VIPER

VOLUME 1 APRIL 1979 ISSUE 9

VP-550
Super Sound

VP-570
Memory Expander

VP-595
Simple Sound Board

VP-590
Color Display

VP-585
Expansion-Keyboard Interface

VP-700
Tiny BASIC

VP-580
Expansion Keyboard

NEW

RCA COSMAC VIP

VP-575
System Expansion Board

$2.00 MOVED TO: PO Box 1142 * Columbia * MD * 21044 ** AN ARESCO PUBLICATION

EDITORIAL

In the last VIPER, the article on the Lunar Lander program was missing some code (see page 17 for the corrections).

Over a hundred people called to inform us of the error and get the corrections. One person wrote and offered to race us - he thought he could write a routine to do the job before we could mail the answer to him. (Wonder who won that one?)

We do aplogogize for the error and for all the time people spent looking for mis-entered code when it was all our (!) fault. There was a very positive side to it all, however. We realized that a lot of people do read the VIPER within a day or two of its arrival, and that many people do enter the programs we publish, and they enjoy. We asked many of our callers what they are doing with their VIPs, and many of them promised to write articles based on their answers.

Everyone we talked to said they intended to renew their subscriptions - and a number of them gave us a credit card number, to renew at once.

So - we know that even if we foul up on a more-or-less constant basis, you still love us. Thanks.

SUBSCRIPTION RATES, ADVERTISING RATES, & OTHER ESSENTIAL INFORMATION

The VIPER is published ten times per year and mailed to subscribers on the 15th day of each month, except June and December. Single copy price is $2.00 per issue and subscription price is $15.00 per year (includes all ten issues of the current volume). Outside USA subscribers should add $10 per subscription for postage costs.

The VIPER is an ARESCO publication, edited by Terry Laudereau and Rick Simpson. For more information about the VIPER, contact the Editors, VIPER, PO Box 1142, Columbia MD 21044, or call (301) 730-5186 between 8AM and 10 PM Mondays through Saturdays.

VIPER advertising rates are

Classified	$10.00/3 lines
Full page	$85.00
½ page	$45.00
¼ page	$25.00

Ad copy should be submitted in camera-ready format. Advertisers will be charged extra for work we have to do here to prepare copy for publication. Payment must accompany ad order.

VIP is a registered trademark of RCA Corporation. The VIPER is not associated with RCA in any way, and RCA is not responsible for its contents. Readers should not call or write to RCA with questions regarding VIPER material. Direct inquiries to ARESCO at our address.

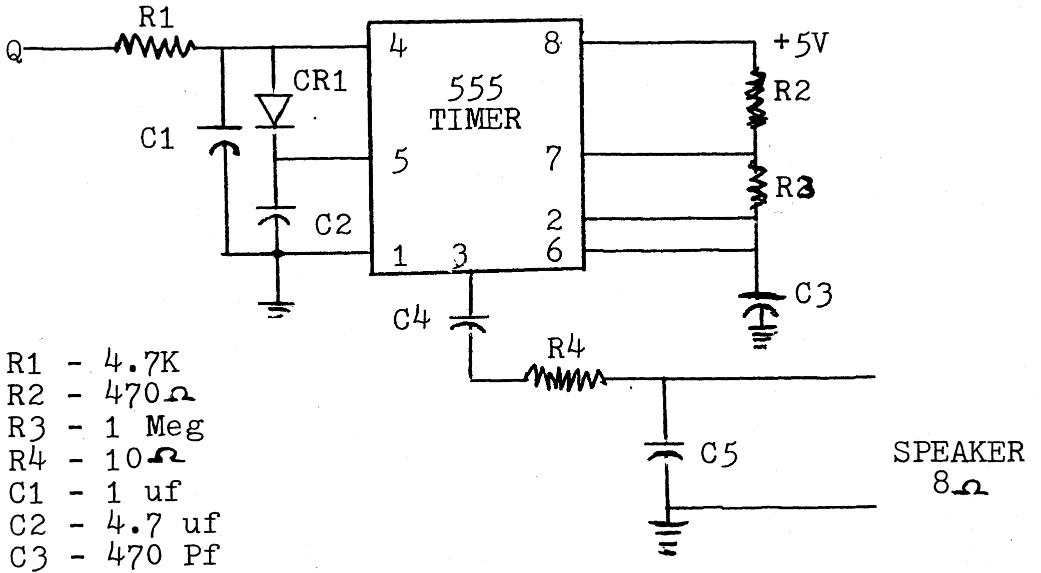
MODIFYING THE ELF-II TO RUN CHIP-8 & VIP GAMES

by Bobby R. Lewis (Part 2)

For a keyboard, I used a small CASIO calculator keyboard, because I like the feel of the keys. Any 16 key matrix will do. I used a 7-strand cable about three feet long, with a 14-pin dip header to plug into my Giant Board.

As a precaution, it is recommended that a +5V regulator be added to the Giant Board in the space provided, to ensure that you don't overload the motherboard regulator.

Many of the VIP games have a "beep" tone built into them, so I've included the tone circuit to be added to Q if you want the tone. See figure 4 for the circuit, and note the modifications to the VIP circuit.



R1	-	4.7K
R2	-	470Ω
R3	-	1 Meg
R4	-	10Ω
C1	-	1 uf
C2	-	4.7 uf
C3	-	470 Pf
C4	-	22 uf
C5	-	1 uf
CR1	-	IN914

Note: Modification of VIP circuit

FIGURE 4 - TONE CIRCUIT

SOFTWARE MODIFICATIONS

First, we'll discuss the changes to be made to the operating system. The OS for the VIP is contained in ROM at addresses 8000 through 81FF. Although this code could be loaded at any page boundary, we'll put it in address 0D00 through 0EFF. You will need to obtain a copy of the VIP manual, or back issues of the VIPER for a complete hexadecimal listing of the OS and the CHIP-8 interpreter. See Table 1 for a summary of instructions that must be changed in the OS to be compatible with the ELF-II. Once you have loaded the OS, with the modifications, put it on tape before you run it, in case you make a mistake while loading it.

