells the CoCo where the actual loop begins. Any following words are executed with each pass thruogh the loop. Finally, the word LOOP terminates the looping. Here is a noisy demonstration of a DO ... LOOP in action. First open CoCo's sound port by typing SPORT ‹ENTER›. Then try the following definition.

```
: NOISE 5000 0 DO RND BIP LOOP ;
```

Within the DO ... LOOP the word RND places a random unsigned single length number on the stack. The next word, BIP , takes that number and sends it to the sound port. If you had the volume turned up on your monitor, you would have heard 2-3 seconds of static or white noise.

Like IF ... ENDIF , DO ... LOOP may be nested as in this example:

```
: NOISE2 100 0 DO I DUP * BIP 50 0 DO I SQRT BIP LOOP LOOP ;
```

In this definition, we see two new words, I and SQRT. I copies the current number from the loop counter to the top of the stack, and SQRT takes the square root of the number at the top of the stack. As you might guess, the phrase, DUP * , will square the top number. One final thought, just as in BASIC, you may adjust the incrementation of the loop. Instead of LOOP you can use the phrase n LOOP+ . Where n is the new increment. In a future article I will discuss one final control structure, BEGIN ... REPEAT.

I had hoped to have a working FORTH program for this month's article, however, I had too much basic information to present. You should keep in mind that this article is only intended to be a brief over view of FORTH. For further information, you should get copies of text books such as 3Starting FORTH4 and 3Thinking FORTH4, both by Leo Brodie. These are two excellent books for beginners like ourselves.

Next month I will have a working program to discuss. We will also go over the editor supplied with PD-10. It is a very basic editor, and it has a few bugs in it still. If you would like to use your word processor until then, be sure that all of the text is in capitol letters.

Lastly, here are the answers to last month's problems and a few more for this month. If you have any questions, comments, or criticisms please don't hesitate to call or write. Don't call collect, and if you want a written reply, please SASE. Here's the info:

Tim Tillman
1479 31st Street NW
Winter Haven, Fl 33881
(813) 293-8619
5:00 to 10:00 EST

3ANSWERS4

1. 70 EMIT 79 EMIT 82 EMIT 84 EMIT 72 EMIT 33 EMIT ‹ENTER› FORTH!
2. : DISTANCE * ." IS THE DISTANCE TRAVELED " CR ;
3. : RATE / ." IS THE SPEED " CR ;
4. : P›D 2700 /MOD CR . ." DOLLARS AND " . . " PESOS REMAINING " ;
5. : 10SPC 10 SPACES ;
   : 8* 42 EMIT 42 EMIT 42 EMIT 42 EMIT 42 EMIT 42 EMIT 42 EMIT 42 EMIT ;
   : *BLOCK 10SPC 8* 10SPC 8* 10SPC 8* 10SPC 8*10SPC 8* 10SPC 8* 10SPC 8* 10SPC 8* ;

3PROBLEMS4

1. Using a DO ... LOOP simplify last month's problem 5

2. Example: Given (a b c --) solve b/(a+c)

   Answer:
   : PROB2 ROT + / ;
   a. (a b c d e --) ((a+c)*(d+e))/b
   b. (a b c x --) aR+bx+c
   c. (a b c d e --) (a+c+e)/(bR+e)

3. Print a triangle of 15 *'s

RENEWAL TIME?

IF 4/88 is beside your name on your address label then your subscription has expired.

FAST DIR

by

Andrew Bartels

Fast Dir is an easy to use Machine Language utility which will provide you with a quick directory of the last drive used when you press SHIFT-RIGHT- ARROW. The BASIC loader for it is listed here. It requires at least 32K.

The program takes up memory from $7000 to $74B5. Not all of it is code; most is buffer space. The code is entirely relocatable, so you may place it anywhere you like, as long as it is out of the way. DO NOT relocate it after it has been executed, however.

RUN the program and follow the prompts. When it is done executing, press SHIFT- RIGHT -ARROW. The screen will clear, and the double column directory of the disk in the last used drive will appear. If it is too long to fit on one screen, you may pause the DIR in progress by press any key. Any key will resume after a pause. After the entire DIR has been completed, press any key to return. When you do, the entire 32 column text screen is returned to it's state when you called the DIR, and the cursor is in the same spot as when you left it. Thus, you can be in the middle of editing a line, entering a command, making a backup, copying a file... whatever...and still you are able to get a DIR of the last drive used without messing up the operation you were in the middle of. The only requirement is that you have the normal flashing cursor at the momement you call Fast Dir.

The program was written in Assembly with the EDTASM editor /assembler.

FAST DIR LISTING

```
20 '‹‹FAST DIR››
30 'BY ANDREW B. BARTELS
35 'LICENSED TO DYNAMIC ELECTRON
   ICS INC.
40 CLS:PRINT"DIGITAL INNOVATIONS
    PRESENTS:":PRINT"‹‹FAST DIR›
   ›":PRINT"COPYRIGHT (C) 1987":
   PRINT"BY ANDREW B. BARTELS":P
   RINT:PRINT"ONE MOMENT...READI
   NG DATA..."
50 CLEAR200,&H6FFF:FORX=&H7000 T
   O &H71B3:READA:POKEX,A:NEXT:S
   OUND1,1:PRINT@160,STRING$(32,
   " ");:PRINT@160,"PRESS ‹ENTER
   › TO INSTALL...";:LINEINPUTA$
60 EXEC&H7000:CLS:PRINT"FAST DIR
    IS NOW INSTALLED.":PRINT"TO
   USE IT, PRESS SHIFT-RIGHT-
   ARROW...":PRINT:PRINT
70 DATA190,1,107,175,141,0,46,48
   ,141,0,4,191,1,107,57,15,112,
   52,1,13,111,38,27,50,98,173,1
   59,160,0,141,34,39,248,129,93
   ,38,9,141,16,52,119,141,77,53
   ,119,79,141,7,53,129,53,1,126
   ,0,0,52,4,198,96,231,159,0,13
   6,53,132,52,7,166
80 DATA141,1,105,167,159,0,136,1
   06,141,1,98,39,2,53,135,166,1
   41,1,89,198,128,231,141,1,84,
```

```
    139,16,129,15,38,2,134,143,16
    7,141,1,71,53,135,52,18,166,1
    28,173,159,160,2,77,38,247,53
    ,146,111,141,1,54,23,0,167,18
    9,169,40,16,190,192
 90 DATA6,134,2,167,164,204,17,3,
    167,34,231,35,52,4,48,141,1,3
    0,175,36,173,159,192,4,109,38
    ,38,100,166,132,39,91,76,39,1
    00,198,8,141,112,134,47,173,1
    59,160,2,198,3,141,102,166,14
    1,0,248,132,1,39,8,134,13,173
    ,159,160,2,32,11,198
100 DATA4,134,32,173,159,160,2,9
    0,38,249,108,141,0,221,48,136
    ,21,173,159,160,0,39,2,141,12
    3,51,141,0,209,51,201,1,0,239
    ,141,0,199,51,141,0,195,172,1
    96,38,173,48,141,0,189,108,22
    8,166,228,167,35,32,153,48,13
    6,32,32,219,48,141
110 DATA0,106,23,255,95,127,255,
    64,48,141,0,119,23,255,85,141
    ,60,141,35,53,130,166,128,173
    ,159,160,2,90,38,247,57,142,4
    ,0,49,141,1,136,236,129,237,1
    61,140,6,0,38,247,158,136,175
    ,141,0,114,57,142,4,0,49,141,
    1,113,236,161,237,129
120 DATA140,6,0,38,247,236,141,0
    ,93,221,136,57,173,159,160,0,
    39,250,57,52,18,48,141,0,59,2
    3,255,7,173,159,160,0,39,250,
    48,141,0,55,23,254,250,53,146
    ,13,13,73,78,80,85,84,47,79,8
    5,84,80,85,84,32,69,82,82,79,
    82,33,13,0,13,80,82
130 DATA69,83,83,32,65,78,89,32,
    75,69,89,46,46,46,0,60,80,65,
    85,83,69,68,62,0,8,8,8,8,8,8,
    8,8,0,0,0,143,128,0,0
```

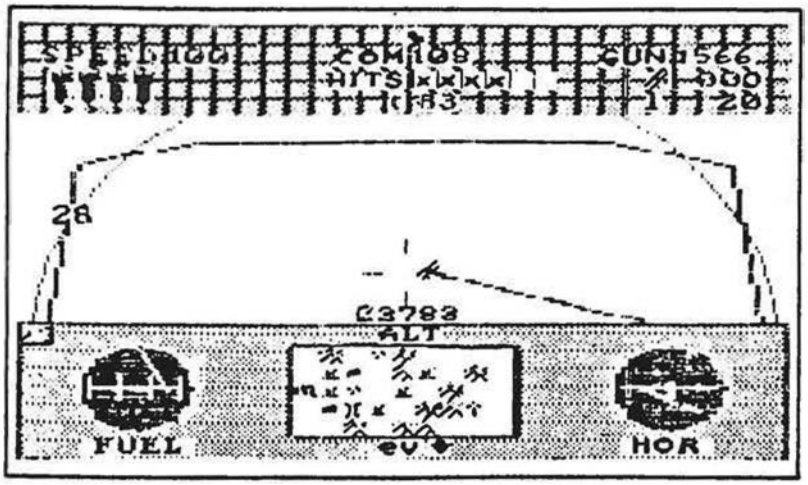
ASSEMBLY LISTING

```
00120 **************************
00130 *     ‹‹FAST  DIR››      *
00140 **************************
00150 *   BY ANDREW BARTELS    *
00160 **************************
00170 POLCATEQU$A000
00180 CHROUTEQU$A002
00190 CLSEQU$A928
00200 ORG$7000
00210 BOOTLDX$16BGET INPUT ROUTI
   NE ADDRESS
00220 STXIADDR,PCRSAVE IT FOR LA
   TER
00230 LEAXIN,PCRPOINT TO MY ROUT
   INE
00240 STX$16BINSTALL FAST DIR NO
   W
00250 RTSRETURN TO BASIC
00260 INCLR$70CLEAR FLAG
00270 PSHSCCSAVE CC
00280 TST$6FIS IT KEYBOARD INPUT
   ?
00290 BNEIDONE2NOPE...THEN SKIP
   THERE
00300 LEAS2,SLIFT STACK
00310 GETKEYJSR[POLCAT]GET A KEY
   PRESS
00320 BSRFLASHGO FLASH CURSOR ON
   CE
00330 BEQGETKEYIF NO KEY, KEEP C
   HECKING
00340 CMPA#93IF SO, WAS IT SHFT-
   RT-ARROW?
00350 BNEIDONENO...RETURN KEY AS
    NORMAL
00360 BSRERASEGO ERASE CURSOR
00370 PSHSD,CC,X,Y,USAVE EVERYTH
   ING
00380 BSRDDIRGO DO DIRECTORY
00390 PULSD,CC,X,Y,UGET EVERYTHI
   NG
00400 CLRADON'T PASS ON THE ARRO
   W
00410 IDONEBSRERASEGO ERASE CURS
   OR
00420 PULSCC,PCRETURN
00430 IDONE2PULSCCGET CC BACK
00440 FCB$7EJMPCODE
00450 IADDRFDB0ADDR OF OLD RTN
00460 ERASEPSHSBSAVE B
00470 LDB#96GET BLANK
00480 STB[$88]ERASE CURSOR
00490 PULSB,PCRETURN
00500 FLASHPSHSD,CC
00510 LDACURS,PCRGET CURSOR VALU
   E
00520 STA[$88] SHOW IT
00530 DECCOUNT,PCRCOUNT DOWN
00540 BEQCHANGEIF TIME TO CHANGE
   , THEN DO IT
00550 PULSD,CC,PCIF NOT, THEN RE
   TURN
00560 CHANGELDACURS,PCRGET CURSO
   R VAL
00570 LDB#128GET CURSOR COUNTER
00580 STBCOUNT,PCRRESET IT
00590 ADDA#16GO TO NEXT CURS
00600 CMPA#15DID WE ROLL OVER?
00610 BNEGOONNO...THEN GO ON
```

```
00620 LDA#143YES...THEN RESET OV
   ER AGAIN
00630 GOONSTACURS,PCRSAVE NEW CU
   RSOR
00640 PULSD,CC,PCRETURN
00650 PRINTPSHSA,XSAVE REGISTERS
00660 PRINT1LDA,X+GET A CHARACTE
   R
00670 JSR[CHROUT]PRINT IT
00680 TSTAWAS IT A ZERO?
00690 BNEPRINT1NO...THEN PRINT M
   ORE
00700 PULSA,X,PCRETURN
00710 DDIRCLRSIDE,PCRSET SIDE TO
   GGLE TO LEFT
00720 LBSRCOPYMOVE TEXT SCREEN T
   O BUFFER
00730 JSRCLSTO CLEAR SCREEN
00740 LDY$C006POINT TO DSKCON PA
   RAMETERS
00750 LDA#2A=2
00760 STA,YSET TO READ
00770 LDD#$1103A=17,B=3
00780 STA2,YSET TRK = 17
00790 STB3,YSET SECT =3
00800 PSHSBSAVE SECTOR ON STACK
00810 LEAXBUFF,PCRPOINT TO BUFFE
   R
00820 STX4,YSET BUFFER TO DSKCON
00830 DDIR1JSR[$C004]CALL DSKCON
00840 TST6,YWAS THERE AN ERROR?
00850 BNEERRORYES, THEN REPORT I
   T
00860 DDIR2LDA,XCHECK FIRST BYTE
00870 BEQDDIR7IF 0, FILE WAS KIL
   LED...SKIP IT
00880 INCAIF IT WAS 255, IT IS N
   OW 0
00890 BEQDONEIF IT WAS 255, THEN
    DIR IS DONE
00900 LDB#8PRINT FILENAME
00910 BSRNAM
00920 LDA#47PRINT A "/"
00930 JSR[CHROUT]
00940 LDB#3PRINT 3 CHARS
00950 BSRNAMPRINT EXTENSION
00960 LDASIDE,PCRGET SIDE TOGGLE
00970 ANDA#1IS IT ODD?
00980 BEQDDIR3NO, THEN PRINT SPA
   CES
00990 LDA#13YES, THEN PRINT <CR>
01000 JSR[CHROUT]
01010 BRADDIR5UNTIE PROGRAM FLOW
01020 DDIR3LDB#4DO 4 SPACES
01030 LDA#32
01040 DDIR4JSR[CHROUT]PRINT A SP
   ACE
01050 DECBDONE?
01060 BNEDDIR4NO...DO MORE
01070 DDIR5INCSIDE,PCRNEXT SIDE
01080 LEAX21,XPOINT TO XNEXT ENT
   RY
01090 JSR[POLCAT]WAS THERE A PAU
   SE KEY?
01100 BEQDDIR6NO...THEN KEEP ON
01110 BSRPAUSEYES..THEN PAUSE FO
   R USER
01120 DDIR6LEAUBUFF,PCRPOINT TO
   BUFFER START
01130 LEAU256,UPOINT TO BUFFER E
   ND
01140 STUHOLD1,PCRSAVE
01150 LEAUHOLD1,PCRPOINT TO POIN
   TER
01160 CMPX,UDONE WITH BUFFER?
01170 BNEDDIR2NO...DO MORE IN TH
   IS ONE
01180 LEAXBUFF,PCRYES..RESET BUF
   FER POINTER
01190 INC,SNEXT SECT
01200 LDA,SGET NEXT ONE
01210 STA3,YLET DSKCON KNOW
01220 BRADDIR1GO CALL DSKCON & D
   O IT OVER
01230 DDIR7LEAX32,XPOINT TO NEXT
    ENTRY
01240 BRADDIR6CONTINUE
01250 ERRORLEAXMSG,PCRPOINT TO M
   ESSAGE
01260 LBSR PRINTPRINT IT ON SCRE
   EN
01270 DONECLR$FF40STOP DRIVE
01280 LEAXMSG1,PCRPOINT TO MESSA
   GE#1
01290 LBSRPRINTGO PRINT IT
01300 BSRWAITWAIT FOR A KEY PRES
   S
01310 BSRRESTORRESTORE SCREEN TH
   E WAY IT WAS
01320 PULSA,PCRETURN FOM DIR
01330 NAMLDA,X+GET A CHAR
01340 JSR[CHROUT]PRINT IT
01350 DECBDONE?
01360 BNENAMNO...PRINT MORE
01370 RTSRETURN
01380 COPYLDX#1024POINT TO TEXT
   SCREEN
01390 LEAYBUFF2,PCRPOINT TO BUFF
   ER
01400 COPY1LDD,X++GET TWO BYTES
01410 STD,Y++SAVE IN BUFFER
01420 CMPX#1536DONE WITH SCREEN?
01430 BNECOPY1NO...COPY MORE
01440 LDX$88GET CURSOR POSITION
```

```
01450 STXCADDR,PCRSAVE CURSOR AD
   DRESS
01460 RTSRETURN
01470 RESTORLDX#1024POINT TO SCR
   EEN
01480 LEAYBUFF2,PCRPOINT TO BUFF
   ER
01490 REST1LDD,Y++GET TWO BYTES
01500 STD,X++RESTORE SCREEN
01510 CMPX#1536DONE WITH SCREEN?
01520 BNEREST1NO...KEEP ON
01530 LDDCADDR,PCRGETOLD CURSOR
   ADDRESS
01540 STD$88RESTORE CURSOR
01550 RTSRETURN
01560 WAITJSR[POLCAT]GET A KEY
01570 BEQWAITNONE...KEEP WAITING
01580 RTSRETURN
01590 PAUSEPSHSA,X
01600 LEAXMSG2,PCRPOINT TO MESSA
   GE#2
01610 LBSRPRINTPRINT IT
01620 PAUSE1JSR[POLCAT]GET A KEY
01630 BEQPAUSE1IF NONE...THEN WA
   IT MORE
01640 LEAXMSG3,PCRPOINT TO MESSA
   GE#3
01650 LBSRPRINTPRINT IT
01660 PULSA,X,PCRETURN
01670 MSGFDB$D0D2 CHR$(13)'S
01680 FCC*INPUT/OUTPUT ERROR!*
01690 FDB$D00
01700 MSG1FCB$D
01710 FCC/PRESS ANY KEY.../
01720 FCB0
01730 MSG2FCC/‹PAUSED›/
01740 FCB0
01750 MSG3FDB$808
01760 FDB$808
01770 FDB$808
01780 FDB$808
01790 FCB0
01800 CADDRFDB0
01810 CURSFCB143
01820 COUNTFCB128
01830 SIDEFCB0
01840 HOLD1FDB0
01850 BUFFRMB256
01860 BUFF2RMB512
01870 ENDBOOT
```

MORE FROM ARK ROYAL!



A C E S is a high resolution, completely machine language game of aerial warfare in WWI. Player flies on many missions to bomb enemy targets including airfields, enemy headquarters, anti-aircraft batteries, bridges and factories, but not player's own air base. He must dodge mountains and dogfight with the enemy's best, including, if unlucky, members of the dreaded Flying Circus. After he shoots down five planes he becomes an ACE and receives special consideration; but the game is far from finished. **A C E S** averages about 82 targets and over 100 enemy aircraft per game.

A C E S plays in real time and displays flight simulated dash and controls. Operates from the keyboard. Included in the display is a high resolution mini-screen featuring terrain, targets, and player's relative ground position. There are 8 zones in each map which changes as player flies over it. Game Save. (It could take days to win!) In addition, **NEWMAP** is included to allow for the creation of a zillion new maps. **A C E S** was created in part with AGS, developed by Ken Schunk. For all CoCo's.

WAR AT SEA: Wooden Ships simulate ship to ship battles during the 18th Century. Player controls a number of sailing ships from different nations and must pit his seamanship against the computer or another player.

RED ALERT: a starship combat simulator. Object of the game is to defeat the computer controlled enemy vessel by using your ship's capacities, strategic maneuvers, and your own smarts.

NEW

A C E S: WWI Aerial Warfare (CC64K D HR ML) $29
RED ALERT: Star Ship Warfare (CC64K D HR MLS J) $27
WAR AT SEA: Wooden Ships (CC64K D HR MLS J) $25

Pro Football: Strategy Gridiron game (CC3 128K HR B) $20
Okinawa: The Big Invasion (Screen Dump inc) (CC64K D HR ML) $27
Blitzkrieg West: A Bigger Bulge (CC64K D HR ML) $27
Bataan: Historial & Hypothetical games in one (CC64K D HR ML) $29
Desert Fox: Rommel (CC64K D HR MLS) $27
Task Force: Modern Naval War in the Med (CC64K D HR MLS J) $27
D DAY: The 6th of June (CC64K HR ML) $25
Battle Hymn: Battle of Gettysburg (CC64K D HR ML) $25
Company Commander: Squad Level Wargame (CC32K SG MLS) $25
(House to House Module included in Company Commander)
Additional Modules for Company Company 3.0
River Crossing $17
Gemini $17
Cauldron $17
Beach Head $17
Fire One! Submarine Simulation (CC3 D HR B) $25
Fire & Steel: Waterloo Campaign (CC64K D HR MLS) $22

Luftflotte: Battle of Britain (CC32K SG MLS) $25
Stalingrad: The turning point. (CC64K HR ML) $25
Final Frontier: War in Space (CC32K D HR MLS) $25
Barbarossa: The War in Russia (CC64K HR ML) $22
RedStar: Nato vs Warsaw Pact (CC32K D HR ML) $22
DarkHorse: Redstar Sequel (CC64K D HR ML) $22
Midway: The Turning Point in the Pacific (CC32K HR MLS) $20
Escape From Denna: Dungeons! (CC32K SG MLS) $15
Tunis: War in the Desert (CC32K SG B) $15
Battle of the Bulge 1 or 2 player (CC32K SG B) $15
Phalanx: Alexander the Great (CC32K HR ML) $15
Rubicon II: Invasion game (CC32K SG B) $10
Guadalcanal: America Strikes Back (CC32K SG MLS) $10
Waterloo: Napoleon (CC32K SG MLS) $10
Bomber Command: Strategic Bombing Mission (CC32K SG MLS) $10
Kamikaze: Naval War in the Pacific (CC32K HR B) $10
Starblazer: Strategy Star Trek (CC32K SG MLS) $10
Mission Empire: Build an Empire in Space (CC32K SG B) $10
Galactic Taipan: Economics in Space (CC32K SG B) $10

Keyboard General: Bi-monthly newsletter yearly sub $15

Barbarossa, Luftflotte, Battle Hymn (256K) available Tandy 1000

New for the Tandy 1000
Gray Storm Rising: War in the North Atlantic $25

Codes: CC — Color Computer, all versions CC3 — CoCo 3 only
D — Disk only (no D means program available tape or disk)
HR — High Resolution SG — Semigraphics ML — Machine Language
MLS — Machine Language Subroutines B — Basic J — Joystick

Write for free catalog!

Prices include shipping to USA and Canada. Others add $3.00. COD's available in USA only, add $3.50. Personal Checks accepted with no delays in USA. Others must send M.O. or Bank Draft in U.S. funds. Programs shipped within 24 hours except on weekends. Sorry, no bankcards. Color Computer and Tandy 1000, TM Tandy Corp.

Florida residents add 6% sales tax.

Canadians may order direct from: **M & M Software**, Post Office Box 1945, Slave Lake, Alberta T06 2A0. Write M & M Software for information.

ARK ROYAL GAMES
P.O. Box 14806
Jacksonville, FL 32238
(904) 786-8603

ARK ROYAL GAMES

ham radio & computers
by
bill chapple w4gqc

Each month I present information on computers that can be applied to ham radio operation. Last month I presented a public domain WEFAX program that uses the cassette interface. I have been looking at using this interface for other applications such as packet. The advantage of using the cassette interface is that there are no circuits to build. The only interface being a plug adapter to allow the receiver's audio to be placed on the line that normally goes to the cassette output.

Last November I presented a teletype program that uses this interface. I have done much experimentation with this and have concluded that the applications are rather limited. The reason for my conclusion is that this is a one bit port. The only thing that can be measured is frequency. This is fine for RTTY and WEFAX, but for packet and CW or Morse Code amplitude is also a requirement.

The problem I have with packet using this port is determining when the packet starts. I need something that will give me a start signal as the packet begins. Also for Morse code, it is necessary to look at amplitude in order to eliminate noise that will cause false data to be printed. I have a solution for these problems that is not expensive. Last year we did editorials on using the joystick ports for various applications. A joystick consists of two potentiometers that divide down 5 volts depending upon the position of the lever. An analog to digital converter inside the computer converts these voltages into numbers from 0 to 63. I have already built an adapter to use one of the joystick ports and will report more on it next month. With amplitude I can do many types of audio processing including voice. Wouldn't it be nice to remove a carrier while trying to copy single side band (ssb)?

This month I have a tuning meter program that consists of 3 sections. In the first position it displays the frequencies from 200 to 3000 hertz. A bar moves across the screen on the top line and positions itself at the closest frequency. The bar moves back and forth as the frequencies change. This can be used for all modes of operation including voice.

The second position is for tuning high frequency (hf) packet signals. This looks at a narrower spectrum and gives 4 positions for each 100 hertz.

The third position is for hf rtty using narrow frequency shift.

The program is easy to use. A machine language subroutine is used to measure the frequency and erase the first display line. The machine language subroutine is carried with the program as data. This program works on all versions of the color computer with extended basic.

TUNING METER PROGRAM

```
5 FOR J=510 TO 572:READ A:POKEJ,
  A:NEXT 'READ IN MACHINE LANGU
  AGE SUBROUTINE
6 'THE MACHINE LANGUAGE SUBROU
  TINE TIMES THE AUDIO SIGNAL A
  ND PUTS THE RESULT IN MEMORY
  LOCATION 501
7 'IT ALSO CHECKS FOR A ZERO AND
   PLACES A 1 IN 501 IF THE VAL
  UE IS 0
8 'IT ALSO ERASES THE FIRST LINE
   ON THE SCREEN
10 CLS:PRINT"AUDIO TUNING METER
20 PRINT"BY BILL CHAPPLE W4GQC
30 PRINT"cOPYRIGHT (c) 1988
40 PRINT"dYNAMIC eLECTRONIC iNC.
50 PRINT"1 DISPLAY TOTAL AUDIO S
  PECTRUM
60 PRINT"2 PACKETT TUNING METER
70 PRINT"3 HF RTTY TUNING METER
80 PRINT"ENTER NUMBER
85 X$=INKEY$:IF X$=""THEN 85
87 X=VAL(X$)
90 ON X GOTO 500,600,705
500 CLS:PRINT
505 'THIS IS FOR ALL AUDIO FREQU
  ENCIES FROM 200 TO 3000
510 PRINT
515 PRINT" 2 4 6 8 1 1 1 1 1 2 2
   2 2 2 3
520 PRINT" 0 0 0 0 0 2 4 6 8 0 2
   4 6 8 0
525 PRINT" 0 0 0 0 0 0 0 0 0 0 0
   0 0 0 0
530 PRINT"          0 0 0 0 0 0 0
   ● 0 0 0
532 PRINT
535 PRINT"     AUDIO FREQUENCIES
537 PRINT"BAR SHOWS LOCATION OF
  FREQUENCY
540 EXEC 510:D=PEEK(501)
550 F=INT(740/D+.5)
555 IF F>31 THEN POKE1055,62:GOT
  O540
560 POKE 1024+F,197
565 X$=INKEY$:IF X$<>"" THEN RUN
570 GOTO540
575 END
600 '
605 'PACKET DISPLAY
610 CLS:PRINT
615 PRINT"          *        *
620 PRINT" 1   1   1   1   1   1
      2   2
625 PRINT" 4   5   6   7   8   9
      0   1
630 PRINT" 0   0   0   0   0   0
      0   0
635 PRINT" 0   0   0   0   0   0
      0   0
640 PRINT:PRINT"    HF PACKET TU
  NING
645 PRINT"    MATCH BARS WITH ST
  ARS
650 EXEC510
655 D=PEEK(501)':IFD=0THEND=1
660 'F=74000/D
665 X=INT(2960/D-54.5)
670 IF X>31 THENPOKE1055,62:GOTO
  650
675 IF X<=1 THEN POKE 1024, 60
677 X$=INKEY$:IF X$<>""THENRUN
680 POKE1024+X,197
685 GOTO650
700 '
705 'HF RTTY DISPLAY
710 CLS:PRINT
715 PRINT"          *        *
720 PRINT" 1   2   2   2   2   2
      2
725 PRINT" 9   0   1   2   3   4
      5
730 PRINT" 0   0   0   0   0   0
      0
735 PRINT" 0   0   0   0   0   0
      0
740 PRINT"    HF RTTY TUNING
745 EXEC510
750 D=PEEK(501)
755 F=74000/D
760 X=INT(2960/D-74.5)
765 IF X>31 THENPOKE1055,62:GOTO
  745
770 IF X<=1 THEN POKE 1024, 60:G
  OTO745
775 POKE1024+X,197
777 X$=INKEY$:IF X$<>""THEN RUN
780 GOTO745
785 '
800 DATA 26,80,182,255,32,132,1,
  39,249,182,255,32,132,1,38,24
  9
810 DATA 95,182,255,32,92,132,1,
  39,248,32,2,18,18,182,255,32,
  92,132,1,38,248,247,1,245,142
  ,4
820 DATA 0,134,96,167,128,140,4,
  32,45,249,182,1,245,77,38,1,7
  6
830 DATA 183,1,245,57
```

ML SUBROUTINE

The assembly listing of the machine language subroutine follows. This was assembled using our DISASM program. All numbers are in decimal.

```
510 ORCCI     80
512 LDA  E   65312
515 ANDA I   1
517 BEQ      512
519 LDA  E   65312
522 ANDA I   1
524 BNE      519
526 CLRB
527 LDA  E   65312
530 INCB
531 ANDA I   1
533 BEQ      527
535 BRA      539
537 NOP
538 NOP
539 LDA  E   65312
542 INCB
543 ANDA I   1
545 BNE      539
547 STB  E   501
550 LDX  I   1024
553 LDA  I   96
555 STA X DIR  R+
557 CMPX I   1056
560 BLT      555
562 LDA  E   501
565 TSTA
566 BNE      569
568 INCA
569 STA  E   501
572 RTS
```

OPERATING HINT

Programs can be stacked by changing vectors in locations 25-28. Do a memory peek and write down the values. Let V=PEEK(27) + 2. Poke this value into 25. POKE 256*V, Ø: NEW. The new program can now be loaded. This occupies memory above your first program. You can return to the first program by restoring the original values in 25-28.