ADDRESS	CODE	COMMENTS
8001	0D	;CHANGES PAGE LOCATION OF CODE
800A	67	;OUTPUT PORT ENABLE FOR KEYPAD
8022	37	;EF4 INSTEAD OF EF3
8056	0E	;CHANGES PAGE LOCATION OF CODE
8104	3F	;EF4 INSTEAD OF EF3
819C	67	;OUTPUT PORT ENABLE FOR KEYPAD
819F	3F	;EF4 INSTEAD OF EF3
81AA	37	;EF4 INSTEAD OF EF3

If you want to use EF3 for your keyboard, don't change the values at locations 8022, 8104, 819F or 81AA. High addresses 80 and 81 will become 0D and 0E when you load the code into RAM, starting at address 00D0.

TABLE 1 - OPERATING SYSTEM MODIFICATIONS

Now you can try the operating system by inserting a long branch (C0 0D 00) at address 0000. When you flip to RUN, while keeping input depressed if you are using EF4, you will see some random bit pattern on the top of the screen and an operating stack toward the bottom of the screen when you release the input switch. Press INPUT four times, and and you should see some address appear at the bottom left, and the contents of that address at the bottom right. (All this assumes you have either not made the keyboard mods <u>or</u> that you have made the mods <u>and</u> selected EF4 as your keyboard flag.) If you have made the keyboard modifications, follow the instructions in the VIP manual for using the OS. If you haven't made the modifications, this is as far as you can go with the operating system, although you can still use CHIP-8 and some of the VIP games.

Once you're sure the OS is working correctly, you can then <u>modify</u> and load the CHIP-8 interpreter at addresses 0000 through 01FF. Table 2 contains the modifications that <u>must</u> be made to the CHIP-8 interpreter before using it on your ELF-II.

ADDRESS	CODE
0000	00 00 00 00 00 00
000A	0E
010B	0E
012A	0E
019A	67
019E	37
01A1	3F

Locations 0000-0002 will now be used for a long branch to the CHIP-8 patch at 0CF0, VIP OS at 0D00, or to the ELF II monitor at F000. Locations 0003-0005 are not used.

TABLE 2 - CHIP-8 INTERPRETER MODIFICATIONS

Because the CHIP-8 interpreter normally expects R1.1 to be initialized to the display page when RUN is flipped on, we have to add an additional patch at 0CF0 (or anywhere you like - see Table 3). This patch must be the first code executed when you run a CHIP-8 program. The original code in the CHIP-8 interpreter at addresses 0000-0005 is not used.

```
ADDRESS    CODE

  0CF0     F807    ;Put 07 in RB.1 for CHIP-8 display page
  0CF2     BB
  0CF3     FF01    ;Subtract 1 for CHIP-8 variable storage area
  0CF5     B2B6    ;Put 06 in R2.1 and R6.1
  0CF7     C000    ;Long branch back to address 0006 to start
  0CF9     06
```

Note: There is no easy way to use the same display page for both the OS and for CHIP-8, so now the OS uses page F and CHIP-8 uses page 7. Using this method, you can display instructions for using the operating system in page F and won't destroy it when you run a CHIP-8 program. Enter the code from Table 5 if you want the instructions displayed on the screen when you bring up the OS. Your name or other info can be inserted at address 0F10 through 0F37 if desired.

TABLE 3 - CHIP-8 PATCH

As you have seen by now, the modifications are very minor. We've only changed the I/O instruction code, page address code, and EF flag instructions. You can experiment and change the software to suit your individual purposes. Refer to table 4 for a memory map of your VIP/ELF-II code. You will probably want to relocate some code if you have very large CHIP-8 programs to run. All programs written in CHIP-8 with no machine language subroutines should work on your ELF-II. You will probably have to modify VIP machine language programs to run on the ELF-II because of the VIP executing some code in the OS before executing a program at address 0000. At first glance, it may look like the VIP executed machine language programs beginning at 0000, when it really goes to address 8000 and initializes R1.1 before jumping to 0000. Keep this in mind and you should have no problems.

In general, you may now run any CHIP-8 program that only requires a single key depression without making the keyboard modifications. If you have made the keyboard mods, you're ready to run any VIP game written in the CHIP-8 language.

PAGE	USE
0	CHIP-8 interpreter
1	"
2	CHIP-8 programs
3	"
4	"
5	"
6	CHIP-8 work area (06A0 - 06FF)
7	CHIP-8 display page
8	Available
9	"
A	"
B	"
C	CHIP-8 patch (0CF0 - 0CF9)
D	VIP operating system
E	"
F	VIP OS display page

TABLE 4 - MEMORY MAP

1. RCA COSMAC VIP instruction manual (VIP-300)
2. RCA VIP Users Guide (VIP-320)
3. VIPER (issues 2 and 3, volume 1)
4. CHIP-8 AND VIP are trademarks of RCA
5. GIANT BOARD and ELF-II are trademarks of Netronics R & D Ltd.

ADDRESS	DATA							
0F00	FF	FF	FF	FF	FF	FF	FF	FF
0F10	83	C8	9C	F0	00	00	00	00
0F18	82	08	88	80	00	00	00	00
0F20	83	8A	88	F0	00	00	00	00
0F28	82	0F	88	10	00	00	00	00
0F30	F3	CD	9C	F0	00	00	00	00
0F38	00	00	00	00	00	00	00	00
0F40	D9	E0	0F	02	6C	88	03	C0
0F48	F9	20	09	04	7C	88	02	40
0F50	A9	E5	4F	08	54	A9	52	40
0F58	89	40	09	10	44	F8	02	40
0F60	89	20	09	20	44	D8	03	C0
0F68	00	00	00	00	00	00	00	00
0F70	F9	E0	0F	02	7C	88	03	C0
0F78	21	20	09	04	10	88	02	00
0F80	21	E5	4E	08	10	A9	53	80
0F88	21	40	09	10	10	F8	02	00
0F90	21	20	0F	20	10	D8	02	00

TABLE 5 - OPERATING SYSTEM INSTRUCTION DISPLAY AREA

REVERSE VIDEO DRAWING

by Udo Pernisz

The program listed below is a Reverse Video Drawing Cursor, written in the CHIP-8 Interpreter, to run on the VIP. It has several very nice features:

* The cursor moves horizontally, vertically, and diagonally
* The cursor has an adjustable blinking frequency
* The cursor switches to automatic repeat mode when a key is held down longer than the delay period
* The delay period is adjustable
* The cursor moves on two backgrounds that are selectable under program control
  # Key C clears the screen to black (all zeros) - Direct video mode
  # Key F fills the screen to white (all ones) - Reverse video mode
* There are three selectable cursor modes
  # Key 0 lets the cursor move around without leaving a trace
  # Key D draws a line onto the background
  # Key E erases any line the cursor moves over
* Keys D and E retain their functional designation in both the Direct video mode and the Reverse video mode

Most of the details of the program should be obvious from the program comments included in the listing. However, the user might be interested in the following points:

1. The key-pad scan routine used for both the cursor keys and the mode keys is very general, and can be readily modified to fit the user's personal requirements.
2. The new position of the cursor is computed from the old position, by adding the components of a vector which are extracted from the geometric representation of the pressed key as "row" and "column" on the keypad.
3. The program routine which displays the cursor symbol consists of only seven CHIP-8 statements. These statements are used for all three cursor modes in both Direct and in Reverse Video Mode.
4. One instruction of the CHIP-8 Interpreter (in location 00E5) is changed by the program, according to the selected video mode, to obtain either ERASE TO FULL screen or ERASE TO CLEAR screen.
5. In order to have key D "draw" in both video modes, location 0251 is changed to 028D by the routine at 0284. This part can be left out if program size is essential. If it is omitted, however, keys D and E reverse their meaning when in Reverse Video Mode.

```
ADDRESS   CODE   COMMENTS
                                         INITIALIZATION ROUTINE
  0200    0290   Do MLS at 0290        : Define display page
  0202    A262   I=0262                : Read Variable values
  0204    F465   V0:V4=MI
  0206    6E00   VE=00                 : Set cursor mode to No Effect

                                         SCAN VECTOR KEYS
  0208    A267   I=0267                : Read each key, one at a time
  020A    F065   MI=V0:V0              : within a loop, into V0
  020C    4000   SKIP; V0.NE.00        : At end of scan (V0=0), if no
  020E    125E   GOTO 025E             : key pressed, exit scan loop
                                       : (to set V5=0)
  0210    E09E   SKIP;KEY=V0           : Exit scan loop with value of
  0212    120A   GOTO 020A             : pressed key assigned to V0

                                         DELAY REPEAT
  0214    4500   SKIP; V5.NE.00        : Skip Compute-Vector routine
  0216    121E   GOTO 021E             : if either V5=0 (i.e., key
  0218    3506   SKIP; V5.EQ.06        : pressed for first time after
  021A    1232   GOTO 0232             : at least one scan with no key
                                       : pressed) or if number of delay
                                       : skips has been reached
  021C    75FF   V5=V5+FF              : Subtract 1 from V5 to prevent
                                       : overflow of V5 (see 0232)

                                         COMPUTE VECTOR
  021E    70FF   V0=V0+FF              : V0 is reduced to the row in
  0220    67FF   V7=FF                 : which the pressed key belongs,
  0222    8035   V0=V0-V3              : while V7 counts the column.
  0224    3F01   SKIP; VF.EQ.01        : Thus, V0 and V7 will hold the
  0226    122C   GOTO 022C             : components of the incremental
  0228    7701   V7=V7+01              : vector to be added to the pos-
  022A    1222   GOTO 0222             : ition vector of the cursor
  022C    7002   V0=V0+02
  022E    8104   V1=V1+V0              : Calculate new coordinates of
  0230    8274   V2=V2+V7              : the cursor
  0232    7501   V5=V5+01              : Increment delay counter

                                         SCAN MODE KEYS
  0234    A270   I=0270
  0236    F065   V0:V0=MI
  0238    E0A1   SKIP; KEY.NE.V0       : If a key from the scan list
  023A    8E00   VE=V0                 : is pressed, assign its value
                                       : to VE
  023C    3000   SKIP; V0=00           : Detect end of scan for V0=0
  023E    1236   GOTO 0236
  0240    3E0C   SKIP; VE=0C           : Branch into ERASE routine if
  0242    4E0F   SKIP; VE.NE.0F        : either key C or key F is pressed
  0244    1276   GOTO 0276

                                         SHOW CURSOR
  0246    A262   I=0262                : Cursor symbol
```

```
ADDRESS  CODE  COMMENTS

  0248   D121  SHOW 1MI @ V1,V2
  024A   3E00  SKIP; VE=0       : Decide on overwriting cursor,
  024C   3F00  SKIP; VF=0       : depending on whether the No-
  024E   D121  SHOW 1MI @ V1,V2   Effect key was pressed or the
                                : pixel held a 1 or a 0
  0250   4E0E  SKIP; VE.NE.0E   : Detects Line Erase Mode
  0252   D121  SHOW 1MI @ V1,V2

                                BLINK CURSOR
  0254   F415  TIME=V4          : Set blink period to additional
  0256   F607  V6=TIME          : V4/60 seconds
  0258   3600  SKIP; V6=0
  025A   1256  GOTO 0256
  025C   1208  GOTO 0208        : Go to start of SCAN VECTOR KEYS

                                RESET DELAY
  025E   6500  V5=00            : Set V5 to initial value of 0
  0260   1234  GOTO 0234        : after a scan with no key pressed;
                                : Enter the SCAN MODE KEYS routine,
                                : skipping over the DELAY REPEAT
                                : and COMPUTE VECTOR routines

ADDRESS  CODE                   DATA
  0262   80 00 00 03 01         : Cursor symbol; two cursor co-
                                : ordinates (initially)
  0267   01 02 03 04            : Vector key list.  Last value
         06 07 08 09 00         : set to 0 to exit scan
  0270   0C 0F 0D 0E 00 xx      : Mode key list.  Last value is
                                : scanned and used to exit scan


ADDRESS  CODE  COMMENTS
                                ERASE SCREEN
  0276   6093  V0=93            : Change CHIP-8 to select either
  0278   3E0C  SKIP; VE=0C      : register 3 or register D by in-
  027A   609D  V0=9D            : struction at 00E5, which con-
  027C   A0E5  I=00E5           : tains as its high-order byte
  027E   F055  MI=V0:V0         : either 00 (clear) or FF (full),
                                : respectively
  0280   0296  DO MLS AT 0290   : Define display page
  0282   00E0  ERASE            : Clear or full screen, depending
                                : on value of VE

                                CHANGE DRAW/ERASE LINE
  0284   600E  V0=0E            : Change program to check approp-
  0286   3E0C  SKIP; VE=0C      : riate key value of VE (see 0250)
  0288   600D  V0=0D
  028A   A251  I=0251
```

```
ADDRESS  CODE   COMMENTS
  028C   F055   MI=V0:V0
  028E   1202   GOTO 0202          : Start over at beginning

                                   MACHINE LANGUAGE SUBROUTINE

  0290   01       LDN
  0291   F8 03    LDI 03           : Defines page 3 as display
  0293   BB       PHI reg B        : page
  0294   E2 D4    SEX SEP

  0296   F8 FF    LDI FF           : Puts FF into high-order byte
  0298   BD       PHI reg D        : of register D to provide all
  0299   D4       SEP              : ones (white screen for Reverse
                                   : video mode)
```