DYNAMIC ELECTRONICS INC.

PUBLIC DOMAIN SOFTWARE

This large collection of programs will allow you to quickly expand your library. All programs are on disk and programs with a * can be supplied on tape. Some programs require a joystick. Instructions are included in some collections as DAT or TXT files

* PD-1 GAMES

MENU	BAS	0	B	1
BEAST	BAS	0	B	1
BEAST	DAT	1	A	1
BOBO	BAS	0	B	3
GUNNER	BAS	0	B	2
HOW	BAS	0	B	3
LANDER	BAS	0	B	3
LIFE	BAS	0	B	3
MAX	BAS	0	B	3
POKER	BAS	0	B	2
BIORITHM	BAS	0	B	3
BLACKBOX	BAS	0	B	2
BLOCKADE	BAS	0	B	1
BUSJUMP	BAS	0	B	1
CHUTE	BAS	0	B	2
GO	BAS	0	B	3
HANGMAN	BAS	0	B	2
OTHELLO	BAS	0	B	2
TARTUS	BAS	0	B	1
TARTUS2	BAS	0	B	1

* PD-2 GAMES

MENU	BAS	0	B	1
RUBIC	BAS	0	B	5
FRACTAL	BAS	0	B	1
KALSCOPE	BAS	0	B	2
TARTUS	BAS	0	B	1
TARTUS2	BAS	0	B	1
WORLD3B	BAS	0	B	4
LIFE	BAS	0	B	2
ADVENT	BAS	0	B	4
ADVENT	DOC	1	A	2
HURKLE	BAS	0	B	2
REVERSE	BAS	0	B	2
GUESSFR	BAS	0	B	2
SCRAMBLE	BAS	0	B	3
PIZZA	BAS	0	B	2
CINQUAIN	BAS	0	B	2

* PD-3 GAMES

MENU	BAS	0	B	1
AANDAN	BAS	0	B	2
STARTREK	BAS	0	B	9
TREKINST	BAS	0	B	3
SEQUENCE	BAS	0	B	2
ALPHABET	BAS	0	B	3
GEOGRAPH	BAS	0	B	4
FLASH	BAS	0	B	4
BAGELS	BAS	0	B	3
OREGON	BAS	0	B	9
MULTIPLY	BAS	0	B	2

* PD-4 ML GAMES

MENU	BAS	0	B	1
PONG	BIN	2	B	1
SQUASH	BIN	2	B	2
BLOCKADE	BIN	2	B	2
GERM	BIN	2	B	1
WIGWORM	BIN	2	B	2
GRID	BIN	2	B	2
ZEROG	BIN	2	B	2
3DTICTAC	BIN	2	B	7
HOPBOP	BIN	2	B	5
ICEWAR	BAS	0	B	6
CIVILWAR	BAS	0	B	4
TICTACTO	BIN	2	B	7

* PD-5 GAMES

MENU	BAS	0	B	1
CAVE	BAS	0	B	4
WARGAME	BAS	0	B	2
WARGAME	BIN	2	B	1
WARGAME2	BAS	0	B	5
WARROOM	BIN	2	B	3
NORAD	BAS	0	B	3
ANDREA	BAS	0	B	5
CURSE	BAS	0	B	4
GARGOYLE	BAS	0	B	6
KINGTUT	BAS	0	B	7
TAIPAN	BAS	0	B	6

DSK-G

SPELL & FIX- FIND SPELLING ERRORS IN TXT DISK FILES

MENU	BAS	0	B	1
MANUAL	TXT	1	A	12
SPELLFX2	BAS	0	B	1
SPELLFX2	BIN	2	B	6
SPELLFIX	BAS	0	B	1
DICT	TXT	1	A	33
COREDICT	TXT	1	A	1
SAMPLE	TXT	1	A	1
BUILD	BAS	0	B	1
LIST	BAS	0	B	1
RESET	BAS	0	B	1
APPEND	BAS	0	B	1
ADDWORDS	BIN	2	B	3

PD-7 DISK UTILITIES

MENU	BAS	0	B	1
BASIC64	BIN	2	B	1
BSEARCH	BIN	2	B	1
DISKCOMP	BIN	2	B	1
DISKTEST	BIN	2	B	3
DISKWASH	BAS	0	B	1
DOS64K	BAS	0	B	2
DSDBOOT	BIN	2	B	1
LIST	BIN	2	B	2
PRINT	BIN	2	B	3
PRINTDIR	BAS	0	B	1
RECOVER	BIN	2	B	1
ROMBACK	BAS	0	B	1
ROMFIX	BIN	2	B	1

PD-8 DISK UTILITIES

SCRN51	BAS	0	B	1
SCRN51	BIN	2	B	1
SCRNDEMO	BAS	0	B	2
SDC	BIN	2	B	1
SQUEEZE	BIN	2	B	1
SSDBOOT	BIN	2	B	1
TAPE2DSK	BAS	0	B	1
TIMER	BIN	2	B	2
UNLOCK	BIN	2	B	1
BACKUP	BIN	2	B	1
BACKUP1	BIN	2	B	1
MORE	BIN	2	B	3
SPEAK	BIN	2	B	3
PCLEARFX	BIN	2	B	1
MULTBACK	BIN	2	B	1
MULTBACK	DOC	1	A	1

PD-9

TERMINAL PROGRAMS

MENU	BAS	0	B	1
TELETERM	BIN	2	B	3
TELETERM	CAS	2	B	3
TTHELP	DAT	1	A	4
MTERM	BIN	2	B	6
MTERM	VIP	1	A	19
MTCONFIG	BAS	0	B	3
MTERM*	BIN	2	B	6
DATATRDE	BIN	2	B	3
KERMIT	BAS	1	A	1
KERMIT	BIN	2	B	2
HAYESAE	BIN	2	B	4
HAYESAE	DOC	1	A	6

PD-10

COLOR COMP. FORTH

MENU	BAS	0	B	1
FORTHMAN	UL1	2	B	7
FORTHMAN	UL2	2	B	7
FORTHMAN	UL3	2	B	1
FORTH	BIN	2	B	3
EDIT	DAT	1	A	3
FRTHDOC2	TXT	1	A	7
FRTHDOC3	TXT	1	A	1
FRTHDOC4	TXT	1	A	7
32KFORTH	BIN	2	B	4
NEWFORTH	BIN	2	B	3
WE	BAS	0	B	1

PD-11 MCPAINT

A COMPLETE GRAPHICS DEVELOPMENT PROGRAM WITH INSTRUCTIONS

RUN-ME	BAS	0	B	1
MCPAINT	BIN	2	B	11
ICONS	SYS	2	B	3
MCDOC	DOC	1	A	11
PRINTDOC	BAS	1	A	1
GLASDEMO	BIN	2	B	6
STARS	BIN	2	B	2
1940S	SET	2	B	1
BLOON	SET	2	B	1
BOLD	SET	2	B	1
FANCY	SET	2	B	1
GREEK	SET	2	B	1
GREEKU	SET	2	B	1
HEBREW	SET	2	B	1
OLDENG	SET	2	B	1
TYPING	SET	2	B	1
EPSON	DRV	2	B	1
EPSON2	DRV	2	B	1
ANIMATE	BAS	0	B	1
ANIMAT	BIN	2	B	1
BANNER	BAS	0	B	2
MCUTIL	BIN	2	B	1

* PD-12

PMODE 4 PICTURES

CHURCH, ROSES, HOUSE
RUN "PIXFILES"
JOYSTICK IS REQUIRED

XIXCMP	BAS	0	A	3
OUTPOST	BAS	0	A	3
OUTPOST	BIN	2	B	3
SFIELD	BAS	0	A	2
SFIELD	BIN	2	B	3
PIXFILES	BAS	0	B	3
TRUCK	BIN	2	B	3
MODEM	BIN	2	B	3
HORSE	BIN	2	B	3
MISSION	BIN	2	B	3
CLOISTER	BIN	2	B	3
RAIN	BIN	2	B	3
EAGLE	BIN	2	B	3
ROSES	BIN	2	B	3
CHURCH	BIN	2	B	3
GARDEN	BIN	2	B	3
PRES	BIN	2	B	3
LONI4	BAS	0	A	3

PD-13

GRAPHICON PICTURE DISK-1. REQUIRES PIXFILES/BAS FROM PD-12 & JOYSTICK

PICTURES GCM 1 B 68

PD-14

GRAPHICON PICTURE DISK-2. REQUIRES PIXFILES/BAS FROM PD-12 & JOYSTICK

PICTURES GCM 1 B 66

PD-15

GRAPHICON PICTURE DISK-3 REQUIRES PIXFILES/BAS FROM PD-12 & JOYSTICK

PICTURES GCM 1 B 68

PD-16

GRAPHICON PICTURE DISK-4 REQUIRES PIXFILES/BAS FROM PD-12 & JOYSTICK

PICTURES GCM 1 B 68

PD-17 DISK UTILITIES

64KBHW	BAS	0	A	1
AUTOSTRT	BAS	0	B	1
BAKDIR	BAS	0	A	3
BIN>BAS	BAS	0	A	1
CASSLABL	BAS	0	B	1
CURSOR	BAS	0	B	1
CUSTOM	BAS	0	B	3
CUSTOMIZ	BAS	0	B	1
DIR	BIN	2	B	1
DIR32	BAS	0	A	2
DIR32C	DOC	1	A	3
DIRLISTR	BAK	0	B	1
DIRLISTR	BAS	0	B	1

PD-18 TAPE TO DISK DISK UTILITIES

DIRSORT	BAS	0	A	1
DISK-DIR	BAS	0	A	1
DISKLABL	BAS	0	A	1
LOADSOLU	BAS	0	B	1
MENU	BAS	0	B	1
PDIR	BAS	0	A	1
SORT	BAS	0	B	1
SORTPRT	BAS	0	B	1
SORTSAVE	BAS	0	A	1
SOULTION	BIN	2	B	1
SUPERBAC	BIN	2	B	1
T2D	BIN	2	B	2
TIMER	BAS	0	B	1
TPTODSK	BIN	2	B	1

* PD-19 GAMES

3DMAZE	BAS	0	A	2
BOXES	BAS	0	B	1
CLOSE EN	BAS	0	B	2
CRITICAL	BAS	0	B	1
GAMMON	BAS	0	B	3
GOLDMINE	BAS	0	A	3
HOCKEY	BAS	0	A	1
HOGJOWL	BAS	0	A	8
HORSERAC	BAS	0	A	3
JUMPING	BAS	0	B	1
KALIDESC	BAS	0	B	1
MASTMIND	BAS	0	B	1
MEMORY	BAS	0	B	1
MOONBASE	BAS	0	B	2
NAMES	BAS	0	B	4
OTHELLO	BAS	0	B	4

* PD-20 GAMES

PEG	BAS	0	B	3
RABBIT	BAS	0	B	1
SAFE	BAS	0	B	2
SAUACER	BAS	0	B	1
SHOOTEM	BAS	0	B	2
SIMMON	BAS	0	A	1
SLITHER	BAS	0	A	2
SPACE WA	BAS	0	B	4
STAR TRE	BAS	0	B	1
SUBCHASE	BAS	0	B	2
SUBDESTR	BAS	0	B	2
SUNDANCE	BAS	0	B	2
TANKS	BAS	0	B	2
TOWER	BAS	0	B	2
UNDROVER	BAS	0	B	1

PD-21 MUSIC

PLAY MUSIC THROUGH YOUR TV OR MONITOR. COMPOSE & EDIT MUSIC

ORCH	BIN	2	B	8
ORCH	DOC	1	A	3
OCNVRT	BIN	2	B	2
GHOSBUST	MUS	4	M	3
STELMO	MUS	4	M	2
MASH	MUS	4	M	2
BOND1	MUS	4	M	2
2001	MUS	4	M	2
ARIA	MUS	4	M	2
INVENTI	MUS	4	M	1
BATTSTAR	MUS	4	M	2
BOND2	MUS	4	M	2
CLOSENCT	MUS	4	M	2
SCARBORO	MUS	4	M	1
FUGUEINC	MUS	4	M	1
MINUET	MUS	4	M	1
LONGTIME	MUS	4	M	2
MESSIAH	MUS	4	M	3

* PD-22 MUSIC-1

LOADM "NAME/MUS" EXEC TO PLAY MUSIC THROUGH TV OR MON.

ADDPLAY	BAS	0	B	1
DEPLAY	BAS	0	B	1
MSQUEZ	BAS	0	B	2
ALSOSPAK	MUS	2	B	5
BOOGIE	MUS	2	B	5
CIRCUS	MUS	2	B	5
CLOWN	MUS	2	B	2
CLOWNS	MUS	2	B	4
HAYDEN	MUS	2	B	8
JBGOOD	MUS	2	B	4
PEACE	MUS	2	B	2
PEACH	MUS	2	B	5
PUFF	MUS	2	B	6
GOODDIEY	MUS	2	B	4

* PD-23 MUSIC-2

LOADM "NAME/MUS" EXEC TO PLAY MUSIC THROUGH TV OR MON.

ADDPLAY	BAS	0	B	1
DEPLAY	BAS	0	B	1
MSQUEZ	BAS	0	B	2
RAIN	MUS	2	B	2
SONATA3	MUS	2	B	3
STRAV	MUS	2	B	4
FOGGY	MUS	2	B	4
FUNERAL	MUS	2	B	3
HARDDAY	MUS	2	B	2
INVENT	MUS	2	B	2
INVENT11	MUS	2	B	3
INVENT15	MUS	2	B	3
INVENT7	MUS	2	B	3
INVENT8	MUS	2	B	2
JOPLIN	MUS	2	B	4
KHAN	MUS	2	B	6

* PD-24 MUSIC-3

LOADM "NAME/MUS" EXEC TO PLAY MUSIC THROUGH TV OR MON.

ADDPLAY	BAS	0	B	1
DEPLAY	BAS	0	B	1
MSQUEZ	BAS	0	B	2
PEANUTS	MUS	2	B	3
ROCK	MUS	2	B	5
ROXANNE	MUS	2	B	5
SCHERZO	MUS	2	B	2
TEACH	MUS	2	B	2
PIANOMAN	MUS	2	B	5
STRANGER	MUS	2	B	5
CAMELOT	MUS	2	B	4
CHACONNE	MUS	2	B	8
DIAMOND	MUS	2	B	3
DOWNROAD	MUS	2	B	4
FANTASY1	MUS	2	B	2

* PD-25 MUSIC-4

LOADM "NAME/MUS" EXEC TO PLAY MUSIC THROUGH TV OR MON.