Functions:

C	Erase to clear screen and enter Direct (white on black) Video Mode
F	Erase to full screen and enter Reverse (black on white) Video Mode
D	Draw lines
E	Erase lines
0	No Effect; move cursor without affecting display

Program language: CHIP-8 Interpreter, 1802 machine code
Program location: 0200 through 0299
Program features: Cursor draws and erases in both direct (white-on-black) and reverse (black-on-white) video modes.

Automatically repeating keys, with adjustable delay before repeat begins

Adjustable cursor blinking rate

Horizontal, vertical, and diagonal cursor movements

A READER REQUESTS HELP FROM VIPERS

Stephen Place would like to have any information and documentation on SWTP's CT1024 terminal - anything which would help the owner of a blown-up unit (Steve's the owner) to repair it. If you can help, send whatever info you can to Steve in care of the American Radio Relay League, 225 Main Street, Newington, CT 06111.

VIP GAME IMPROVEMENTS

Phil Sumner

While playing the various games available for the VIP, have you ever become frustrated or irritated at something about the game? Maybe something that the game didn't do but should have, or did do but should not have? If so, you are not alone. After keying in and playing most of the available games, my reactions ranged all the way from "excellent" (Tic-Tac-Toe and Reversi) to "forget it" (VIP Video Display Drawing Game). There were many games, however, that seemed to lack some little something that would make them better. This article presents the detailed changes I derived to improve 3 of the more interesting games; additional ideas for improving others are given at the end.

VIP WIPE OFF

As written, the program serves a new "ball" whenever the on-screen ball goes out of play and any key is depressed. This is very irritating, since a key is quite often depressed to move the paddle when the ball goes off-screen (a near miss), in which case a new ball is immediately served. To change the game so that a new ball is served only if key 5 is depressed, change location 0234 to 12CE (Go 02CE) and add:

```
     02CE      FFOA      VF = Key
     03D0      3F05      Skip if VF = 05
     02D2      12CE      Go 02CE
     02D4      1236      Go 0236
```

A similar technique is used to re-start the program after a game is over, thus eliminating some use of the RUN/RESET switch. To clear the screen and start the game over at the beginning when key A (for Again!) is depressed, change location 02C8 to 12D6 (Go 02D6) and add:

```
     02D6      FE0A      VE = Key
     02D8      3E0A      Skip if VE = 0A
     02DA      12D6      Go 02D6
     02DC      00E0      Erase
     02DE      1200      Go 0200
```

To re-start the game if key A is depressed while a ball is in play, change location 0278 to 12E0 (Go 02E0) and add:

```
     02E0      6E0A      VE ←0A
     02E2      EE9E      Skip if VE = Key
     02E4      1242      Go 0242
     02E6      00E0      Erase
     02E8      1200      Go 0200
```

The degree of difficulty of the game is set by making the "paddle" longer or shorter; this is accomplished by changing location 02CD to E0 (short), F8 (medium), or FF (long). To make this selection available to all players, change the program as given below. When starting a new game, depressing keys 1, 2, or 3 will select the difficult, moderate, or easy paddle setting; depressing any other key will re-start the game without changing the previous paddle setting. To incorporate the changes, set location 0200 to 12EA (Go 02EA) and add:

```
     02EA      FE0A      VE = Key
     02EC      A2CD      I = 02CD
     02EE      6000      V0 ← 00
     02F0      4E01      Skip if VE ≠ 01
     02F2      60E0      V0 ← E0
     02F4      4E02      Skip if VE ≠ 02
     02F6      60F8      V0 ← F8
     02F8      4E03      Skip if VE ≠ 03
     02FA      60FF      V0 ← FF
     02FC      3000      Skip if V0 = 00
     02FE      F055      MI ← V0:V0
     0300      A2CC      I = 02CC
     0302      1202      Go 0202
```

The game can be made more interesting if the ball is allowed to move vertically in addition to moving diagonally. This is accomplished by changing the code at locations 023C and 028A to call a new subroutine at 0304. With the code changes, there is a 25% chance that vertical motion will result after each impact with a spot or the paddle. The code changes are:

```
     023C      2304      Do 0304
     023E      1242      Go 0242
     0240                Not Used
     ~~~
     028A      2304      Do 0304
     028C      1290      Go 0290
     028E                Not Used
     ~~~
```

Add the following code for subroutine V4

```
     0304      C401      V4 = RND
     0306      3401      Skip if V4 = 01
     0308      64FF      V4 ← FF
     030A      CE03      VE = RND
     030C      4E00      Skip if VE ≠ 00
     030E      6400      V4 ← 00
     0310      00EE      Return
```

With these changes, the memory location for temporary storage of the score should also be changed (the old location is now used for something else). Location 02AE should therefore be changed from A2F0 to A320.

VIP BREAKOUT

As written, the program serves a new "ball" whenever the on-screen ball goes out of play and any key is depressed. To change the game so that a new ball is served only if key 5 is depressed, change location 0234 to 1300 (Go 0300) and add:

```
     0300      FF0A      VF = Key
     0302      3F05      Skip if VF = 05
```

```
     0304        1300        Go 0300
     0306        1236        Go 0236
```

A similar technique is used to re-start the game after the game is over, thus eliminating some use of the RUN/RESET switch. To clear the screen and start over at the beginning when key A (for Again!) is depressed, change location 02C8 to 1308 (Go 0308) and add:

```
     0308        FE0A        VE = Key
     030A        3E0A        Skip if VE = 0A
     030C        1308        Go 0308
     030E        00E0        Erase
     0310        1200        Go 0200
```

To re-start the game if key A is depressed while a ball is in play, change location 0278 to 1312 (Go 0312) and add:

```
     0312        6E0A        VE←0A
     0314        EE9E        Skip if VE = Key
     0316        1242        Go 0242
     0318        00E0        Erase
     031A        1200        Go 0200
```

As received, the game seemed to suffer from a lack of action; in particular, the tendency of a "hit" to rebuild some parts of the wall seemed to interfere with game enjoyment. A section of code controlling ball motion was therefore re-written with some major changes; these changes increased the "action" considerably and reduced the effects of the wall rebuilding. V4 is allowed to have a 00 value 25% of the time, thus introducing purely vertical ball motion part of the time. In addition, the code at 0328 and 032E simply reverses the direction of vertical flight of the ball at wall impact; the probability that this will not happen on each impact is set by location 032A. The ball will therefore break through the wall on occasion; half the time if 032A is CE01, 25% if CE03, 12% if CE07, etc. To incorporate the changes, set location 028A to 131C (Go 031C) and add:

```
     031C        C401        V4 = RND
     031E        3401        Skip if V4 = 01
     0320        64FF        V4←FF
     0322        CE03        VE = RND
     0324        4E00        Skip if VE ≠ 00
     0326        6400        V4←00
     0328        6DFE        VD←FE
     032A        CE03        VE = RND
     032C        3E00        Skip if VE = 00
     032E        85D3        V5←V5 ⊕ VD
     0330        1242        Go 0242
```

It was found experimentally that the quickest breakthrough occurred if the paddle was not moved at all; just serve the balls and let them go where they may. To rule out this technique in play and also to add more interest, I added a patch to move the paddle to a new and random location after each ball goes out of play. To incorporate, change location 02A8 to 1332 (Go 0332) and add:

```
     0332      A2CD      I = 02CD
     0334      D011      Show 1MI at V0, V1 (Erase Paddle)
     0336      C03B      V0 = RND
     0338      D011      Show 1MI at V0, V1 ( New Paddle)
     033A      1228      Go 0228
```

VIP KALEIDOSCOPE

The VIP Kaleidoscope game is much more interesting if diagonal spot motion and "freeze motion" capability is added. For diagonal spot motion, keys 1, 3, 7, and 9 were selected to control the motion. To add this capability, change location 0242 to 1300 (Go 0300) and add:

```
     0300      4001      Skip if V0 ≠ 01
     0302      1320      Go 0320
     0304      4003      Skip if V0 ≠ 03
     0306      1328      Go 0328
     0308      4007      Skip if V0 ≠ 07
     030A      1330      Go 0330
     030C      4009      Skip if V0 ≠ 09
     030E      1338      Go 0338
     0310      A277      I = 0277
     0312      1244      Go 0244
     ~~~
     0320      71FF      V1←V1 + FF
     0322      72FF      V2←V2 + FF
     0324      1310      Go 0310
     ~~~
     0328      7101      V1←V1 + 01
     032A      72FF      V2←V2 + FF
     032C      1310      Go 0310
     ~~~
     0330      71FF      V1←V1 + FF
     0332      7201      V2←V2 + 01
     0334      1310      Go 0310
     ~~~
     0338      7101      V1←V1 + 01
     033A      7201      V2←V2 + 01
     033C      1310      Go 0310
```

Program freeze capability using key F is added by changing location 0230 to 1340 (Go 0340) and adding:

```
     0340      690F      V9←0F
     0342      E9A1      Skip if V9 ≠ Key
     0344      1340      Go 0340
     0346      1350      Go 0350 (To re-start patch below)
```

I also added the capability to re-start the program using key A, which is very handy when exploring for new and interesting patterns. Don't hold the key down too long, however, or it will register as the first selected digit (a blank).

```
     0350      690A      V9 ← 0A
     0352      E99E      Skip if V9 = Key
     0354      121E      Go 021E
     0356      00E0      Erase
     0358      1200      Go 0200
```

ADDITIONAL IDEAS

Several other games could also be improved, given the motivation, time, and ingenuity. Several ideas that have occurred to me while playing the games are given below; maybe one of these will turn you on sufficiently to produce the noted change or an even better one.