FANTASY2	MUS	2	B	3
GRENGRAS	MUS	2	B	4
HUMOR	MUS	2	B	4
INCROW	MUS	2	B	3
STARWARS	MUS	2	B	2
SUITEGM	MUS	2	B	6
SUPERMAN	MUS	2	B	2
WHENIM64	MUS	2	B	4
ROOTBEER	MUS	2	B	7
WAYUARE	MUS	2	B	3
AXELF	MUS	2	B	2
TOCATTA	MUS	2	B	3

* PD-26 LAST WILL

LOAN	BAS	0	B	1
LASTWILL	BAS	0	B	6
IMEGA	BAS	0	B	3
AWARI	BAS	0	B	1
BACARAT	BAS	0	B	2
BAGELS	BAS	0	B	1
BLACKJAC	BAS	0	B	1
CHUCK	BAS	0	B	1
CONCENTR	BAS	0	B	1
CUBES	BAS	0	B	2

* PD-27 GAMES

DEFUZE	BAS	0	B	1
DR ZEE	BAS	0	B	1
FLIPFLOP	BAS	0	B	1
GO-FISH	BAS	0	B	2
HANGMAN	BAS	0	B	2
HIGHLOW	BAS	0	B	1
JACKPOT	BAS	0	B	1
KEYS	BAS	0	B	1
L E M	BAS	O	B	3
LUNARLD	BAS	0	B	2
NUMBERS	BAS	0	B	1
OBSTACLE	BAS	0	B	1
POOLGAME	BAS	0	B	4
RETURN	BAS	0	B	1
REVERSI	BAS	0	B	2
STARTREK	BAS	0	B	2
TTREK	BAS	0	B	3

PD-28 COMM. CC-TALK, BBS, TERM

BBS'S	DAT	1	A	1
CCT	IO	2	B	1
CCTALK	BAS	0	B	1
CNFG40V1	BAS	0	A	5
CNFG40V2	BAS	0	A	4
CTLKEY	BAS	1	A	1
MTERM1	DOC	1	A	11
MTERM2	DOC	1	A	8
MTERM40	BIN	2	B	8
REDIAL	BAS	0	A	1
PACREDIA	BAS	0	A	1

PD-29 COMM, WORD PRO, GAMES

GOSTSHIP	BAS	0	B	8
INT RATE	BAS	0	B	2
INVSTANL	PC	0	B	4
MENU	BAS	0	B	4
MOTOJUMP	BAS	0	B	3
SCREEN	MAX	2	B	6
SCREEN1	BIN	2	B	3
SCREEN2	BIN	2	B	3
SCREEN2	MAX	2	B	6
STRINGTU	BAS	0	B	4
TTERM	DSK	2	B	4
USING	BAS	0	B	3
WF-DOC	JP	0	B	2
WORDFILE	JP	0	B	4
PARM1	DAT	1	A	1

PD-30 CHECK BOOK, UTILITIES

CHECKBOK	BAS	0	B	4
CHECKBOK	DOC	1	A	9
DIRR	CMD	2	B	1
DVIEW	BAS	0	B	1
FILEMAID	BAS	0	B	2
LISTER	BAS	0	B	1
PAINTPOT	BAS	0	B	4
SCREEN	MAX	2	B	6
SCREEN1	BIN	2	B	3
SCREEN2	BIN	2	B	3
SCREEN2	MAX	2	B	6
SPECZAP	BAS	0	B	5
TAPETYPE	BIN	2	B	1
TTERM	DSK	2	B	4
DVIEW	DSK	0	B	1
MENU	BAS	0	B	4

PD-31

PIRATES TREASURE - As you explore the cave looking for the treasure, a picture appears on the screen as you go from room to room. These pictures are loaded from disk. A computer with a disk drive is required and a ramdisk is preferred.

PD-32

Color Computer 3 moving pictures. Consists of a beautiful waterfall and a colorful bouncing ball.

WATRFALL	BAS	0	B	1
WATRFALL	BIN	2	B	1
WATRFALL	MGE	1	B	
BALL	BAS	0	B	1
BALL2	BAS	0	B	1
BOUNCE	BIN	2	B	1
BALL2	HR1	2	B	4
BALL2	HR2	2	B	4
BALL2	HR3	2	B	4
BALL2	HR4	2	B	4

PD-33

EDUCTIONAL PROGRAMS

ABBREV	BAS	0	B	4
ABCPOP	BAS	0	B	3
ALPHAAL	BAS	0	B	1
EDUCATE	BAS	0	B	1
HANGP	BAS	0	B	1
HOMONYM	BAS	0	B	1
SPELWORD	BAS	0	B	1
MATH	BAS	0	B	2
DRILL	BAS	0	B	2
MLTP	BAS	0	B	1
ROUND	BAS	0	B	2
AREA	BAS	0	B	6
METCONV	BAS	0	B	3
NUMBERS	BAS	0	B	2
SIEVE	BAS	0	B	1

PD 34

!! BULLETIN BOARD!! With this software you can run your own bulletin board at 300 or 1200 baud. Instructions are included.

SCF	EDI	0	B	3
SMF	EDI	0	B	4
SUL	EDI	0	B	4
SMP	EDI	0	B	2
64K	BAS	0	B	1
STARTUP	BAS	0	B	2
COTERM	BIN	2	B	1
USER	SYS	0	B	8
COBBS	SYS	0	B	9
STARTI	DOC	1	A	6
USER	DOC	1	A	1
COBBSREV	DOC	1	A	5
OPERAT	DOC	1	A	7
SMH	EDI	0	B	3
MENU	DOC	1	A	11

PD 35

ADDRESS FILES AND FINANCE PROGRAMS

PHONE	BAS	0	B	1
LABELPRT	BAS	0	B	1
LETTER	BAS	0	B	3
MAILLST	BAS	0	B	2
PHONLST	BAS	0	B	1
MINIWORD	BAS	0	B	2
LNWIDTH	BAS	0	B	1
CHKWRITE	BAS	0	B	2
CHKANAL	BAS	0	B	4
PRNTCHK	BAS	0	A	1
CHECKS	BAS	0	B	4
CHCKSTUB	BAS	0	B	1
TOTALS	DAT	1	A	1
CHECKS	DAT	1	A	1
GRAPH	BAS	0	B	4
LOAN	BAS	0	B	3
CALC	BAS	0	B	1
PAYMENT	BAS	0	B	1
CASHJNL	BAS	0	B	3
AMORT	BAS	0	B	3

PD 36

COMP.SCIENCE PGMS 1: These programs are tutorials on basic programming.

COMPSC1	BAS	0	B	8
COMPSC2	BAS	0	B	3
COMPSC3	BAS	0	B	9
COMPSC4	BAS	0	B	5
COMPSC5	BAS	0	B	9
COMPSC6	BAS	0	B	5
GETPUT	BAS	0	B	2

PD 37

COMP. SCIENCE PGMS 2:

These programs are tutorials on basic programming.

IFTHEN	BAS	0	B	8
EXTENDED	BAS	0	B	2
GETPUT	BAS	0	B	2
COMPSCI8	BAS	0	B	8
COMPSCI9	BAS	0	B	5
COMPSCI7	BAS	0	B	7
EXTDEMO	BAS	0	B	3

* PD 38

EDUCATIONAL PROGRAMS These programs are excellent learning tools for school children.

ABBREV	BAS	0	B	4
ABCPOP	BAS	0	B	3
ALPHAAL	BAS	0	B	1
EDUCATE	BAS	0	B	1
HANGP	BAS	0	B	1
HOMONYM	BAS	0	B	1
SPELWORD	BAS	0	B	2
MATH	BAS	0	B	2
DRILL	BAS	0	B	2
MLTP	BAS	0	B	1
ROUND	BAS	0	B	2
AREA	BAS	0	B	6
METCONV	BAS	0	B	3
NUMBERS	BAS	0	B	2

PD 39

ADDRESS FILES AND FINANCE PROGRAMS

PHONE	BAS	0	B	1
LABELPRT	BAS	0	B	1
LETTER	BAS	0	B	3
MAILIST	BAS	0	B	1
WORDPROC	BAS	0	B	3
MAILLST	BAS	0	B	2
PHONLST	BAS	0	B	1
MINIWORD	BAS	0	B	2
LNWIDTH	BAS	0	B	1
CHKWRITE	BAS	0	B	2
CHKANAL	BAS	0	B	4
PRNTCHK	BAS	0	A	1
CHECKS	BAS	0	B	4
CHCKSTUB	BAS	0	B	1
TOTALS	DAT	1	A	1
CHECKS	DAT	1	A	1
GRAPH	BAS	0	B	4
LOAN	BAS	0	B	3
CALC	BAS	0	B	1
PAYMENT	BAS	0	B	1
CASHJNL	BAS	0	B	3
AMORT	BAS	0	B	3

*PD-40

TAPE-DSK & DSK-TAPE With these programs you can copy a disk to tape or a tape to disk.

T2D	BIN	2	B	2
DTCOPY	BIN	2	B	1
DSK-TP	BAS	0	B	1
DISKLIST	BAS	0	B	1
DIRLIST	BAS	0	B	2
DISKDUMP	BAS	0	B	1
CASSDIR	BAS	0	B	1

Pictures can be loaded with CoCo MAX or our PIXFILES/BAS progras. They can be printed on a graphics printer. See Dynamic Color News issue #44 for a graphics screen dump program. Our DYPRINT package allows large blown up pictures to be printed using standard print.

All program collections are available on disk. Collections with a * are also available on tape.

1-4 $4.95, 5-9 $4.50, 10 - $4.00

Add $1 shipping. Specify Tape or Disk. Checks, VISA, or MC.

DYNAMIC ELECTRONICS
BOX 898 (205) 773-2758
HARTSELLE, AL 35640

• PD-41
Pictu re files

STAMPS	MAX	2	B	3
STARTREK	MAX	2	B	3
ST-TREK2	MAX	2	B	3
SCHOOL	MAX	2	B	3
SATURN	MAX	2	B	3
ESCHER	MAX	2	B	3
LABOR	MAX	2	B	3
MASK	MAX	2	B	3
BUG BOX	MAX	2	B	3
SPACE	MAX	2	B	3
EASTER	MAX	2	B	3
SPACE 2	MAX	2	B	3
POPEYE	MAX	2	B	3
GARFIELD	MAX	2	B	3
BEETLE B	MAX	2	B	3
POLO	MAX	2	B	3
HAGAR	MAX	2	B	3
X-PAD	MAX	2	B	3
CASTLE	MAX	2	B	3
MUSIC TV	MAX	2	B	3
COCO	MAX	2	B	3

• PD-42
Pictu re files

TITLES	MAX	2	B	3
PIXFILES	BAS	0	B	3
THOLIAN	MAX	2	B	3
3001AD	MAX	2	B	3
F15	MAX	2	B	3
QUEEN	MAX	2	B	3
BRONCOS	MAX	2	B	3
STARTREK	MAX	2	B	3
ROOM	MAX	2	B	3
RAMBO	MAX	2	B	3
OWL	MAX	2	B	3
ENTERPR	MAX	2	B	3
STAR-T 3	MAX	2	B	3
NCC-1701	MAX	2	B	3
SAT-2	MAX	2	B	3
ATMOSP	MAX	2	B	3
STARWARS	MAX	2	B	3
ORIENTAL	MAX	2	B	3

• PD-43
Picture files

STAMP	MAX	2	B	3
STRIPE	MAX	2	B	3
WOMAN	MAX	2	B	3
BLUEJAY	MAX	2	B	3
LUCY	MAX	2	B	3
OLD EHO	MAX	2	B	3
MENU1	MAX	2	B	3
OWL	MAX	2	B	3
VAN GOG	MAX	2	B	3
WOMAN1	MAX	2	B	3
PSN	MAX	2	B	3
DUCKPOND	MAX	2	B	3
RANGER	MAX	2	B	3
PLANET	MAX	2	B	3
CHRSTMAS	MAX	2	B	3
PEACE	MAX	2	B	3
WOMAN3	MAX	2	B	3
HAWK	MAX	2	B	3
PHASER	MAX	2	B	3
PIXFILES	BAS	0	B	3

PD-44
Terminal program with documentation. This will work with the CoCo- 3. Instructions are included.

MTRM43	BIN	2	B	B
CONFIG43	BAS	0	B	4
MTSTART	BAS	0	B	4
MTERM1	DOC	1	A	11
MTERM2	DOC	1	A	B
MTERM3	DOC	1	A	7
DOS BOOT	DAT	1	A	1
*	*	0	B	1
**	*	1	A	1
READDOC	BAS	0	B	1

• PD-45
Picture Files

DRAGON	MAX	2	B	3
HOT LIPS	MAX	2	B	3
ANIMALS	MAX	2	B	3
CLOWN F	MAX	2	B	3
FISH	MAX	2	B	3
3 MEN	MAX	2	B	3
S MAP	MAX	2	B	3
BUOS	MAX	2	B	3
CFISH	MAX	2	B	3
HERO	MAX	2	B	3
WMAP	MAX	2	B	3
GSCOTT	MAX	2	B	3
STATES	MAX	2	B	3
HORSE	MAX	2	B	3
CROSS	MAX	2	B	3
FOODW	MAX	2	B	3
RSTONE	MAX	2	B	3
COCO	MAX	2	B	3
ALIEN	MAX	2	B	3
PIXFILES	BAS	0	B	3

• PD-46
Talk and Music Files (C)LOADM "FILE" then EXEC.

TALK	BIN	2	B	11
TALK2	BIN	2	B	11
WILLTELL	BIN	2	B	9
MUSICBOX	BIN	2	B	1
BEATLES	BIN	2	B	4
JUMP	BIN	2	B	5
GRELN	BIN	2	B	5
GHOST	BIN	2	B	4
JINGLE	BIN	2	B	3
WORLD	BIN	2	B	5
CTRYROAD	BIN	2	B	2

• PD-47

Miscella neo usPgms

T	BAS	0	B	2
SANTEE2	BAS	0	B	1
MILEAGE	BAS	0	B	1
M	BAS	0	B	1
DIGITS	BAS	0	B	1
NUMBLIST	BAS	0	B	1
COUNT	BAS	0	B	1
SC	BAS	0	B	1
DRAWTEXT	BAS	0	B	1
SAMPLE	BAS	0	B	1
ORSCRWRT	BAS	0	B	2
HRTEXT2	BAS	0	B	3
DRAW	BAS	0	B	2
WRITER	BAS	0	B	1
TYPEBET	BAS	0	B	2
WRITEBET	BAS	0	B	2
TEXT2	BAS	0	B	2
SANTEE	BAS	0	B	2
SHUTTLE	BAS	0	B	1
AJOCK	BAS	0	B	1
PLATFORM	BAS	0	B	1
MAZE	BAS	0	B	4
DISKZAPR	BAS	0	B	2
ZAP	BAS	0	B	3
DETMSHIP	BAS	0	B	3
BACKUP35	BAS	0	B	1
BOOT	BAS	0	B	1
SCRNLIST	BAS	0	B	1
DOSSTART	BAS	0	B	1
LABEL	BAS	0	B	2
DSKDSABL	BAS	0	B	1
NOFREEO	BAS	0	B	1
FORMATER	BAS	0	B	1
ROMRAM	BIN	2	B	1
SUPDUP	BIN	2	B	1
TESTTEXT	BAS	0	B	1

• PD-48

Mis cellaneous Pgms

EXTBAS	BAS	0	B	3
DISAPEAR	BAS	0	B	1
PAINT	BAS	0	B	1
DATA	BIN	2	B	1
DATA2	BIN	2	B	1
SCRDATA	BIN	2	B	1
FILL2	BIN	2	B	2
QUADDRAW	BAS	0	B	1
CELTIC	BAB	0	B	2
ALL RAM	BAS	0	B	1
CHAROEN	BIN	2	B	1
ROMRAM	BIN	2	B	1
OBSTACLE	BAS	0	B	1
64K RAM	BAS	0	B	1
COLORSEL	BAS	0	B	1
TRIG	BAS	0	B	4
ALGEBRA	BAS	0	B	4
PLAY	BAS	0	B	1
STATECAP	BAS	0	B	2
MLSOUNDS	BAS	0	B	1
ROTATION	BAS	0	B	2
PARABOLA	BAS	0	B	2
INSTAPIC	BAS	0	B	1
CLOVER	BAS	0	B	1
MAT-PLOT	BAS	0	B	1
WHEEL 1	BAS	0	B	1
LETTER-R	PAR	1	A	1
3-LINES	ROT	1	A	1
TRAPZOID	ROT	1	A	2
PYRAMID	ROT	1	A	2
CUBE	ROT	1	A	3
51X24	BAS	0	B	2
WINDOW	BAS	0	B	5
GGPRTSU	BAS	0	B	1
KALEIDO	BAS	D	B	1
OKB3APR1	BAS	0	B	1
NUMCNVIR	BAS	0	B	1
ADVRTN	BAS	0	B	1

•PD-49

Miscellaneo usPgms.

BC	BIN	2	B	10
PEDRO	BIN	2	B	11
BLOCKADE	BAS	0	B	3
REPEAT	BAS	0	B	1
AIRPLANE	BAS	0	B	1
BUSTOUT	BAS	0	B	1
GOLF	BAS	0	B	7
CITY	BAS	0	B	2
AIR-RAID	BAS	0	B	2
MAZE	BAS	0	B	4
DUALDUP	BIN	2	B	2
DIRMAP	BAS	0	B	3
CHESS	BAS	0	B	5
WHATZIT	BAS	0	B	4
BATLSNIP	BAS	0	B	3
SP*ROCKS	BAS	0	B	1

• PD-50
Miscellaneo us PGMS

GOBBLER	BAS	0	B	2
PYTHON	BAS	0	B	2
LUNAR	BAS	0	B	2
LUNALANA	BAS	0	B	1
AMAZING	BAS	0	B	2
BALLOON	BAS	0	B	1
VAPORWRM	BAS	0	B	2
ABM	BAS	0	B	3
BULLSEYE	BAS	0	B	1
CRASH	BAS	0	B	1
DOTS	BAS	0	B	3
E-16	BAS	0	B	3
KRYPTON	ART	2	B	3
KRYPTON	BAS	0	B	1
KRYPTON	GAM	0	B	1
NUKEATTK	BAS	0	B	2
ASTEROID	BAS	0	B	1
PRIX	BAS	0	B	2
ONE	BIN	2	B	3
TWO	BIN	2	B	3
THREE	BIN	2	B	3
FOUR	BIN	2	B	3
TEMPEST	BAS	0	B	2
SNAKE	BAS	0	B	2
SCORE	DAT	1	A	1
OTHELLO	BAS	0	B	4
ROCKS	BAS	0	B	3
LANDER	BAS	0	B	2

• PD-51
Games & Programs

DRAGRACE	BAS	0	B	1
WORMER	BAS	0	B	2
SIMON	BAS	0	B	2
RIDER	BAS	0	B	2
MISSILE	BAS	0	B	3
LETSHOOT	BAS	0	B	1
SHOOTOAL	BAS	0	B	2
MISSILE2	BAS	0	B	3
FENCE	BAS	0	B	3
BANDIT	BAS	0	B	1
CHICKEN	BAS	0	B	2
MAXIMUM	BAS	0	B	3
FLIGHT	BAS	0	B	2
COVERUP	BAS	0	B	2
WORLDMAP	BAS	0	B	4
POUNCE	BAS	0	B	1
MARTIANS	BAS	0	B	2
FINDIT	BAS	0	B	3
SCRAMBLE	BAS	0	B	5
BOUNBABY	BAS	0	B	2
CHICK	BAS	0	B	3
BOBO	BAS	0	B	3
RUBIC	BAS	0	B	4
HCJUMP	BAS	0	B	3

• PD-52
Pictu re files

COCO	MAX	2	B	6
COL COCO	MAX	2	B	6
MOOSHEAD	MAX	2	B	6
COKE	MAX	2	B	6
CUBS	MAX	2	B	6
REDS	MAX	2	B	6
BREAKERS	MAX	2	B	6
USFL	MAX	2	B	6
SPACE	BIN	2	B	3
GIZMO	MAX	2	B	3
DINASOUR	MAX	2	B	3

• PO 53
Pict ure Files

INDIAN	MAX	2	B	6
HOMECOME	MAX	2	B	6
GRIN	BIN	2	B	3
TARD	BIN	2	B	3
STUD	BIN	2	B	3
COMET	BIN	2	B	3
DESERT	BIN	2	B	3
FOOD	BIN	2	B	3
SMIRK	BIN	2	B	3
PLAYA	BIN	2	B	3
HELLO	BIN	2	B	3
GROVER	BIN	2	B	3
DRIVE IN	BIN	2	B	3
TIME	BIN	2	B	3
KOALA	BIN	2	B	3
PATTERN	BIN	2	B	3
HAGAR	BIN	2	B	3
CHIPS	BIN	2	B	3

• PD 54
Picture Files

PENTAGON	PIC	2	B	3
GRID 2	PIC	2	B	3
SNOWFLAK	PIC	2	B	3
CONETUNL	PIC	2	B	3
4-POINT	PIC	2	B	3
BALTSTR	MAX	2	B	3
CARTOON	MAX	2	B	3
HUELEWIS	MAX	2	B	3
STARTREK	MAX	2	B	3
HOUSE1	MAX	2	B	6
HOUSE2	MAX	2	B	6
LIFECYCL	MAX	2	B	6
COCOMAO	MAX	2	B	3
MASCASTL	MAX	2	B	3
COLUMBIA	MAX	2	B	3
POLO	MAX	2	B	3
ET	BAS	0	B	7
WHEEL 1	PIC	2	B	3

• PD-55
Picture Files

PARKERPT	MAX	2	B	3
TOWER	PIC	2	B	3
TOWER2	PIC	2	B	3
SCREEN	PIC	2	B	3
BOMB	PIC	2	B	3
ANDRON	PIC	2	B	3
SALE	PIC	2	B	3
CHIPS	PIC	2	B	3
TUNLROAD	BIN	2	B	3
LONEROAD	BIN	2	B	3
CITYROAD	BIN	2	B	3
LAKEROAD	BIN	2	B	3
CROSROAD	BIN	2	B	3
BLACK	BIN	2	B	3
CAL1	BIN	2	B	3
CAL2	BIN	2	B	3
CAL3	BIN	2	B	3
3-LEAF	PIC	2	B	3
5-STARS	PIC	2	B	3
SPHERE	PIC	2	B	3
15-LEAF	PIC	2	B	3

PD-56

Glossary, Memory Maps, Programs

COCO	VIP	1	A	4
VIP ON 3	VIP	1	A	1
BEEP	VIP	1	A	1
MCTRM3	VIP	1	A	1
GLOSSARY	VIP	1	A	7
POKEPEEK	VIP	1	A	17
WIDTH	VIP	1	A	1
COCO 3	VIP	1	A	17
MISSLES	BAS	0	B	2
CLOCK	BAS	0	B	1
JET	BAS	0	B	4

• PD-57
Pict ureFiles

VAMPIRE	PIC	2	B	3
ATLANTA	BAS	0	B	3
NOGHOST	PIC	2	B	3
AIRPORT	BAS	0	B	4
S EASTON	BAS	0	B	4
1SMLSTEP	BAS	0	B	4
HAGAR	PIC	2	B	3
SUNSET	BAS	0	B	3
S NICKS	BAS	0	B	4
SNOOPY1	BAS	0	B	3
MICKEY	BIN	1	B	B
DONALD	BIN	2	B	B
SNOOPY2	BAS	0	B	4
SNOOPY3	BAS	0	B	4
SNOOPY4	BAS	0	B	4

• PD-58
Miscellaneo us Pgms.

DISKLIST	BAS	0	B	1
DIRLIST	BAS	0	B	2
ML ADDR	BAS	0	B	1
DISKDUMP	BAS	0	B	1
PRINUTIL	BAS	0	B	2
CALPRINT	BAS	0	B	3
ALPHSONG	BAS	0	B	1
PAINT	BAS	0	B	1
DOGPICT	BAS	0	B	2
EVADER	BAS	D	B	1
NUKATTC	BAS	0	B	2
BASICMAP	BAS	0	B	3
JOYPAINT	BAS	0	B	1
PUMPKIN	BAS	0	B	1
HOMOYMS	BAS	0	B	1
ABBREV	BAS	0	B	4
CONVERT	BAS	0	B	3
CASSDIR	BAS	0	B	1
CVERT	BAS	0	B	1
FLASCARD	BAS	0	B	1
MESSAGE	BAS	0	B	1
RELOCAT	BAS	0	B	1
COUNT	BAS	0	B	1
CALENDAR	BAS	0	B	1
DOGS	BAS	0	B	1
DOGFIGHR	BAS	0	B	1
BEAST	BAS	0	B	1

• PD-59
GAMES, UTILITIES

64X64F	BAS	0	B	1
RND#'S	BAS	0	B	1
SCROLLER	BAS	0	B	1
COCOBUG	BAS	0	B	2
DRWBOARD	BAS	0	B	1
SPACE	BAS	0	B	1
DIR-ADDR	BAS	0	B	1
BACKGAMN	BIN	2	B	2
CHESS	BIN	2	B	3
BATTLE	BIN	2	B	2
GERM	BIN	2	B	1
BLEEP	BAS	0	B	2
TICKER	BAS	0	B	3
LEAKYTAP	BAS	0	B	3
UTOPIAN	BAS	0	B	4
COLORDOT	BAS	Q	B	3
STAYALIV	BAS	0	B	2
TIMEFLT	BAS	0	B	3
NAVYGUNS	BAS	0	B	2
ATACMAN	BAS	0	B	3
CALENDAR	BAS	0	B	1
POKER2S	BAS	0	B	1
VIEWERS	BAS	0	B	1
STUFF	BAS	0	B	1

PRODUCT REVIEWS

This section is open to all producers and dealers of color computer products. We will review your 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.

VIP SPELLER

Last month we reviewed VIP WRITER which is a super word processor by S D Enterprises. VIP Speller is included in the package for the color computer 3. A speller checks the spelling of words in a text file. It uses its own dictionary as a reference and displays words that are not spelled correctly. The file does not have to be a VIP file, but can be any ASCII file.

To start the process type LOADM"SPELLER" ‹ENTER›. It loads and checks the computer's memory. It then displays a menu which contains the following:

```
S SPELL CHECK FILE
E EXAMINE DICTIONARY
R REMOVE WORDS FROM
    DICTIONARY
A ADD WORDS TO DICTIONARY
Q QUIT
```

To spell check a file select "S". The computer asks for the file to spell check. Enter the file and its extension if the extension is not VIP. It then asks for the dictionary file name. If there are several dictionaries then the file name for the desired one should be entered. To use the default dictionary press the enter key and the "DICT.DOC:0" file is selected. It then asks for a dict index table file name. Again press enter and the default index table will be used. Next the following appears:

```
WORD DETERMINATION TYPE

H NORMAL USE
L LITERAL MODE
W WITH NUMBERS
Q QUIT

SELECTION:
```

Press a letter for the type desired. The program reads in the file and sorts the unique words into alphabetical order. It then requests inserting the spelling disk and begins checking the words against those on the speller. It displays the letters as they are being checked. After this is completed they can be compared against another dictionary. Press enter if this is not desired. After a few seconds it asks for the disk with the original file to be inserted. The following options are available:

```
M MARK WORDS IN FILE
C CORRECT WORDS
S SHOW 'WRONG' WORDS
W WRITE WORDS TO NEW FILE
P LINE PRINT WORDS
A ADD ALL WORDS TO DICT
D DISK COMMANDS
Q QUIT
```

If "M" is selected the present file is renamed using the /BAK extension. A new file is created with each of the spelling errors marked with a # sign preceeding each one. If "S" is selected then the errors are shown on the screen. The words can be written to a new file. Press "W" for this option and enter the name of the file. To print the words to a printer press "P". To correct the words press "C". You can then examine each word and enter the correct spelling. A new file is created with the same name and the /BAK extension. This file has the corrected words in it.

Words can be added to or deleted from a a dictionary. The spelling can be checked against several dictionaries.

We found VIP SPELLER to be a superb spelling checker. The cost of VIP WRITER is $79.95 including VIP Speller. SD Enterprises, P.O. Box 1233, Gresham, OR 97030.

Question & Answers

Hi Bill

Just thought I should drop you a line and let you know how pleased I am with your Dynamic Color News. I am also a ham W9RKU and am interested in Packet Radio Transmissins on 2 Meter FM.

I especially like your small programs that the average hobbiest can type in the program material even if it's games or utilities or whatever. It's good practice but I would like it typed at 32 line structure.

I have a lot of faith in your publication to be honest with you I like Dynamic News better than Rainbow.

Bill enclosed please find a check for this Months disc programs #47 Mar 88.

I remain

Paul Flaishaker

ANSWER: Paul thanks for your letter. We list our programs in 32 or 42 characters /line format. We reduce the 42 character listings. We indent each line that is continued to make the programs easier to read. Read my editorial in this issue. I am still working on decoding the packet signals but am going to have to switch to a joystick to get amplitude variations. I will have more on this next month. Thank you for the kind words and for your support.

* * *

Dear Bill

I have been reading your ham radio articles for the past year and have just renewed for another year. So far the articles have been very interesting. I hold a general class lic. I am involved with 2 mtrs as well as HF. I work a lot of packet radio using my COCO 2 and a midland 13-510 with a KKPC-2 TNC. I am interested to see how your program will work. I am using the Mickey-term terminal program. The program works quite well. I am using your RTTY program. It seems to work well, however a good filter would help. I am going to work up something to put in line other than the diode system you used. One thing I would like to see is how to send the info received to a printer such as the Dmp105. I would like to see some articles on WEFAX. , I am using a program from Rainbow from Feb. 1985 which, works well but is setup for Epson graphics. This lets me out as far as a printout is concerned. I would like to see what you can come up with that will work with a Dmp105. I would also like to see something on slow scan. I have a tape that I recorded from W00RE on board the SPACE SHUTTLE CHALLANGER a short time before the accident, that I would like to be able to see.

I ordered a back issue that I have not received yet and also you might check your records on my subscription which should have expired this month 2/88 and when I renewed you show my exp. date of 1/89. I like reading Dynamic Color News and think you are doing a fine job.

73's de N6EHI
William (Bill) Wise

ANSWER: First of all a filter will help for any type of communications. I am switching to a joystick port so that I can detect amplitude variations and use digital filtering within the software. The cassette port is only 1 bit where the joystick ports are 5 bits each.

The software would have to be modified to use other printers with the RTTY program. I don't have a patch for using Radio Shack printers. I printed a WEFAX program last month.

Dean has corrected your subscription and you should have received your back issue by now. If not let us know. Thanks for your letter.

* * *

Dear Bill,

Just a short letter to tell you how thrilled I am with the WEFAX in the March issue of "Dynamic Color News". I'm watching the Fax program run right now. It only took a short time to have this typed up and running.

I've been a ham for 4 years and enjoy most of the modes. But I have been always thinking about weather fax, and figued it was out of my league. I purchased a CoCo 2 extended about 1 year ago and have been using RTTY and also built your interface to use on CW transmitt. I still use the "old ear" to receive and will keep that going as long as I can.

I've been searching out other FAX stations and have been receiving photos and sometimes just old looking prints in the 17,18,19 MHZ Bands. My rig is a Kenwood 30S and is a full coverage receiver. So there is a lot of searching still ahead. I'm looking forward to your slow scan programs. But the interfaceless Packet has really got my attention. I've been holding off buying a packet modem as I've been hearing some negative reports on the mode. But your program will be a real cost saver and probably sell a few CoCo's.

My only question is "how can the RTTY program be made to change BAUD Rates?" Many of the world press and other broadcasters run 300 WPM & higher.

Well keep up the great work on a wonderful magazine. I've shown a few hams your publication and hope they see the light.

Also I am interested in a bulletin board type program that would give me a reason to buy a phone modem.

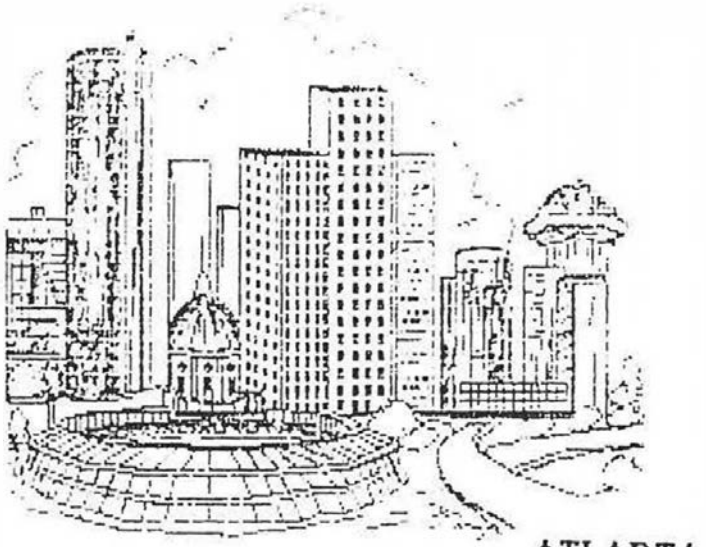
73's Doug Alderton

ANSWER: Doug thanks for your letter. My work on the packet is slower than I had hoped it would be. However I have copied some packet signals that I could piece together. I am changing over to a joystick port so I can get amplitude detection for better decoding.

There is a RTTY program in the January 1988 issue of 73 Magazine. We typed in the program but it has some errors. It operated at 3 rates. If anyone has a corrected version of the program we would like to have a copy to print. We will trade a couple of public domain programs for a working copy.

Our PD-44 public domain terminal program will work with the CoCo 3 at 300 baud using the printer port. I use it to transfer files from my model 100 to the CoCo.

Also we have a bulletin board program PD-34. Thanks for your letter and your comments.



ATLANTA

This is a picture of Atlanta. If you have never been to Atlanta then this scene will give you an idea of how the city looks. This is a PMODE 4 picture and will work with all color computers. The data is carried in DATA statements as two hex characters for each byte. The program can be typed in as listed. It takes a few minutes for the data to be read and poked into memory. The memory is displayed on the screen in 100 byte increments.

After the data is transferred to memory, a menu displays options which allow the picture to be saved to a cassette or disk, or to be viewed. Lines 60 and 62 allow the picture to be viewed. The picture can be printed on a printer with our graphics print program in our issue #44 (December 1987). It can also be used with COCO MAX by renaming it as follows:

```
RENAME "ATLANTA/BIN" TO "ATLANTA/MAX
```

ATLANTA PROGRAM LISTING

```
2 PCLEAR 4:CLS
4 PRINT"* * * * * ATLANTA * * * * *
6 PRINT
10 PRINT:PRINT"STANDBY WHILE MACHINE LANGU
   AGE  PROGRAM IS BEING GENERATED":PRINT
12 EN=256*PEEK(25):M=EN-6144:BE=M
14 READ X$
16 IF X$="@" THEN 42
18 L=LEN(X$)
20 FOR J=1 TO L STEP 2
22 A$=MID$(X$,J,2):B=PEEK(M)
24 C$=LEFT$(A$,1):D$=RIGHT$(A$,1)
26 X=ASC(C$):Y=ASC(D$):X=X-48:Y=Y-48
28 IF X>9 THEN X=X-7
30 IF Y>9 THEN Y=Y-7
32 V=16*X+Y:POKE M,V
34 M=M+1:IF M=EN THEN 42
36 NEXT J
38 PRINTM
40 GOTO14
42 PRINT"DATA IS TRANSFERRED":PRINT"1 VIEW
   PICTURE":PRINT"2 SAVE TO CASSETTE":PRI
   NT"3 SAVE TO DISK"
44 '
45 INPUT"ENTER NUMBER";X
50 ON X GO TO 60,70,80
55 END
60 PMODE 4,1:SCREEN 1,1
62 GOTO 60
70 PRINT"SAVING TO THE CASSETTE"
75 CSAVEM"ATLANTA",BE,EN,BE
77 END
80 PRINT"SAVING TO A DISK"
85 SAVEM"ATLANTA",BE,EN,BE
90 END
100 DATA FFFFFFFFFFFFFFFFFFFFFFFF007FFFFF
   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
   FFFFFFFFFFFFFFFCFF8FFFFFFFFFFFFFFFFFFF
   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
   FF3FFF0FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
   FFFFFFFFFFF
110 DATA FFFFFFFFFFFFFF3FCFFFFF3FFFFFFFFFFF
   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
   FFFFFFF3FFFFFDFFFFFFFFFFFFFFFFFFFFFFFF
   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEFFFFFE
   7FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
   FFFFFFFFFF
120 DATA FFFFFFEFFFFFFFBFFFFFFFFFFFFFFFFF
   FFFFFFFFFFFFFFFFFFFFFFFFFFF83FFFFFFFDF
   FFFFFFDFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
   FFFFFFFFFFFFF09FFFFFFFDFFFFFFFDFFFFFFF
   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF0
   1FFFFFFFBF
130 DATA FFFFFFEFFFFFFFFFFFFFFFFFFFFFFFFF
   FFFFFFFFFFFFFFFFFF01FFFFFFFBFFFFFFEF
   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
   FFFFF75FFFFFFFFFFFFFFFF7FFFC3FFFFFFFFF
   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF75FFFFFF
   F5FFFFFFFF
140 DATA FFF3C7FFFFFFFFFFFFFFFFFFFFFFFFFF
   FFFFFF00037DFFFFFFF0FFFFFFFFFFFCFFF7F
   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF000007C7
   FFFFFCFFFFFFFFFFF8FFFFFBFFFFFFFFFFFFFF
   FFFFFFFFFFFFFFFFFC000007C5FF9FFDFFFFFFF
   FFE7FFFFFDF
150 DATA FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF80F
   FFF791FC67FBFFFFFFFFFFFFFFFFEFFFFFFFFF
   FFFFFFFFFFFFFFFFFFFFFF0F000175DF3F8FBFF
```

```
    FFFFFFFFFFFFFFFF7FFFFFFFFFFFFFFFFFFFFF
    FFFFFFFF382081745EFFE7FFFFFFFFFFFFFFFFF
    FF7FFFFFFFF
160 DATA FFFFFFFFFFFFFFFFFFFFFF482001759EF
    FFB7FFFFFFFFFFFFFFFFFBFFFFFFFFFFFFFFFF
    FFFFFFFFFFFE7F00020175DDFFFDFFFFFFFFFFF
    FFFFFFFBFFFFFFFFFFFFFFFFFFFFFFFFFF9FF
    0082A275DBFFFFFFFFFFFFFFFFFFFFFFFF7FFFF
    FFFFFFFFFFF
170 DATA FFFFFFFFFFE7FF48E03075DFFFFFFFFFF
    FFFFFFFFFFFFFFF9FFFFFFFFFFFFFFFFFFFFFF
    FFDFFF4022C175DFFFFFFFFFFFFFFFFFFFFFFF
    FDFFFFFFFFFFFFFFFFFFFFFFFFFFFFF40800075
    DFFFFFFFFFFFFFFFFFFFFFFFFFE3FFFFFFFFFF
    FFFFFFFFFF
180 DATA FFBFFF00808075DFFFFFFFFFFFFFFFFFF
    FFFFFFFCFFFFFFFFFFFFFFFFFFFFFFF7FFF49
    8284759FFFFFFFFFFFFFFF80000001FFFF7FFFF
    FFFFFFFFFFFFFFFFFF7FFF4882807DDFFFFFFF
    FFFFFFFB80000007FFFFFFFFFFFFFFFFFFFFFF
    FFFFF7FFF40
190 DATA 880461DFFFFFFFFFFFFFFBF8000001FF
    FFFFFFFFFFFFFFFFFFFFFFFF7FFF48809075DF
    FFFFFFFFFFFFFBFFFFFFFDFFFFFFFFFFFFFFFF
    FFFFFFFFFFBFFF48000175DFFFFFFFFFFFFFFF
    BFB8E318DFFFFFFFFFFFFFFFFFFFFFFFFFBFF
    F48009074DF
200 DATA FFFE7FFFFFFFFFBFB8E318DFFFFFFFFFF
    FF1FFFFFFFFFFFFFFDFFF00021074DFFFF6FFFF
    FFFFFFBFB8E318DFFFFFFFFFFF8FFFFFFFFFFF
    FFFDFFF00591274DFFFF9FFFFFFFFFFBFF8E318
    DFFFFFFFFFFE6FFFFFFFFFFFFFFFE7FF009B917
    CDFFFFDFFFF
210 DATA FFFFFFBFBFFFFFDFFFFFFFFFFFFFFFFFF
    FFFFFFFFFFFFF505F117CDFFFFFFFFFFFFFFFBF
    BFFFFFDFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF5
    05BA17CDFFFFFFFFFFFFFFFFBFB8E318DFFFFFFF
    FFFFFFFFFFFFFFFFFFFFFFFF4033817CDFFFFFF
    FFFFFFFFFBF
220 DATA B8E318DFFFFFFFFFFFFFFFFFFFFFFFFF
    FFFFF0833A17CDFFFFFFFFFFFFFFFFBFB8E318DF
    FFFFFFFFFFFFFFFFFFFFFFFFFFFF50BB817CD
    FFFFFFFFFFFFFFFFBFB8E318DFFFC3FFFFFFFFF
    FFFFFFFFFFFFFFFF40FBB57CDFFFFFFFFFFFFFF
    FBFBFFFFFDF
230 DATA FFBDFFFFFFFFFFFFFFFFFFFFFFFFF08B
    B817CDFFFFFFFFFFFFFFFBFBFFFFFDFFFFE7FFF
    FFFFFFFFFFFFFFFFFFFFFF583F8574DFFFFFFFF
    FFFFFFFBFB8E318DFFFFFFFFFFFFFFFFFFFFFFF
    FFFFFFFF483B8574DFFFFFFFFFFFFFFFBFB8E31
    8DFFFFF7FFF
240 DATA FFFFFFFFFFFFFFFFFFFFFF58B30174DFF
    FFFFFFFFFFFFFBFB8E318DFFFFFFFFFFFFFFFFF
    FFFFFFFFFFFBFF40B3A574DFFFFFFFFFFFFFFFB
    FB8E318DFFFFFDFFFFFFFFFFFFFFFFFFFFF97FF
    52FBA974DFFFFFFFFFFFFFFFBFBFFFFFDFFFFFE
    FFFFFFFFFF
250 DATA FFFFFFFFFC7FFF40FB1D7CDFFFFFFFFFF
    FFFFFBFBFFFFFDFFFFFE7FFFFFFFFFFFFFFFFF
    F3FFFF40FB3B7CDFFFFFFFF8000000BFB8E318D
    FFFFFF7FFFFFFFFFFFFFFFFFFF7FFFF00FB6D7C
    DFFFFFFFFA00000003B8E318DFFFFFFFFFFFFFF
    FFFFFFFFFF
260 DATA F7FFFF40FB897CDFFBFFFFFA640000003
    8E318C0000FFFFFFFFFFFFFFFFFFFEFFFFF50
    FB097CDFFFFFFFFA667DFFFFF8E318D7FFEFFFF
    FFFFFFFFFFFFFFFFFEFFFFF50BF017CDF80003F
    FBE66533323FFFFFD7FFEFFBFFFFFFFFFFFFFF
    FFFF7FFFF50
270 DATA BF0D78DF9FFFBFFBFE6533323FFFFFD76
    46FFFFFFFFFFFFFFFFFFFFF7FFFF52BF1D78DF
    9FFFBFFB3FE5333238E318D7FFEFFF1FFFFFFFF
    FFFFFFFFFFBFFFF50BF0D78DF9FFFBFFB33FDFF
    FFB8E318D7FFEFFFEFFFFFFFFFFFFFFFFFFFFF
    F42BF9778DF
280 DATA 9FFFBFFB333D333238E318D4836FFFF7F
    FFFFFFFFFFFFFFFF9FFFF52BFE378DF9FFFBFFB
    F335333238E318D7FFEFFFF7FFFFFFFFFFFFFFF
    FE7FFFF62F7C178DF9FFFBFFBFF3533323FFFFF
    D7FFEFFFF7FFFFFFFFFFFFFFFF9FFFFF5637C37
    8DFBFFFBFFB
290 DATA 3FF533327FFFFFD7FFEFFFFFFFFFFFFFF
    1FFFFFFBFFFFF16BFE378DFAFFFBFFB33FDFFFF
    B8E318D7FFEFFFE7FFFFFFFF003FFFFFFFFFFF1
    FB7E378DFAFFFBFFB333D333238E318D7F62FFF
    EFFFFFFFC080FFFFFFFFFFF5767E778DFAFFFB
    FFBF3353332
300 DATA 38E318D7642FFFFFFFFFFF00107FFFFF
    FFFFF5767C378DFAFFFBFFBFF35333238E318D7
    FFEFFFDFFFFFFFC01401FFFFFFFFFFFF777E378D
    FAFFFBFFB3FF533323FFFFFD6C3AFFFFFFFFFFF
    802102FFFFFFFFFF67F7E178DFAFFFBFFB33FDF
    FFFBFFFFFD7
310 DATA FFAFFFE1FFFFFF0826807FFFFFFFFF477
    7F378DFAFFFBFFB333D333238E318D7FFEFFFBE
    FFFFFF1560D03FFFFFFFFF0F7FF378DFAFFFBFF
    BF335333238E318D7FFAFFFFF7FFFFE2569503F
    FFFFFFFF0F77AB78DFAFFFB003FF35333238E31
    8D7ECEFFFFF
320 DATA 73FFFC6C7F521FFFFFC0FF6F278D78DFA
    FFFB0033FF5333238E318D7FFEFFFFFFFFFF9DE
    7F940FFFFFD6FF4F278B78DFAFFFB5FB33FD333
    23FFFFFD7FF8FFFFFFEFFFBFD7FD7EFFFFFD6FF
    7FBFBB78DFAFFFB5FB333DFFFFBFFFFFD587EFF
    FFFFF7FF280
330 DATA 510007FFFFD6FF7FFF9178DFAFFFB5FBF
    335333238E318D7FFEFFFFFFFBFE7FFFFFF23FF
    8001005FF79378DFAFFFB5FBFF35333238E318D
    7FFEFFFC7FFDFCFFFFFFFF1FF7FFFFF5FF79378
    DFAFFFB5FB3FF5333238E318D7FFEFFE3FFFDF1
    FFFFFFFFC7F
340 DATA 7FFFFF1FFFB378DFAFFFB5FB33FD33323
    8E318D6C9EFF9BFFFEE7F0000107F7F7AEBAF5F
    F7B778DFAFFFB5FB333DFFFFBFFFFFD7FFEFFFF
    FFFEE7E4000001F7F7BEFBF5FBF9778DFAFFFB5
    FBF33533323FFFFFD7FFEFFFFFFFDE7801F5FF8
    77F7BEFBF6F
350 DATA BFB378DFAFFFB5FBFF35333238E318D30
    68FFFFFFFDE01F8C003F0FF7FFFFF6FFFB178DF
    AFFDB5FB3FF5333238E318D7FFEFFFFFFFDFA78
    70FFCE0FF0000002FFFB378DFAFFDB5FB33FD33
    3238E318D7FFEFFFFFFFBF827E000F07FF00001
    501BF9178DF
360 DATA AFFDB5FB333DFFFFB8E318D2008FFFFFF
    FBFF181EFD00FFF00000B01BF1178DFAFFDB5FB
    F33533323FFFFFD7FFEFFFFFFFFFF86F6FDFCFF
```

```
   F00000723BB9374DFAFFDB5FBFF3533323FFFFF
   D7FFEFFFC001FFFF6F6F5B3FFF00001F23BFB36
   4DFAFFDB5FB
370 DATA 3FF5333238E318C41BEFF807C03FFF6B6
   F5B7FFF00003F23379164DFAFFDB5FB33FD3332
   38E318D7FFEFF1FFFF1FFF6B6F5B7FFF5DC0002
   1331164DFAFFDB5FB333DFFFFB8E318D7FFEFF7
   FFFEDFFF6B6F5B7FFF5DC8FF25BB1164DFAFFDB
   5FBF3353332
380 DATA 38E318D7FFEFF7FFF21FFF6B6FDB7FFF5
   DD2FE217BB164DFAFFDB5FBFF3533323FFFFFD7
   66EFF788221FFF6B7FDB7FFF5DC400047F8164D
   FAFFDB5FB3FF533323FFFFFD7FFEFF7EFF65FFF
   6BFFDB7FFF40007F247B8164DFAFFDB5FB33FD3
   33238E318D7
390 DATA FFEFF7FFF65E0000FFFB7FFF5DC03F057
   BC164DFAFFDB7FB333DFFFFB8E318D7B32FF643
   D65EFFFEFFFB7FFF5DC1002D7BA164DFAFFCF7F
   BF335333238E318D7A32FF000561EFFFEFFFB7F
   FF5DC21F255B2164DFAFFB77FBFF35333238E31
   8D7FFEFF6CF
400 DATA D69E90FE00007FFF5DC0AF26DB0164DFA
   FFB77FB3FF533323FFFFFD7FFAC06CFF61EFFF8
   00007FFF504100267FA164DFAFF7B7FB33FD333
   23FFFFFD6C9ACD6CFD09EF0F0FFFE7FFF5DC00F
   275FC564DFAFF7B7FB333DFFFFB8E318D7FFEC9
   106D49EFFF2
410 DATA 00037FFF5DC1AF017E8964DFAFF7B7FBF
   335333238E318D7FFEC96D7009EFFF64FFF7FFF
   5DC347087F896CDFAFF7B7FBFF35333238E318D
   766EC96C6029EFEF1CFFF7FFF5DC340187FC96C
   DFAFEFD7FB3FF5333238E318D7FFEC9612641E1
   EF30FFF7FFF
420 DATA 41036B187ECD6CDFAFE817FB33FD33323
   FFFFFD7FFEC9696465EF0F5CFFF7FFF5D836B10
   FE816CDFAFE057FB333DFFFFBFFFFFD4836C961
   2465EFEF3CFCF7FFF5D8375303E856CDFAFC143
   FBF335333238E318D4836C96D2C25EFEF7CFFF7
   FFF5D83700B
430 DATA BF8D6CDFBF1100FBFF35333238E318D7F
   FEC9693405EE0F7CFDF7FFF5DBB7E09BECD6CDF
   BE11007B3FF5333238E318D7FFEC96DA405EFEF
   70FDF7FFF5000000ABE8D6CDFBC01103B33FD33
   3238E318D7FFEC96DA08DED1D4CFDF7FFF5DBB7
   F02BE8D6CDF
440 DATA B801849B333DFFFFBFFFFFD7B22C96DA4
   8DEFDD3CFDF7FFF5DBB7F083A896CDFB009820B
   F33533323FFFFFD7B22C96DA08DEFDD7CFDF7FF
   F5DBB6F003A816CDFA090CA8BFF35333238E318
   D7FFEC96DA589EFDD70FBF7FFF500007083A856
   CDFA032C903
450 DATA 3FF5333238E318D641AC96DA589EE5D6C
   FBF7FFF5DA00709BAC95CDF8252C48333FD3332
   38E318D641AC96DEDC9EFDD5CFBF7FFF5D91B31
   820495CDF80AAED43333DFFFFB8E318D7FFEC97
   DEDC9EFDD7CFBF7FFF5D097D1080495CDF852AA
   5C3F3353332
460 DATA 3FFFFFD7FFEC97DEDC9EFFD7CFBF7FFF0
   43EFD0800015CDF88EAE2C3FF3533323FFFFFD5
   006C9FDEDC9EF0D60FFF7FFF54BF3E0000015CD
   F89EEEF633FF5333238E318D7FFECAFDEC59EFD
   D08FFF7FFF1400014008015CDF97DEF76327FD3
   33238E318D7
470 DATA FFECB6DEC69E81D64FFF7FFF57F87F503
   0015CDFAFDEF7B3267DFFFFB8E318D0016CB6DE
   849EFD95CFFF7FFF67E13F4000015CDFAFDEFFB
   BE665333238E318D7FFECB6DE849EFD97CFFF7F
   FF07C25F1C20415CDF9FFFFFBBFE6533323FFFF
   FD7FFEC36DE
480 DATA 9C9EFD978FFF7FFF7FC02F1481055CDF8
   88FF8233FE533323FFFFFD7FFEC26DE9C9EFD96
   4FFF7FFF00000715C1015CDFBFC110FB27FD333
   238E318D7FFEC26D2AC9EFFD1CFEF7FFF000001
   1751055CDF807FFFC3267DFFFFB8E318D7646C2
   6DE8C9EDFD7
490 DATA CFEF7FFF7FFFFF008B015CDF9FC0003BE
   665333238E318D7FFEC27C0889ED7D64FEF7F9F
   0000000103415CDF9FD7F5FBFE65333238E318D
   7FFEC6FDE889ED7D1CFEF7E6F7C7F3700353D5C
   DF9CD7359B3FE533323FFFFFD4836C6FDE8C9ED
   757CFEF7EEF
500 DATA 739F360077855CDF9856150B27FD33323
   FFFFFD7FFEC47DEB09EF757CFEF7EF70FFC4320
   22015CDF9A56950B267DFFFFB8E318D7FFEC9FD
   EE09EF7570FEF7EFB4FEB750202015CDF9CD735
   9BE665333238E318D7FFECAFDF809EF714CFEF7
   DF93FFEF800
510 DATA 0A215CDF9FD7FDFBFE65333238E318D7F
   FECAFDF829EF717CFEF7DFB3FEEF90203015CDF
   98D61DAB3FE5333238E318D7B22CAFDF829EF71
   7CFEF7DF34FFC391225215CDF98D61D8B27FD33
   323FFFFFD7B22D2FFFA8DEF757CFEF7EF97FFF3
   C1227415CDF
520 DATA 98D61D8F267DFFFFBFFFFFD7FFED2C000
   00000004FEF2CB57FFF591035015CF118D61D8B
   E665333338E318D641AC25DB86EED3774FEF783
   537FF97002B015C0AA8D61D8B7C65333338E318
   D641AC25DBB6EEDB774FEF77715FFC38601C010
   07E10D6DD80
530 DATA 01E5333338E318D7FFEC2400000000004
   FEF4F773FE28800240077FEF5D6DD9A21FD3333
   38E31857FFEC25DBB6EEDB774FEF3FB10000800
   0A9380FFEE846DD70247D33333FFFFFB7E46C25
   DBB6EEDB774FEF3FFD3FFF07C00003FFFEEA421
   4C32465FFFF
540 DATA B8E31F80FFEC2400000000004FEF5FE38
   0002FFFFFFFFFEEB7A000F3465333238E31FDF
   3FEC25DBB6EEDB774FEF439FFFFFAFFFFFFFE0
   02B7FDFEE35E5333238E31D1F836C25DBB6EEDB
   774E2F586BF3FFAFFFFFE007FEB7BDFEE24003
   33238E318FF
550 DATA DFECA5DBB6EEDB7749D3706BCDFFAFFFB
   0000000100000000006EC001E7E002FFE0000000
   0000000007FB6EE3DDFFAFF006DBB6EDB000000
   0066EEBD418000FFFEC000000000000000FFD1F
   FEDEFFAF05B6DBB6ED9FFC0003FE6EEB55079FE
   7FFF7FFC000
560 DATA 000000000FFD7FFEDF7FACB5B6DBB6ED8
   00000000004EB5550E79FFFF181FFFFE6FE3FDF
   FFF37FFBBF3FAFB5B6CB0000000084003006000
   554F9BFFFFC7E1F8019FDFFEFFFEF3FC0BF7FB1
   B5B60000000100040020058005573 1BFFFFC7FD
   F2E67FFFFEF
570 DATA FF9F801FBE7F8E350018000909A426613
   3340075578E1FFEE387ECF9CFFFFF9FAD8018FF
   DF3C79808381861248492079039 99C040158E3F
```

```
    FCFF7E3F99FFFFFFFC03FFFFF96A3867C040326
    9FFFFFFFFFFFFFFE0582031C7F5FFBEFF67FFFF
    CC03FFFFFFF
580 DATA 869F39E700001FE401FC0000000041FC4
    03CE70ABFFB3FE9FFFD402FFFFFFFFFCFA4C79F
    98DBC3FFFFFFFFFFFFFFFC0001CF38C3FFE67FE
    DFFF3FFFFFFFFFFFF80333E7E79000000001FFF
    FFFC00000233F7CF0BFFDCFFF3F80FFFFFFFFFF
    FC0E78C39F1
590 DATA E7CF80000000000000002C795CF9F0B8F
    F3DFFF387FFFFFFFFF0800808387CF9E3E7F7B
    F780000400F63F0FE71E71E383FBFFFB3FFFFFF
    F0000C01FF3D0F13E7DF9FCEFEFEFBFDF3CFDC7
    C2F9E78F83FBFBFFF87FFFFFF1FF802FFFFBB21
    E39F7E7F3DF
600 DATA BFDFBFDF9E7EF8FC3E787C17FFFBFFFE3
    FFFFF0FB07FF0FFF3A6C3C3CF9FCFBF7FDFBFEF
    DFBE3F1F8F99E09BFFFBFFFF9FFFFCF80FFFE22
    FF725D8F8BF7F3F7EFFBFBFF7EFCF9FC7FAA706
    CBFFFBFFFFE7FFC387FFFFCFCF9B4DBB0F0104F
    CF9FF7FBFFB
610 DATA E7E7E7F480383749FFFBFFFFF8FC3C3FF
    FFF3FE7ECC9B7B1FFFC03F3FEFFBFFBF3F28001
    7FC1B965FFF9FFFFFE73C0FFFFFF7FE3EC9A377
    C0807F000040880000003FFFFC00BDB66FFFDFF
    FFFF9E37FFFFFEFFFDECB3477DFBF80FFFFFFFF
    FFFFFFE0002
620 DATA 3DFDD1A37FFCFFFFFFC9FFFFFFFEFFFFE
    5F478FBFBE7E000202000400201FDFF7EFDC793
    7FFD7FFFFFF0FFFFFFFFFFFFF76C9E0BF3EFEFE
    FF7FFBFFEFF7CFEFFBF7E3859BFFC7FFFFFFF1F
    FFFFFFFFFFFAEDE1F0F7CFDFDFEFFFBFFEFF7EF
    FBFCF31E6E9
630 DATA 3FFEBFFFFFFFE3FFFFFFFFFFFCBDEC3F0
    7DFDFDFEFFFBFFF7FBE7FDFF78697767FFF1FFF
    FFFFFC3FFFFE7FFFFF4BDBC3E81FDFDFEFFFBFF
    F7FBF7FC00031BBBDFFFF0FFFFFFFFFC7FFFF8F
    FFBFF3DBEC7F83BFBFDFFFBFFFBFDFA03FFF8FB
    DA3FFFC87FF
640 DATA FFFFFFF8FFFE13FFCFF8B7DFD1BC0FBFD
    FFFBFFFBFC09FE0046FDDCFF86BC1FFFFFFFFFF
    0FF1FFFFFFFCB7DFBE07F041DFFFFFFFC01FF12
    FFBF7DE9FEFFFF07FFFFFFFFFF20FFFFFFCFF0F
    BFBF790FFD080000421FFC0FF7FBF7EF7D5FFFF
    83FFFFFFFFF
650 DATA FFFFFFFFFFFFE7BF7EFF7007BFFFFFFFF
    FEBEFF7FDFBE4FFFFBFFC0FFFFFFFFFFFFFFFFF
    FFFFF87F7EFF7F60089000420017F7FBFDFB0BF
    FFFC7FE03FFFFFFFFFFFFFFFFFFFFF40FDFE7F
    7FBFFFBFFFEFFBF7FBFEFC7FFFFFFCFF80FFFFF
    FFFFFFFFFFF
660 DATA FF87FFFF0DFEFF7FBFFFBFFFEFFBFBFDF
    E81FFFFFFFD1FE03FFFFFFFFFFFFFFFFFF83CFB
    F0FEFF7FBFFFBFFFF7FDFBF8003FFFC03FFC61F
    10FFFFFFFFFFFFFFFFFFFFF0DFF0EFEFF7FFFBF
    FFF7FDFDE7FFF8FFFFFFE31DFCC3FFFFFFFFFFF
    FFFFFFFFF0
670 DATA FFF1FEFF7FFFBFFFF7FEFC1FFE07FFFFF
    FDFFEFE70FFFFFFFFFFFFFFFFFFFFE0FFE0081
    S0200082001000FC11FFFFC1FFF7EFF3C3FFFF
    FFFFFFFFFFFFFFFFFF03FFFFFFFFFFFFFFFFFF
    83FFFFFC03FFF8F91FC70FFFFFFFFFFFFFFFFF
    FFFFFC3FFFF
680 DATA FFFFFFFFFFFF807FFFFF83FFFFC774E18
    3C7FFFFFFFFFFFFFFFFFFFFFFF80080FFFFFFFF
    F8007FFFFFFFFFFFFFFFFFFDB371FFFFFFFFFFF
    FFFFFFFFFFFFFFFFF000000FE03FFFFFFFFFFF
    FFFFFFFFFEB8FCFFFFFFFFFFFFFFFFFFFFFFFF
    FFFFFFFFC81
690 DATA FFFFFFFFFFFFFFFFFFFFFFFEBFFFF7023
    DCE037FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
    FFFFFFFFFFFFFFF97FFFF1C63C46C71FFFFFFFF
    FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
    FFFFFFE8C63A02C68FFFFFFFFFFFFFFFFFFFFFF
    FFFFFFFFFFF
700 DATA FFFFFFFFFFFFFFFFFFFFECC63B10C6CFF
    FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
    FFFFFFFFFFFFDCC63718C5CFFFFFFFFFFFFFFFF
    FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFC0
    C6301CC40FFFFFFFFFFFFFFFFFFFFFFFFFFFFF
    FFFFFFFFFFF
710 DATA FFFFFFFFFFFFDCC6071EC5CFFFFFFFFFF
    FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
    FFFFFFFFFFFFFFFF00265C000A8722544849532
    049532041204D414348494E45204C414E475541
    474520434F4E5645525349 4F4E2050524F47524
```

BULLETIN BOARDS COMPUTER CLUBS & PEN PALS

If you want a free listing send us the information. These listings will be kept current.

BULLETIN BOARDS

BBS The Dungeon BBS of Newport NC Supporting IBM, Commodore, CoCo, Apple, Atari, and Tandy Computers. On-Line Special Interest Groups, Forms, Multi Message Bases, Upload and Download areas. On-line shopping and Business services. 300/1200 baud 24 hours. 919-726-9737.
Sysop: Chuck Katsekes
Address: 410 Scott Dr
Newport, NC 28570
Telephone (voice) 919-726-0018

****

Dungeon Bulletin Board System
North Carolina 919-726-9737
Chuck Katsekes
410 Scott Dr
Newport, NC 28570

****

FAT NEWSLETTER 300/1200 Baud
FAT 1010
Concord Av
Piqua, OH 45356

(513) 778-9624

****

COLOR COMPUTER CLUBS

Jan Colucci Editor C'Crier

The Color Computer Club
PO Box 478
Canfield, OH 44406

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.

We did not receive any new product information this month.

* * *

If you have not written I would like to hear from you. All of our letters were from ham radio operators. If you are having a problem maybe we can help. If not we will ask our reader's for their help. All replies to letters will be printed in our Question & Answer column. - Bill

ADVERTISER'S INDEX

We would appreciate it if you would let these advertisers know that you saw their advertisement in Dynamic Color News

DYNAMIC COLOR NEWS
CUMMULATIVE INDEX

Editor's Comments, Questions and Answers, and Operating Hints are included in all issues. Product reviews are in issues since #17. Numbers in Parentheses indicate the part of a series.

#1 Feb 84
Basic Programming
Programming Theory
Multiprogram Manager
MPM Program Listing
Machine Lang. Prog.

#2 Mar 84
Basic Programming
Memory Expansion
Machine Lang. Prog.

#3 Apr 84
ASCII Part 1
Basic Programming
Machine Lang. Prog.

#4 May 84
ASCII Part 2
ML Programming-
Interrupts
Inatalling an
Interrupt switch
Utility Program

#5 June/July 84
Powerful Remarks
(Remarks for data)
ASCII Part 3
Uninterrupted Power
Machine Lang. Prog.

#6 Aug 84
Powerful Remarks (2)
Data in Remarks
Check Book Program
Word Proceasor
Machine Lang. Prog.

#7 Sept 84
Machine Lang. Prog.
Writing Position
Independent Code
Powerful Remarks (3)

#8 Oct 84
Basic Programming
(Data Handling Tech)
Machine Lang. Prog.
(Memory Searching)

#9 Nov 84
Computer Sound (1)
Basic Programming
(Sorting)
Ball Team Sort Pgm

#10 Dec 84
Random Numbers
Computer Sound (2)
Sound Learning Pgm
Sound Demo Program

#11 Jan 85
Random Numbers (2)
Card Shuffling
Computer Sounds (3)
Machine Language Subs
Sound Program
Discussion

#12 Feb 85
Large Memory Pgms (1)
Computer Graphics (1)
Video Reverser
(hardware)

#13 Mar 85
Large Memory Pgms (2)
Computer Graphics (2)
Writing Programs (1)

#14 Apr 85
Large Memory Pgms (3)
Computer Graphics (3)
Graphics Demo Program
Writing Programs (2)
Print Demo Pgm

#15 May 85
Writing Programs (3)
Gas Mileage Program
Large Memory Pgms (4)
Data Move Program
Computer Graphics (4)
Graphics Demo Program

#16 June 85
Writing Programs (4)
Grade Book Program
Computer Graphics (5)
Character Gen. Pgm
Large Memory Pgms (5)
Address File Program

#17 July 85
Large Memory Pgms (6)
64K RAM Program
Writing Programs (5)
Alarm Clock Program
Computer Graphics (6)
Character Gen. Pgm.
CoCo Heat Problem

#18 Aug 85
Writing Programs (6)
Address File Program
Large Memory Pgms (7)
Study Program
Computer Graphics (7)
Line Demo Program

#19 Sept 85
Writing Programs (7)
Fast Food Program
Computer Graphics (8)
Bar Graph Program
Vector Correction Pgm
Large Memory Pgms (8)

#20 Oct 85
Writing Programs (8)
Word Processor Pgm
Bar Graph with
Character Generator
Second Port
Page -1 for 32K + MEM
Large Memory Pgms (9)

#21 Nov/Dec 85
Writing Programs (9)
Check Book Program
Computer Graphics(10)
Circle Demo PGM
Recipe Program
RAM Disk Program
Electric Cost Program

#22 Jan 86
Writing Programs (10)
Inventory Program
Computer Graphics(11)
ARC & Circle Demo PGM
Large Memory Pgms(11)
Ship War (Game)

#23 Feb 86
Writing Programs (11)
File DEMO Program
Basic Basic (1)
Computer Graphics(12)
Draw Demo Program
Interfacing Comp. (1)
Bouncing Ball (Game)

#24 Mar 86
Interfacing Comp. (2)
Electronic Billboard
Writing Programs (12)
Basic Basic (2)
Computer Graphics(13)
Draw Program
(Scalling)
Large Memory Pgms(12)
Ramdisk Subroutines

#25 Apr 86
Interfacing Comp. (3)
Writing Programs (13)
Basic Basic (3)
Tanks (Game)
Large Memory Pgms(13)
Upper Memory Program
Computer Graphics(14)
Graphics Programming
(GET & PUT)

#26 May 86
Writing Programs (14)
ML Programming (1)
Interfacing Comp. (3)
Roulette (Game)
Page -1 Pgm Dev.
Basic Program Restore
Large Memory Pgms (F)
Computer Graphics(14)
Graphics Draw Program

#27 June 86
ML Programming (2)
ML Program (Addition)
Page -1
Mem. Peek & Poke Pgm
Writing Programs (15)
Inventory Program
Interfacing Comp. (5)
Chords (Music Pgm)
Computer Graphics(16)
Graphics Draw Program

#28 July 86
ML Programming (3)
ML Add with Carry
Graphics Zoom Program
Writing Programs (16)
File Program
Interfacing Comp. (6)
ASCII Demo Program
Astro Dodge Game
Computer Graphics (F)

#29 Aug 86
ML Programming (4)
ML Subtract Program
Interfacing Comp. (7)
Organize VCR Tapes
Ham Radio & Comp. (1)
Morse Code Program
Disk Disassembler
Basic Prog (17)

#30 Sept 86
ML Programming (5)
ML Data Move Program
Disk File Utility
Basic Programing
File Program
Ham Radio & Comp. (2)
Antenna Design Pgm
Interfacing Comp. (8)
ML Output Subroutines

#31 Oct 86
Money Chase (Game)
ML Programming (6)
Multiple Choice Test
Basic Programming
Address File Pgm (1)
Introduction to OS-9
Interfacing Comp. (9)
Ham Radio & Comp. (3)

#32 Nov 86
Star Constellations
ML Programming (7)
CoCo 3 (1)
Basic Programming
Address File (2)
Duelling Cannons
Ham Radio & Comp. (4)
DX Program (Ham)
Interfacing Comp.(10)
Hardware Interface

#33 Dec 86
ML Programming (8)
CoCo 3 (2)
OS-9
Lucky Money (Game)
Interfacing Comp.(11)
DYTERM Terminal Pgm
Basic Programming
(Sorting)
Ham Radio & Comp. (5)

#34 Jan 87
Cassette Control Sw.
ML Programming (9)
Jungle Adventure
Interfacing Comp.(12)
Ham Radio & Comp. (6)
Morse Code Keyer Pgm
CoCo 3 (3)
Basic Programming
Address File & Sort

#35 Feb 87
ML Programming (10)
Interfacing Computers
(Joystick Voltmeter)
Ham Radio & Comp. (7)
Scrolling Around
Basic Programming
Address File

#36 Mar 87
OWARE (Game)
Joystick Ohmmeter
EPROMS (1)
ML Programming (11)
CoCo 3 (Lowercase
Char & Rev.Video)
Ham Radio & Comp. (8)
Basic Programming
Invoice Program

#37 Apr 87
Diver (Game)
EPROMS (2)
ML Programming (12)
Using Joystick Port
for Measuring Temp.
Ham Radio & Comp. (9)
CoCo 3 (Error Trap)
Basic Programming

#38 May 87
Joystick Digital
Thermometer
Accounts Payable
(Business Pgm)
EPROMS (3)
ML Programming (13)
CoCo 3 Memory Manager
Ham Radio & Comp.(10)
Basic Programming
(Fast Sorting)

#39 June/July 87
Geneology Program
ML Programming (14)
Measuring Light with
Joystick Ports
EPROMS (4)
CoCo 3 Graphics Demo
Ham Radio & Comp.(11)
Calendar Program

#40 Aug 87
Job Costing Program
ML Programming (15)
Basic Programming
Ham Radio & Comp.(12)
Compound Interest Pgm
CC-3 Hi-Res Graphics
Save Program
Dog Race Program

#41 Sep 87
Astro Dodge Game
ML Programming (16)
Reformatting Data (1)
Meteors Program
Computer Terminology
Ham Radio & Comp.(13)
Relay Interface
(Hardware Project)

#42 Oct 87
Taking Control (1)
(Basic Programming)
ML Programming (17)
Disk Cataloger Pgm
Reformatting Data (2)
Parachute (Game)
Ham Radio & comp.(14)
HAM RTTY Program

#43 Nov 87
Save the Maiden (G)
Taking Control (2)
ML Programming (18)
Reformatting Data (2)
Music Program
HAM Radio & Comp.(15)
(Packet Radio)

#44 Dec 87
Kingpede (Game)
Taking Control (3)
Printer Utilities
Ham Radio & Comp.(19)
Audio Generator Pgm
Reformatting Data (3)
Screen Dump Program

#45 Jan 88
Living Maze (Game)
ML Programming (20)
Ham Radio & Comp.(17)
Frequency Counter Pgm
Taking Control (4)
Reformatting Data (6)
FANTASY2 Music Pgm

#46 Feb 88
Using ROM Routines
Taking Control (5)
OS-9 (New Series)
BARRACAT (Card Game)
Ham Radio & Comp.(18)
Improved Ham Log Pgm

#47 Mar 88
Using ROM Rout. (2)
Taking Control (6)
OS-9 & Basic 09
Coast-Coast (Game)
Logic (Ed. Game)
Forth Prog. Lang.
ML Loader Pgm.
Ham Radio & Comp.(18)
WEFAX -Weather Pgm.

DYPRINT

Now you can print LARGE signs for special occassions such as birthdays, parties, or yard sales. Even make your own FOR SALE signs when you need to sell that old car or lawnmower. BANNER uses standard print characters and is compatible with any printer. The characters are formed by a 21 x 27 dot pattern and are printed sideways across the paper. The basic character can be expanded up to 4 times for making large characters up to a full page.

MAXPRINT allows graphics to be blown up and printed on a standard printer. Any PMODE 4 picture can be printed. The program supports all 8 graphics pages for a total of 12288 bytes. MAXPRINT prints 8 characters per byte for a total of 98304 characters. Blow up pictures of friends and family generated by the DS-69B or other digitizer or make posters announcing sales or special events.

The DYPRINT package contains both BANNER and MAXPRINT. The cost is only $19.95

COLOR COMPUTER 2 KIT
(SPECIAL PURCHASE)

Now you can build your own Color Computer 2. These kits were designed for a school and are complete with a step by step instruction manual plus the normal Radio Shack operating manuals. They use 4164 memory chips and sockets are included for all integrated circuits. If you have an older CC-1 or CC-2 then this is an excellent source for spare parts. Replacement parts would cost more than this kit. CC-2 Kit $59.95.

SOFTWARE
Available on Tape or Disk

TERMINAL PROGRAM

DYTERM -Allows a Color Computer to interface with Modems, Terminals, or other Computers using the ASCII port. 300- 2400 baud, 1 or 2 Stop bits, 7 or 8 bit words, variable parity. $9.95

COCOMAX II
(Closeout)

The best graphics program for the Color Computer 2. Draw a picture, label it, rotate it, copy it, and shrink it. Then print it on a graphics printer. Needs a "Y" cable or multipack expander for disk version.

COCOMAX II disk version	$59.95
Y cable	24.95

DS-69B DIGITIZER

Capture pictures from your VCR or video camera. Display them on the COCO 3's high resolution screen. Label them with COCO MAX and print them on a graphics printer or save them on disk. 256 x 256 resolution, 64 levels of grey, & 8 images per second. Plug in ROM pack requires a multipack expander. Works with all color computer disk systems. DS-69B $149.95.

CC-THERM 2

CC-THERM 2 is a dual digital thermometer for Radio Shack Color Computers. It consists of two thermistors wired to the end of 10' and 20' flat cables for measuring inside and outside temperatures. The other end of the cable is wired to a joystick plug. The thermistors can be mounted on a wall, inside equipment, or outside for temperature measurements. Basic software on tape or disk continuously prints the temperature in both Fahrenheit and Centigrade. T or D software. $19.95

CC-LT (new)

Now you can measure both temperature and light. The joystick assembly includes a light and temperature sensor at the end of a 20' flat cable. Uses one joystick plug. T or D Software 19.95.

128K MEMORIES For D, E, F, 285, or CC-2
(Memory Manager Software Included)

MR-10A - Upgrade CoCo-2 Computers with two 4464 chips to 128K. Specify T or D Software. $49.95.

MR-12 - Upgrade 8-chip 4164 type 64K computers to 128K. Specify T or D Software. $49.95.

DECIMAL ML ASSEMBLER

DISASM is a 6809 Assembler-Disassembler that allows machine codes to be assembled using English mnemonics & decimal arithmetic. It supports all 6809 codes and is especially useful for beginners. $9.95.

VIDEO REVERSER
for the CC-2

Reduces eye strain by producing bright characters on a dark background. Integrated circuit mounts on the 6847 chip. Minor soldering required. $9.95.

MEMORY MANAGER
(for the Color Computer 2)

Did you know that the 64K Color Computer 2 and earlier computers have an extra 32K that is generally not used? Our Memory Manager allows basic or machine language programs to be run in either 32K bank. Banks are exchanged with an EXEC command. Also the second bank can be used as a ramdisk to store programs. This makes cassette operation faster than a disk. A third option configures the computer for the all ram mode allowing data or programs to be stored in the upper memory. The Memory Manager software is available on either cassette or disk.
$19.95.

MEMORY SAVER II

Have you ever had a power failure or brownout to wipe out your program? The Memory Saver II is a battery backup assembly that prevents loss of programs due to power failures. It mounts under the keyboard and works with all color computers. Consists of gel rechargeable battery, control circuit, & miniature toggle switch. Will power a color computer for up to a couple of hours during a power failure. Price reduced. $39.95

Add $3 S/H Checks, VISA, & MC.

Dynamic Electronics Inc., Box 896, Hartselle, AL 35640