1. Change Armored Car Clash so that diagonal as well as horizontal and vertical tank motion, pointing, and firing is possible. This is a toughie, but would be well worth it.

2. Expand Mastermind to allow digits 0 through 9 in the 4 digit version. Also evaluate the possibility of a better method of handling duplicate digits; the present program tells you about it if a digit is used, but does not tell you if the same digit is also used elsewhere in the secret number.

3. Change Deduce as required to handle duplicated digits in a more logical way.

4. Change Figure Shooting at a Moving Target so that more skill is involved in hitting the target. Right now, firing on a random basis is likely to give as good a score as intentional firing.

5. Revise Dot Dash so that the spot acceleration feature is toned down considerably. So far, everyone trying the game at my house has immediately lost interest due to an inability to keep from crashing. I can run the obstacle course myself by keeping 0 depressed and only letting up long enough to change spot direction with one of the other keys, but this solution is too drastic. The game would be much more interesting if the accelerative action were cut to half (or a fourth) of its present value, thus allowing controllable acceleration.

6. Add key selection of the maze pattern in VIP A-Mazing. Use a technique similar to the paddle selection routine in Wipe Off? And maybe add other maze patterns?

7. Change the hunter shoot conditions in the Most Dangerous Game. Requiring 0 to be pressed to shoot on the first 3 moves is OK; requiring 0 to also be pressed if you don't want to shoot after 3 moves is ridiculous.

RCA COSMAC VIP **VP-575**

System Expansion Board

Adds Four Buffered Expansion Sockets To The COSMAC VIP

Simple Plug-In Connections

No Wiring or Soldering

Sturdy Self Supporting Mount

Allows Simultaneous Use of Up to Five Accessory Boards

All Signals to All Sockets

Now you can significantly expand your COSMAC VIP System using simple plug-in connections. The sturdy, self supporting plug-in System Expansion Board, VP-575 includes 4 buffered expansion sockets plus a duplicate of the COSMAC VIP unbuffered expansion socket. Plugs directly into the VIP System Expansion Socket. You can operate up to five plug-in card edge accessory boards without wiring or soldering.

With the VP-575, System Expansion Board your COSMAC VIP system can include:

- Up to 24k total RAM including the 4k of the COSMAC VIP. Just plug-in five VP-570 Memory Expander boards.
- Up to 20k of ROM. This can include the VP-700, Tiny Basic ROM board to allow programming with this VIP high level language.
- Combinations of RAM and ROM.
- The VP-590, Color Display board (which must be used in the unbuffered socket) plus combinations of RAM and ROM.
- Many user-designed accessories or I/O ports for control applications.

Outputs:	
Buffered	**4 22-pin card edge connector sockets – all COSMAC VIP expansion socket signals to all 4 sockets.**
Unbuffered	**1 22-pin card edge connector socket duplicate of the COSMAC VIP expansion socket – (for applications where timing is critical).**
Mounting	**Self supporting. Plugs directly into the prewired 22-pin card edge expansion connector of the COSMAC VIP.**
Power	**From VIP. Additional VIP power may be necessary to accomodate the accessory boards used.**
Size	**4.5" x 6.8" x 1.7" height.**
Shipping Weight	**1.5 pounds.**

NEW PRODUCT ANNOUNCEMENT

George Risk Industries, 1222 P Street, Lincoln, NE 68508 has announced production of the Model 771 Keyboard Subsytem designed sepecially for personal, small business, and educational microcomputer systems. They Model 771 includes 56 alphanumeric keys which provide the entire 128 character ASCII set, including lower case and a separate 15 key numeric/cursor control keypad.

Four modes of encoding, four power supply options, versatile interface, and parity sense are among the features of the 771 Keyboard. An optional adapter, mounted inside the keyboard enclosure, permits conversion of the keyboard to a completely self-contained keyboard transmitter with 110-9600 baud RS232 or current-loop serial data output.

Other standard features are auto-repeat, two-key rollover, fully buffered outputs, parallel interface, Dseries connector, two-shot molded keycaps, and all-steel desktop enclosure.

High reliability GRI KBM Series key switches and LSI coding techniques assure dependable operation and low power consumption. Full documentation and application notes are provided.

Fully assembled, tested, and warranteed for 90 days, the GRI Model 771 Keyboard is ready for immediate use. The pricing begins at $150, and is dependent upon user-selected options. For ordering and delivery information, contact George Risk Industries, GRI Plaza, Kimball, NE 69145, or call (308) 235-4645.

* And tell 'em you read about them in The VIPER!

PERFORMING LONG BRANCHES IN MACHINE LANGUAGE WITH VIDEO ON

by Tom Swan

One thing I've had trouble with in the past is performing long branches in machine language with the video interface on. One way out of the problem is this:

Use a utility register (I always reserve Rf for these things) as a temporary program counter. Let's say you're using R3 as your program counter (P=3), and you want to jump from 0250 back to 0165. Try this:

```
ADDRESS    CODE          COMMENTS
024A       F8    LDI     Load D with
024B       01            High order of jump address and
024C       BF    PHI     put it into Rf.1
024D       F8    LDI     Load D with
024E       65            Low order of jump address and
024F       AF    PLO     put it into Rf.0
0250       DF    SEP     Set program counter = Rf
```

Control will now jump to 0165 with Rf as the program counter. At the jump-to point, it will be necessary to repeat the above routine, setting R3 equal to the appropriate address, and restoring it with a D3 instruction as the program counter. A more complex answer would be to use a subroutine to manage the exchange of program counters and control the long jumps.

Also, you could turn off the video, perform the long branch, then turn it back on at the jump-to point; or just use the long branch instructions (although the screen will flicker each time the instruction is encountered, as it takes three machine cycles to execute, and it messes up the video timing).

CORRECTIONS

Editor's apology delivered with bright blushes and sincere sighs of regret - not to mention a generous helping of chagrin:

Sorry, folks. We goofed in retyping the code for the Lunar Lander in last month's issue. Udo had sent his program in all neatly typed - but we couldn't get clean copy from it, so had to retype it. And (as one might suspect) introduced a few (!) errors.

So here are the corrections

At location 0254: F0 33
At location 02F8: 00 EE

And, at location 0340, there's a whole subroutine missing!

```
0340  41 00
0342  13 5E
0344  82 85
0346  80 F0
0348  80 E5
034A  81 04
034C  31 00
034E  00 EE
```

TIDBITS

One of the most interesting and startling events of the month came in the form of a letter from Larry Dolce of Valley Cottage NY. Larry explained that his VIP is too small to hold the 1802 machine language assembler he wanted, so he turned to the only other machine he had available: an IBM 370.

Larry wrote an 1802 cross-assembler in COBOL (!) - a language not usually considered optimum for such tasks. (guess we all have to make do with what we've got on hand!)

Larry will be glad to share his COBOL program with anyone who has a 360 or 370 handy which supports DOS VS COBOL Release 2.4. Larry sent along a source listing of the cross-assembler - and it really is in COBOL! And he sent a sample 1802 assembler listing from the program - and it really does work! Can any one top this one for pure inventiveness? If you'd like a copy of Larry's COBOL program, write to him at 2 Mayfield Street, Valley Cottage, NY 10989.

Has anyone tried out the EPROM programmer from Optimal Technology? That is, anyone who is willing to write it up and let us all know what he/she thinks of it?

One vote arrived at last against the Studio II and ELF-II articles we've been running. That makes it seventeen to one. Anyone else care to vote?

Stephen F Condon noted on his subscription form that he'd like to form up a VIP User group in the Medford NJ area. Interested VIPpers can contact Steve by writing to 27 Oak Court, Medford NJ 08055

J. Rothberg of Silver Spring MD would like to contact other VIP people in his area. People interested in being in touch with other VIPpers in or near the 20910 zip code area should call him at (301) 585-7322 - evenings only.

The front cover on this month's issue shows the names of some of the VIP expansion products we have ordered from RCA - which you have ordered from us. (The ASCII keyboard isn't included in the list because we don't have any promotional material from which to make a cover copy). Thanks to all for your patience. For those of you who have received your boards, how about reviewing (and writing them up) for The VIPER?

RELOCATE A PROGRAM IN SAM HERSH'S EDITOR

by Udo Pernisz

There are only two problems when using Sam Hersh's beautiful and convenient editor. The first is relocating programs written in CHIP-8 (which begin at location 0200) so you can run them and modify them in the editor. The second is relocating programs written, checked out, and debugged in the editor, so they'll run in CHIP-8 without the editor.

Naturally, the programs themselves will still be at their original locations. You have to physically move them yourself by saving them on tape and then reloading them at the desired address. The program segments below simply corrects the target addresses for all the GOTO, JUMP, computed GOTO, and I instructions, so that when you reload the program, all the target addresses have the correct values.

The addresses affected by the modifications here are all of the type XM$_1$M$_2$M$_3$, where X=0, 1, 2, A, or B, and where M$_1$ ≠ 0 or 1. To relocate "up" from 0200 to 0400, the mods add 2 to the value of X, and to relocate "down", 2 is subtracted from X. One new command, "C" (for "change addresses") is added to the command set already available in the editor.

After the user presses "C", the modified editor waits for the entry of either a 1 (for renumbering "UP") or a 2 (for renumbering "DOWN"). The program extends the numbering process over all CHIP-8 statements between location 0400 and the bottom address shown on the screen at the time key C is pressed. Before entering "change" mode, then, the user presses "A", enters the top-of-program address, and only then presses "C". This allows relocation of statements which contain data and machine language subroutines, provided they are located at addresses higher than all instructions. Since it is essential that a CHIP-8 instruction starts at an even-numbered address, the "change" routine checks for this, and returns to command mode if it finds that the address displayed starts at an odd number. After entering the command "C", the user can exit without any changes having been made, by entering any key other than 1 or 2. Both exit modes are accompanied by a "beep" sound.

In the Hersh Editor, make the following changes:

1. Change location 027C <u>from</u> 1206 <u>to</u> 1282

2. Add to location 0282 340C
0284 1206
0286 1348

3. Add the CHIP-8 and Machine language subroutine programs starting at location 0348, as listed here:

```
0348   A202    I=0202          Point to present address as
034A   F165    V0:V1=MI        given in the editor's address
034C   6201    V2=01           scratch pad, read it in.  Set
034E   8212    V2=V2&V1        a mask to check for oddity of
0350   3200    SKIP;V2.EQ.00   address by extracting LSbit of
0352   135C    GO 035C         low order address and continue
0354   F20A    V2=KEY          if even.  If not, exit "change"
0356   3201    SKIP;V2.EQ.01   mode via 035C.  Enter direction
0358   4202    SKIP;V2.NE.02   of change, check for valid entry,
035A   1362    GO 0362         if none, exit change mode, giving
035C   6248    V2=48           a "beep" tone to signal the user.
035E   F218    TONE=V2         and go back into the command mode
0360   1206    GO 0206         of the editor.

0362   A379    I=0379          Point to location of hi address
0364   F055    MI=V0:V0        in MLS and write V0 there, which
0366   8010    V0=V1           is hi current address, then do
0368   A37C    I=037C          the same with V1, which is the
036A   F055    MI=V0:V0        lo current address.

036C   60F3    V0=F3           Set V0 to F3, which turns into
036E   8024    V0=V0+V2        ADD (F4) or SD (F5) upon adding
0370   A39F    I=039F          direction entry, and write it
0372   F055    MI=V0:V0        into proper place of MLS, which
0374   0378    DO MLS at 0378  accomplishes the changes.  Then
0376   1206    GO 0206         go back into command mode.

0378   F8 xx   LDI xx          Set up register D to point to high-
037A   BD      PHI reg D       est   address of program.  The
037B   F8 yy   LDI yy          values of xx and yy are supplied
037D   AD      PLO reg D       by the CHIP-8 program in 0348

037E   F8 03   LDI 03          Set up hi-order address for data
0380   BC BE   PHI reg C       pointers, which are registers C
               PHI reg E       and E.
0382   F8 93   LDI 93          Register C points to a permanent
0384   AC      PLO reg C       address in MLS.
0385   ED      SEX reg D       Define register D as the data
                               pointer for the program instructions.
0386   F8 AB   LDI AB          (Re-)set register E to the first
0388   AE      PLO reg E       value of instructions to be changed.
0389   F8 F0   LDI F0          Load a mask for the instructions
038B   F2      AND             to be extracted: MSByte of instruction.
038C   32 96   BZ 96           If 0, found - go change it.
038E   5C      STR reg C       Else store at 0393 for checking
038F   4E      LDA reg E       against data scanned with register E.
```

```
0390   32 A3    BZ A3               Detect end-of-scan data: if 0, go on.
0392   FD zz    SDI zz              Find if scan data matches result
0394   3A 8F    BNZ 8F              of ANDing. If not, go back and load
                                    next scan data.
0396   F0       LDX                 Found one: load it and check further
0397   FA OF    ANI OF              Check for instruction not referring
0399   FF 02    SMI 02              to the first two pages (X0 or X1)
039B   3B A3    BL A3               If it does, ignore it & go on.
039D   F8 02    LDI 02              Load difference by which to change X
039F   Fx       ADD or SD           depending of value of V0 in 0372
03A0   73       STDX                Write back changed instruction.
03A1   30 A4    BR A4               Skip 1 DEC instruction, since STDX
03A3   2D 2D    DEC reg D twice          already includes one
03A5   9D       GHI reg D           Check: address still above or at 0400
03A6   FF 04    SDI 04
03A8   33 86    BGE 86              If it is, go get next instruction
03AA   D4       SEP reg 4           If not, all done; return
03AB   10 20 A0 B0 00               Hi order instruction XY.. to be
                                    changed
```

ERROR IN THE VIP USER's GUIDE

Michael Ward, an RCA employee in Wardstown, NJ, writes to tell us about some "snafoos" in the new VIP CHIP-8 User's Guide. We're reprinting his comments here so you can fix your copy accordingly.

Page 36: If one loads the program on page 36, it is soon evident that it doesn't run as intended. The displayed data (say F0 F0 F0 F0 at memory locations 0300 thru 0303) appear as expected. Pressing any key causes a flicker, due to erasure and subsequent turn-on, but no movement. The reason lies in the nature of the Keypress detection routine and the instruction F30A at M0214. Contrary to the statement on page 35, the program does not resume when or during actual key depression, but rather when the key is released (the program stops at keypress and value of key is stored in V3). A simple change from the EXA1 instruction at 0218, 021C, 0220, 0224, and 0228 to 4302, 4304, 4306, 4308, and 4305 respectively, will allow comparison of the value of thedepressed key (which was stored in V3 at 0204) with the constant contained in the 4XKK instruction, and when equal, the appropriate X or Y variable is changed.

Page 33: A typo on the lower right of the page. Change program entry at 0214 from 321B to 321E and comment to 1E

COMMENTS

SIGN UP NOW FOR VOLUME TWO!!!

There is only ONE issue remaining in Volume I of The VIPER. As of this writing, 19 people have renewed their subscriptions. Does this mean we're falling down on the job? When you renew, jot down - HERE - what you'd like to see more of next year. We'll try to be responsive. Be sure to check the box in the subscription blank below for VOLUME II if you're renewing - so we won't get you confused with people who are still subscribing to Volume I.

YES! I'd like to subscribe to the VIPER and receive all ten issues of this year's volume! I enclose $15.00 in full payment.

Name ______________________________
(Please print or type)

Address ______________________________

City ______________ State ______________ Zip ________

Cash, Check, Money Order Enclosed ______________________________

MC/VISA/BAC Number ______________________ Exp. Date ____________

MC Interbank No. ______________________________

Required Credit Card Signature ______________________________

□ You may let other VIP owners in my area know I have a VIP, so they can contact me.

□ I am interested in forming/joining (circle one) a VIP Users group

□ I'd like to see articles in the VIPER about:
____
____
____
____

MAIL TO: VIPER; P.O. Box 1142, Columbia, MD 21044 VOLUME I □ VOLUME II □

VIPER
P.O. Box 1142
Columbia, MD 21044

Place
Stamp
Here

ARESCO
P.O. Box 1142
Columbia, MD 21044
THE PAPER · VIPER · RAINBOW · SOURCE

TO: