VIPER

AvRIL - MAY 81

Volume 3 number 1 Journal of the VIP Hobby Computer Assn. 2nd Printing

The VIPER was founded by ARESCO, Inc. in July 1978

****

Contents

EDITORIAL.................................... 3.01.02

CONSTITUTION 3.01.03

READER I/O 3.01.04

SOFTWARE

VIP Clock Program ....(by Bill Fisher)... 3.01.05

Little Loops ...Scroll the VIP... (by Tom Swan)......................... 3.01.06

ADVERTISEMENT 3.01.10

TUTORIAL

VIP Flow Charts .....(by Phil Sumner)... 3.01.11

CHIP-8 INTERPRETER

CHIP-8Y ....(by Bob Casey)..... 3.01.18

ODDS & ENDS

75 Ohm Video Output ...(by Bob Casey).... 3.01.19

VIP Tape Copy Program 3.01.19

ADVERTISEMENTS 3.01.20
3.01.21

READER I/O 3.01.22

The VIPER, founded by ARESCO, Inc. in July 1978, is the Official Journal of the VIP Hobby Computer Association. Acknowledgement and appreciation is extended to ARESCO for permission to use the VIPER name. The Association is composed of people interested in the VIP and computers using the 1802 microprocessor. The Association was founded by Raymond C. Sills and created by a Constitution, with By-Laws which govern the operation of the Association. Mr. Sills is serving as Director of the Association, as well as editor and publisher of the VIPER.

The VIPER will be published six times per year and sent to all members in good standing. Issues of the VIPER will not carry over from one volume to another. Individual copies of the VIPER and past issues, where they are available, may be sent to interested people for $3 each. Annual dues to the Association, which includes six issues of the VIPER, are $12 per year.

VIP and COSMAC are registered trademarks of RCA Corporation. The VIP Hobby Computer Association is in no way associated with RCA, and RCA is not responsible for the contents of this newsletter. Members should not contact RCA regarding material in the VIPER. Please send all inquiries to VIPHCA, 32 Ainsworth Avenue, East Brunswick, NJ 08816.

Membership in the VIP Hobby Computer Association is open to all people who desire to promote and enjoy the VIP and other 1802 based computers. Send a check for $12 payable to "The VIP Hobby Computer Association" c/o Raymond C. Sills 32 Ainsworth Avenue, East Brunswick, NJ 08816 USA. People outside of U.S. Canada, and Mexico, include $6 extra for postage. All funds in U.S. dollars, please.

Contributions by members or interested people are welcome at any time. Material submitted by you is assumed to be free of copyright restrictions and will be considered for publication in the VIPER. Articles, letters, programs, etc., in camera ready form on 8½ by 11 inch paper will be given preferential consideration. Many dot-matrix printers will not copy well, so this material has to be re-typed before it can be used. Please send enough information about any programs so that the readers can operate the program. Fully documented programs are best, but "memory dumps" are OK if you provide enough information to run the program. Please indicate in your material key memory locations and data sections.

Advertising Rates:

1. Non-commercial classified ads from members, 5¢ per word, minimum of 20 words. Your address or phone number is free.

2. Commercial ads and ads from non-members, 10¢ per word, minimum of 20 words. Your address or phone number is free.

3. Display ads from camera ready copy, $6/half page, $10/page.

Payment in full must accompany all ads. Rates are subject to change.

If you write to VIPER/VIPHCA please indicate that it is OK to print your address in your letters to the editor if you want that information released. Otherwise, we will not print your address in the VIPER.

EDITORIAL

Dear VIPers:

Well, VIPER is back! I hope that we will be able to approach the quality of the VIPER from ARESCO. First, I want to thank all of you for your support, the money which makes this project possible, and the material which several of you have already sent in. We do need material from all of you to make this effort truly successful. As mentioned in the form letter that most of you received, anything would be appreciated, even a postcard with an idea or hint. I was quite pleased to receive the programs, comments and letters that many of you sent in reply to my letter. Particular thanks to: Bill Fisher, who even included a tape of his program; George Gadbois, who sent in some very nice Ham Radio related VIP material, and the big box of stuff from Tom Swan. All of that material is just great and will be included in future issues of the VIPER.

The idea of forming a non-profit association seemed to me to be a good idea, mainly because Terry told me that the VIPER was never really profitable. Therefore, why not "not worry" about making a profit and use the non-profit status to our advantage? The money you sent in to our association is going to be used to run the association and publish the newsletter.

In order to form the association, I drafted a Constitution and consulted with several VIPers in my area. I guess we were the "Founding Fathers" of the Association. I've agreed to act as the Director of the Association in order to get it going, and I'll keep it going, until you, as members, decide that you want a change. The use of a constitution makes us a real organization with democratic means to change the association and its Directors if necessary. In drafting the Constitution, I tried to be flexible yet simple in the wording to make it easy for the Association to function and pursue the main goal: publish the VIPER!

I'm planning to publish the VIPER on a bi-monthly basis, so you can expect the next issue in about two months. Of course, if the situation permits, we may be able to publish more than 6 times per year. It all depends on how much material we have to publish and how many members we have. So spread the word.

I think that one of the most valuable functions of a newsletter is to put members in touch with one another. We all have a common interest and I think that other VIPers would like to hear about what you have done with your system. Some of you have developed programs, interfaced your VIP with printers, control systems, keyboards, and so on. That fact that you have done so will interest someone who might be planning to do the same thing.

So, I hope that you will find this effort useful and rewarding. Please pass the word on to anyone you know who would be interested in being a part of our association. I'll keep you posted on how the association is doing.

Yours, Ray

Raymond C. Sills

P.S. For those of you who are Ham Radio operators, we have a "net" of sorts on Tuesdays, 9PM on 3860KHz, +/- QRM. Let me know if you think another frequency would be better. Look for me (K2ULR) or K2ZLU, AE1I, W2OC, KA1IU.

Constitution

and

Articles of Association

I. The name of this association shall be The VIP Hobby Computer Association.

II. A. The purpose of this association shall be to promote and facilitate the hobby and recreational use of the VIP and similar computers and their accessories for the enjoyment and education of the membership.

B. Toward this end, the association will publish a Journal to permit the exchange of news, ideas, programs, information, and so on, as it pertains to the VIP computer or computers using the 1802 microprocessor.

III. Membership shall be open to all persons who desire to support the purpose of the association, regardless of race, creed, color, sex, or physical handicap.

IV. The Association shall be governed by a Board of Directors elected by the membership, in accordance with the By-Laws of the Association.

V. Meetings may be held from time to time, but members shall primarily associate by means of the Journal.

VI. The Director(s) of the Association may establish By-Laws governing the operation of the Association, subject to the approval of the membership.

VII. The Constitution and the By-Laws may be amended by majority vote of the membership.

VIII. A. The Association shall be empowered to collect money from the members in accordance with the By-Laws.

B. All money under the control of the Association shall be used to operate the Association and publish the Journal.

IX. In the case of dissolution of the Association, any money, less the expenses of dissolution, shall be returned to members in good standing, as defined in the By-Laws, in equal portions to each member.

Reader I/O

Hi Ray --- Was glad to hear you are going to get a VIP Newsletter going again. Like you, I looked forward to reading the many articles.

I don't have a VIP, just an ELF I put together from the article in Popular Electronics. Right now I have 4½K memory using a modified VIP operating system at location 8000-81FF. My cassette tape input and output uses a 8251 chip at a standard 300 baud, Kansas City 12 bit format, using the DMA IN and OUT feature of the 1802. The 8251 is located at address D000.

The only programming I have done is changing the VIP operating routine to run my tape input/output, a search program to find a certain byte in a program and a cassette tape checking program using the DMA IN circuit of the 1802.

Now I am trying to get a display circuit working using the 6847 chip. I would like to add 4 to 8K more of membry. This time using a plug in card and using the 1802 44-pin buss.

I don't have a great deal of time to spend on the computer, but I do enjoy working with it and reading about what other people are doing with it. I'm waiting to read the first newsletter.

Harold W. Panzer 756 N. Morada Ave. West Covina, CA 91790

Dear Ray ---Yes, I was upset when I found out the VIPER was unable to continue. I always looked forward to receiving it. I still haven't done all the projects published in the VIPER that I wanted to do. I still fire up the VIP and practice programming now and then.

I did give it to some of my friends and buy a TRS-80. Mainly because I have someone to trade programs with and talk to. The VIP does many things better, expecially the graphics and games. The TRS-80 is not as much fun as the VIP because you can't modify it like the VIP. I do use the TRS-80 for more serious things such as word processing and record keeping.... If I get the VIP and TRS-80 interfaced, I'll let you know. Jim Brooks

Dear Ray --- I'm glad to see that the VIPER may be resumed. I missed it. Enclosed are some articles on what I've done on the VIP in the past several months. I figure you'll need articles if you start the VIPER again. (yes, indeed! - R.S.) It makes sense to publish it on a non-profit basis with dues covering expenses. I can't see anyone making much of a profit on it as there's probably not enough of us. Besides, a club is friendlier anyhow.

The articles describe projects I've designed and built or wrote. I dug out the I/O instructions out of CHIP-8X and wrote them to fit into some empty area of regular CHIP-8. Then I build the Prom board. I have about 6 sets of CHIP-8 proms I could sell or trade to anyone interested. Then I modified my VIP for the 1K by 8 EPROM that contains CHIP-8Y and the operating system and made CHIP-8Y switchable in or out. I then build a board that has a DAC chip on it. It performs D/A and A/D functions and is a programmable power supply. One other thing I've found out is what the 1802's "do not use" instruction "68" does.... Bob Casey 855 Oradell Ave. Oradell, NJ 07649

As Time Goes By.....

Bill Fisher* was kind enough to send in a neat little clock program written in CHIP-8 which is the perfect program to run when someone asks, "That's nice, but what can your computer DO?" The program features "reverse" video numerals on the screen, which is a nice change from the usual white numbers on a black background. Bill's instructions are:

1. Load the program starting at 0000---4 pages (which includes the CHIP-8 interpreter, not listed here.) I believe that the interpreter is the "regular"--not modified CHIP-8 version.
2. Set RUN/RESET switch of RUN
3. Type six digits on the Hex Keypad for the desired clock starting time using 24 hour format (ex:173055).
4. Hit any hex key to start clock running at above time setting.

Note: The byte at 02D9 can be varied around the value FA for accurate timekeeping adjustment.

```
0200  F10A    0250  1260    02A0  7201    02F0  672E
  02  F20A      52  6D3B      A2  4204      F2  F529
  04  F30A      54  FD15      A4  12BA      F4  D785
  06  F40A      56  22E8      A6  420A      F6  00EE
  08  F50A      58  7601      A8  12AE      F8  6722
  0A  F60A      5A  460A      AA  2308      FA  F429
  0C  00E0      5C  126A      AC  1260      FC  D785
  0E  6701      5E  22E8      AE  6200      FE  00EE
0210  22CE    0260  FD07    02B0  2308    0300  6718
  12  7801      62  3D00      B2  2310      02  F329
  14  F129      64  1260      B4  7101      04  D785
  16  D785      66  02D8      B6  2310      06  00EE
  18  670B      68  1252      B8  1260      08  670C
  1A  22CE      6A  6600      BA  4102      0A  F229
  1C  7801      6C  22E8      BC  12C2      0C  D785
  1E  F229      6E  22F0      BE  2308      0E  00EE
0220  D785    0270  7501    02C0  1260    0310  6702
  22  6717      72  4506      C2  6200      12  F129
  24  22CE      74  127A      C4  2308      14  D785
  26  7801      76  22F0      C6  2310      16  00EE
  28  F329      78  1260      C8  6100
  2A  D785      7A  6500      CA  2310
  2C  6721      7C  22F0      CC  1260
  2E  22CE      7E  22F8      CE  6807
0230  7801    0280  7401    02D0  A2E0
  32  F429      82  440A      D2  D787
  34  D785      84  128A      D4  7701
  36  672D      86  22F8      D6  00EE
  38  22CE      88  1260      D8  F8FA
  3A  7801      8A  6400      DA  AF2F
  3C  F529      8C  22F8      DC  8F3A
  3E  D785      8E  2300      DE  DBD4
0240  6737    0290  7301    02E0  FCFC
  42  22CE      92  4306      E2  FCFC
  44  7801      94  129A      E4  FCFC
  46  F629      96  2300      E6  FC00
  48  D785      98  1260      E8  6738
  4A  FD0A      9A  6300      EA  F629
  4C  6D25      9C  2300      EC  D785
  4E  Fd15      9E  2308      EE  00EE
```

*Bill Fisher 2 Barnard Road Armonk, NY 10504

LITTLE LOOPS by Tom Swan

SCROLL UP FOR THE MYSTERY TOUR

The normal method for displaying numbers in CHIP-8 programs is a three step formula going something like this:

1) Initialize the display
2) Erase the old number
3) Display the new number : GO to 2

But when <u>all</u> you want to do is display a numerical set of data, it is troublesome to have to always erase old numbers by redisplaying over top of them. As discussed last month, when translating a BASIC listing into CHIP-8 it is a bother to figure what to <u>do</u> with the old numbers to make room for new ones.

BASIC Interpreters use PRINT statements which may or may not be ended with carriage returns. In other words, all the lines displayed on the screen will scroll up to make room at the bottom for new lines if that is what you want. Doing that in CHIP-8 is practically impossible, producing one of the major stopping blocks when trying to CHIP-8-ize a BASIC program.

Here is a simple machine language subroutine that will scroll an entire one page CHIP-8 display upwards by one bit position. Just call the subroutine with the instruction ØMØØ (M is the page where you have entered the routine). After the scroll, the last bit row of the display is cleared to zeros. If this was not done, the display bits (if any) on the bottom would be continuously copied upwards.

Rather than present the routine followed by a demonstration program, it has instead been incorporated into the CHIP-8 version of last month's PRIME TIME BASIC*program as promised. You may enter the program exactly as listed. Use a copy of regular CHIP-8 in memory locations 0000-01FF. When you are finished, save 5 pages from 0000 onto a tape just in case you made a mistake. <u>Then</u> flip the run switch to "RUN."

The first thing you will see is a question mark in the lower left corner of the display. Enter a two digit hex number from 00 to 36 hexadecimal. The computer will first calculate and display the decimal equivalent of the value you entered followed by that many prime numbers. When finished, the program asks for the next input.

If you enter a value greater than 36 hex, it will be rejected and you will be asked for a new value. This is because the 54th prime is equal to 251, the largest prime number that may be contained in a single 8-bit variable. For primes larger than that you would need to go to double precision arithmetic -- and a much more complicated program.

One problem you will notice is speed. This CHIP-8 program is so fast you barely have time to see the results before they scroll away! A timing loop somewhere would fix that, but I've left it

*VIPER 2:08/9:38

as a project for you to consider. (Without the scrolling, CHIP-8 is capable of calculating 54 primes in 15 seconds. TINY BASIC requires 15 minutes to do the same job).

The MLS (Machine Language subroutine) is quite simple. All that it does is set two pointers RE and RF to address bytes adjacent vertically in the display. The lower bytes are moved up while both pointers are incremented through the entire display. As said before, the last line is erased to prevent it from being scrolled up continuously. To see why this is needed, enter a D4 (Return) byte at location 040F and run the program. The resulting mess would not happen if the last line was blank tc begin with and this may suggest a modification with graphics possibilities.

Finding ways to make CHIP-8 do things it's not supposed to do has always been a pet project of mine. Hope this routine proves to be useful to you!

PROJECTS:

1) Use a timing loop to slow the prime number program to a more viewable rate. Where is the best place for this loop?

2) Modify the Scroll Controller CHIP-8 sub (at 0336-0340) to show more lines at a time. (Hint: This is a one byte change!)

LAST MONTH'S ANSWER:

1) CHIP-8 PRIME TIME (for 2K unmodified VIPS)

Variables V3=X/V4=N/V5=J

```
0200  BEGIN: 6D1B  ;VD=1B -- Set VY for display -- will not change
  02  RSTRT: 2336   SCRLC -- Scroll display -- needed on loops back
  04  PRIM1: 6CØØ  ;VC=ØØ -- Set VX for display -- will change
  06         A266   QUEST -- Point I to bits for '?' mark
  08         231E   SHOW  -- Do sub to display "?" advancing VX
  0A         232E   GETDG -- Do sub -- Input and show single digit
  0C         83ØE  ;SHLVØ -- Undocumented CHIP-8 instruction
  0E         833E  ;SHLV3 --   FXYE shifts VY left
0210         833E  ;SHLV3 --   into VX.  Four shifts move
  12         833E  ;SHLV3 --   LSD to MSD
  14         232E   GETDG -- Do sub -- Input and show digit
  16         83Ø1  ;OR    -- Combine 2 digits into byte
  18         843Ø  ;V4=V3 -- Pass byte in V4 to next sub
  1A         2308   SHO1  -- Do sub -- Display "= Decimal Value"
  1C         6FØ1  ;VF=1  -- Let VF=1 for subtract next
  1E         8F35  ;VF-V3 -- If negative, V3>1 (i.e.≥2)
0220         3FØØ  ;NEG   -- Skip on negative
  22         12Ø2   RSTRT -- Don't accept bad input
  24         6F36  ;VF=36 -- Let VF=36=maximum entry
  26         8F35  ;VF-V3 -- If negative, V3>36
  28         3F01  ;POS   -- Skip on positive (i.e. V3≤ 36)
  2A         12Ø2   RSTRT -- Don't accept bad input
  2C         A501  ;BASE  -- Set I to array base address +1
  2E         6ØØ2  ;VØ=2  --
```

```
0230          FØ55   ;PUT    -- Set A(1)=2 by storing VØ
  32          64Ø2   ;V4=2   -- Simulate first prime
  34          65Ø1   ;V5=1   --   and set J=1
  36          23ØØ    SHOPR  -- Do sub to show prime
  38          64Ø3   ;V4=3   -- Set N=3
  3A   PRIM2: 75Ø1   ;V5+1   -- J=J+1 - number primes found
  3C          A5ØØ   ;BASE   -- Set I to base of array
  3E          F51E   ;I+V5   -- Add V5 to I to address A(J)
0240          8Ø4Ø   ;VØ=V4  -- Transfer "N" to VØ
  42          FØ55   ;PUT    -- Let A(J)=N
  44          23ØØ    SHOPR  -- Do sub -- Display prime A(J)
  46          935Ø   ;SK≠    -- If X=J then  end
  48   STOP:  12Ø2    RSTRT  -- Go do it again
  4A   PRIM3: 74Ø2   ;V4+2   -- N=N+2 next prime candidate
  4C          76Ø2   ;V6=2   -- K=2 -- index to array
  4E          A5Ø2   ;BAS+2  -- Set I to base address @A(2)(i.e.K=2)
0250   PRIM4: FØ65   ;GET    -- Get A(K);I=I+1 equals K=K+1
  52          82ØØ   ;V2=VØ  -- Save A(K) in V2 for later
  54          81ØØ   ;V1=VØ  -- V1 is the divisor A(K)
  56          8Ø4Ø   ;VØ=V4  -- VØ is the dividend N
  58          Ø417    DIVID  -- Do MLS divide VØ/V1 (N/A(K))
  5A          41ØØ   ;SK≠Ø   -- If remainder V1≠Ø, skip next
  5C          124A    PRIM3  -- Not prime, loop back
  5E          82Ø5   ;V2-VØ  -- Subtract to compare V2:VØ
0260          3FØ1    POS    -- Skip on VØ< =V2
  62          125Ø    PRIM4  -- Keep dividing (K already=K+1)
  64          123A    PRIM2  -- Go display prime number
```

BIT PATTERNS

```
0266   QUEST: EØ 2Ø          -- Bits for question mark using
  68          4Ø ØØ          --   a five line character.  Last
  6A          4Ø ØØ          --   ØØ byte not used.
```

DISPLAY PRIME SUB

```
0300   SHOPR: 2336    SCRLC  -- Scroll display
  02          6CØØ   ;VC=ØØ  -- VX for display (VD set before)
  04          8Ø5Ø   ;VØ=V5  -- Pass J to NUMB3 sub
  06          23ØE    NUMB3  -- Do sub to display J
  08   SHO1:  A328    EQUAL  -- Set I to bits for = sign
  0A          231E    SHOW   -- Do sub to display "="
  0C          8Ø4Ø   ;VØ=V4  -- Pass N to VØ for displaying prime
  0E   NUMB3: A324    WORK   -- Point I to work area
0310          FØ33   ;3-DD   -- Convert VØ to 3 decimal digits
  12          F265   ;GET    -- Get those digits into VØ, V1, V2
  14          FØ29   ;SET I  -- Point I to bits for VØ
  16          231E    SHOW   -- Do sub -- display one digit prime
  18          F129   ;SET I  -- Same for digits #2 and #3
  1A          231E    SHOW   --   of the prime
  1C          F229   ;SET I  --  "          "         "
  1E   SHOW:   DCD5   ;SHOW   -- Note how bottom of sub
0320           7CØ6   ;VC+6   --   is a subroutine itself!
  22           ØØEE   ;RETN   -- Return
  24   WORK:   ØØØØ   ;       -- Reserve at least 3 bytes for
  26           ØØØØ   ;       --   work space
  28   EQUAL:  ØØEØ   ;       -- Bit pattern  for equals sign
  2A           ØØEØ   ;       --    "              "          "
  2C           ØØØØ   ;       --    "              "          "

                      ;INPUT   V5=J, V4=N, VD=VY
                      ;OUTPUT  V5=V4 shown @ VCVD; VC changed
```

GET DIGIT SUB

```
032E   GETDG:  FØØA   ;VØ=KY -- Let VØ = key pressed
  30           FØ29   ;SET I -- Point to bits for LSD VØ
  32           231E    SHOW  -- Do sub to display digit & advance VX
  34           ØØEE   ;RETN  -- Return from subroutine
```

CHIP-8 SCROLL CONTROLLER

```
0336   SCRLC:  6FØ8   ;VF=Ø8 -- Set up loop counter in VF
  38   SCRL1:  Ø4ØØ   ;SCROL -- Do MLS.  Scroll up one bit
  3A           7FFF   ;VF-1  -- Subtract one from loop count
  3C           3FØØ   ;SK=ØØ -- If=0, skip next instruction
  3E           1338    SCRL1 -- Loop to continue scrolling
0340           ØØEE   ;RETN  -- Return from subroutine

                      ;NOTE:  This sub or a similar one is needed to
                      ;work the scroll MLS.  The purpose of the above
                      ;subroutine is to simply call SCROL (@0400) the
                      ;number of times required to move the prime
                      ;numbers up to the next line.
```

*** SCROLL - CHIP-8 MLS ***

```
0400   9B          SCROLL:  GHI   RB       ;Get the current display page
  01   BE                   PHI   RE       :Set RE.1=current display page
  02   BF                   PHI   RF       ;Set RF.1=current display page
  03   F8 ØØ                LDI   #00      ;Set RE.0=00, addressing
  05   AE                   PLO   RE       ;  top line of display
  06   F8 Ø8                LDI   #08      ;Set RF.0=08, addressing
  08   AF                   PLO   RF       ;  next line below top

                      ;Main loop begins here

  09   4F          SCROL1:  LDA   RF       ;Get a byte from "below"
  0A   5E                   STR   RE       ;Store byte "above"
  0B   1E                   INC   RE       ;Increment "above" pointer
  0C   8F                   GLO   RF       ;Test "below" pointer for page cross
  0D   3A Ø9                BNZ   SCROL1   ;Loop until done
```

```
                        ;Erase bottom line of display

040F  F8 ØØ   SCROL2: LDI  #0      ;Get a 00 byte into D register
  11  5E              STR  RE      ;Store in display at bottom
  12  1E              INC  RE      ;Increment pointer
  13  8E              GLO  RE      ;Test pointer
  14  3A ØF           BNZ  SCROL2  ;If not = 00 yet, loop back
  16  D4              SEP  R4      ;Return control to CHIP-8

                      ;To use in any CHIP-8 program, simply call
                      ;the number of times, or bit`lines, you
                      ;want the display to scroll up.
                      ;
                      ;Routine is page relocatable and will run
                      ;if entered into any memory page beginning
                      ;at hexadecimal $ØMØØ.
```

MACHINE LANGUAGE DIVIDE

```
0417  F8 FØ    DIVID: LDI  $FØ    ;Set up R6.Ø=$FØ to address
  19  A6              PLO  R6     ;  location in memory of CHIP-8 VØ
  1A  46              LDA  R6     ;Get value of CHIP-8 VØ
  1B  AF              PLO  RF     ;Put in RF.0 = DIVIDEND
  1C  F8 ØØ           LDI  $ØØ    ;Set RE.0=00= ANSWER
  1E  AE              PLO  RE
  1F  E6              SEX  R6     ;X=6.  R6 addresses V1 now
  20  38              SKP         ;Skip into the loop
  21  1E       DIV1:  INC  RE     ;Count number subtracts (skipped
                                  ;  first time)
  22  8F              GLO  RF     ;Get dividend (VØ)
  23  F7              SM          ;Subtract divisor (V1) addressed
                                  ;  by R6
  24  AF              PLO  RF     ;Put temporary result back in RF.0
  25  33 21           BPZ  DIV1   ;If subtraction still positive, loop
  27  F4              ADD         ;Else add back last subtraction
  28  73              STXD        ;Store remainder as new V1
  29  8E              GLO  RE     ;Get answer from RE.Ø
  2A  56              STR  R6     ;Store answer as new VØ
  2B  D4              SEP  R4     ;Return control to CHIP-8

                      ;INPUT   VØ=DIVIDEND/V1=DIVISOR
                      ;OUTPUT  VØ=QUOTIENT/V1=REMAINDER
```

VIP FLOW CHARTS

Phil Sumner

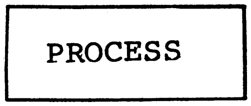
At least one of the VIPER readers has indicated a desire for an article on software flow charts, with examples. Although I personally had some reservations about any tutorials in the VIPER, that request caused some re-thinking which led to at least a tentative change of mind. I realized that there were probably many readers who were learning software "from scratch" and who were therefore completely unfamiliar with flow charts, and that many other readers were probably not completely competent or comfortable using flow charts.

I now feel that an in-depth discussion of flow charts (slanted specifically toward the VIP) might benefit both groups of readers. So those of you who would like to become more proficient with flow charts, here it is -- enjoy! And those of you who don't need it and don't want it -- have a little patience while we do a little more sorting out as to what is appropriate in the VIPER and what is not. If you have any strong feelings either way, please let the editor know; if the response is positive enough, we may even expand the tutorial theme to satisfy other requests.

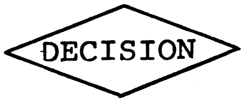
INTRODUCTION TO FLOW CHARTING

Flow charts are essential to everyone who is interested in understanding software or in writing programs; the importance of good flow charts cannot be overstated. A good flow chart is vital in generating new programs; this is particularly true for the VIP, since the program can be defined in fairly good detail (and some of the bugs removed) before any code is written. A good flow chart is also very useful with existing programs like the published games; the software flow is often much easier to understand if the existing listings are converted to flow chart form.

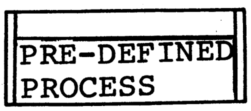
Only a few of the available flow chart symbols are needed for drawing flow charts for the VIP. The first 5 symbols given below are needed for flow charting most of the games; the last 2 symbols may also be used by advanced programmers, although the process symbol can be used instead for simplicity.



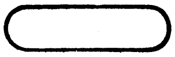
Processing function; execution of a defined operation or group of operations which results in a change in value, form, or location of information.



Switching or branching operation that determines which of a number of alternate paths is to be followed.

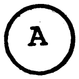


A named process consisting of one or more operations or program steps that are defined in detail elsewhere. For VIP software, this symbol is used almost exclusively for subroutines.



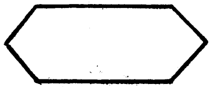
TERMINAL

A terminal point in the software flow, usually the start and end of a program, module, subroutine, or other software entity.



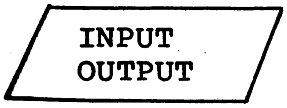
CONNECTOR

Symbol used to connect or to join 2 points in a line of flow. Used mainly to show flow continuity from one page to another, but may also be used on the same page where the flow paths are complex.



PREPARATION

Indicates the modification of an ̀instruction or group of instructions to cause a change in the program itself. Includes initialization of a routine, software-controlled code changes, or changing the designation or contents of a modifying index register.



Input or output of data to or from the computer. Documents communication between the VIP and the outside world via the I/O port.

SPECIFIC EXAMPLES

The use of the flow chart symbols can probably best be explained by showing some actual flow charts. For this purpose, I have picked a couple of the subroutines used in the VIP Armored Car Clash game.

Figure 1 shows the subroutine used for controlling the bullet position while in flight. The subroutine changes the screen location of the bullet (V3, V4) in accordance with whether key 2, 4, 6, or 8 was the last key depressed (which way the car was facing when the bullet was fired).

For VIP operations, I prefer showing the associated CHIP-8 code alongside the flow chart symbols. This has a couple of worthwhile advantages:

1. It is easier to understand the code initially. This is especially useful when I'm trying to figure out how an existing program works.

2. If I want to experimentally change something, it's easier to locate a specific function in the flow chart. Then after locating the function, the code and memory location are also right there -- I don't have to hunt for them.

Note that notations inside the flow chart retain the same sense as the code; the "skip if not" in the code is retained in the chart. The double negative interferes somewhat with an easy reading of the flow, but I personally like to maintain a functional equivalency between the flow and the code. If I were going to publish the flow chart as a separate figure, without the code, then I would change all the signs for easier reading; this is shown in Figure 2. The actual meaning is unchanged, but the flow is easier to read.

```
036A  4102  Skip if V1 ≠ 02
036C  74FF

036E  4104  Skip if V1 ≠ 04
0370  73FF

0372  4106  Skip if V1 ≠ 06
0374  7301

0376  4108  Skip if V1 ≠ 08
0378  7401

037A  00EE  Return
```

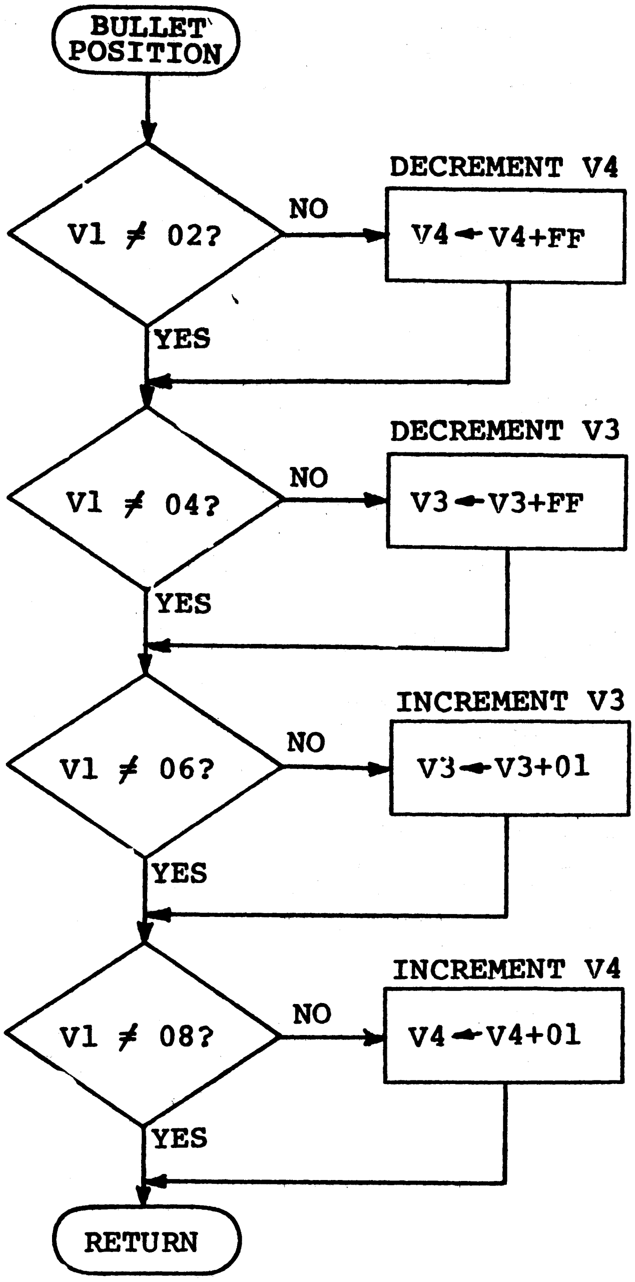


Figure 1. VIP flow chart illustrating use of the process, decision, and terminal symbols

A more comprehensive flow chart is shown in Figure 3. This illustrates the use of the subroutine call and the connector symbols. A useful technique for adding comments is also shown. Intelligent comments let you state what function the code is performing (erasing the present bullet, detecting the screen edge, etc.).

ADDITIONAL CONSIDERATIONS

There are several other points that are worth keeping in mind; these are given in the paragraphs below.

For completeness, I have shown both the standard comments and the flow in the flow charts, but you don't need both. I use just the memory location, code, and flow chart in my own work.

For your own use, you don't need to be as neat and clean as I was in the earlier figures. You can make the symbols smaller, thus fitting the flow chart to the same scale as a normal single column listing. This is shown in a copy of one of my worksheets, Figure 4.

Remember that flow charts are a tool for you to use, and that some personal preferences are therefore permissable. Use the basic symbols as they are used here; keep the flow direction from top to bottom (primary flow) or left to right (secondary flow). Add enough comments so that you will still understand the flow diagram when you look at it next week or next month.

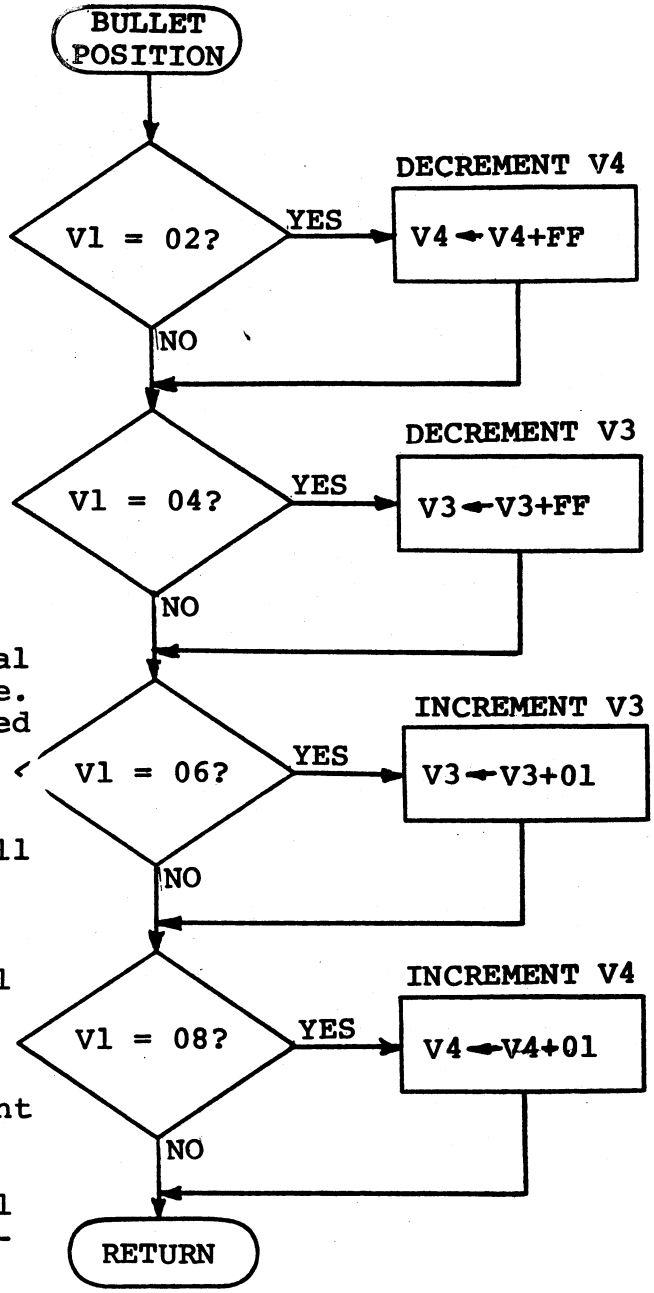


Figure 2. Flow chart optimized for reading the flow

For developing software, your initial flow charts will necessarily not be this detailed. The normal design sequence is to start with a concept or project expression and any relevant constraints, express this in a rough flow chart, then successively expand this further to define as much detail as you need. Subroutines are identified and even named in the early stages; such details as the use of CHIP-8 (utility) or machine language (operating speed) are also settled early. A final, rather detailed flow chart is then drawn just prior to writing the actual code. Quite often, the coding for subroutines is done first; location-dependent bytes are identified in some manner for later insertion of final values.

Use of this successive iteration flow chart technique gives you the best possible chance of producing a software program that runs and does what it was supposed to do. And once in a lifetime or so, this

Con'd on 3:01:16

```
02BC   A423   I = 0423

02BE   F565   V0:V5  MI

02C0   4500   Skip if V5 ≠ 00
02C2   00EE   Return

02C4   A3E9   I = 03E9

02C6   D341   Show 1MI@V3, V4

02C8   236A   DO 036A

02CA   6C02   VC = VC + 02

02CC   238E   DO 038E

02CE   4BBB   Skip if VB ≠ BB
02D0   12DA   Go 02DA
```

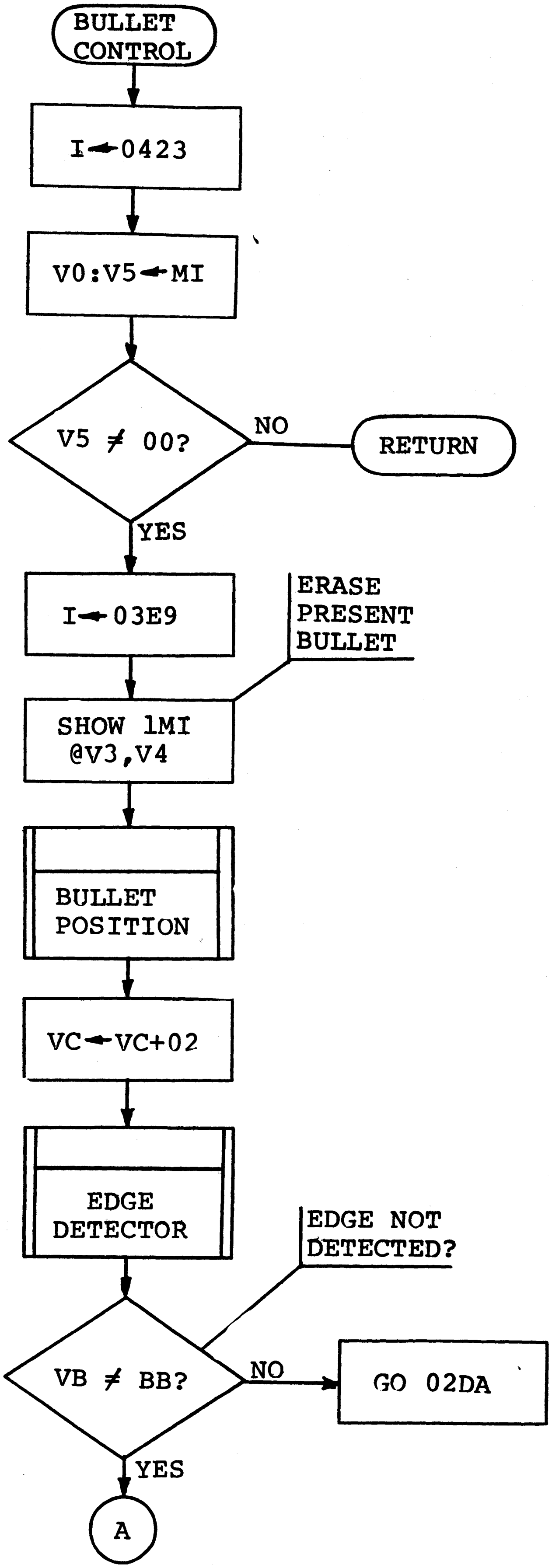


Figure 3. VIP flow chart showing use of all of the common flow chart symbols

```
02D2   D341   Show 1MI @ V3, V4

02D4   A423   I = 0423

02D6   F555   MI = V0:V5

02D8   00EE   Return
```

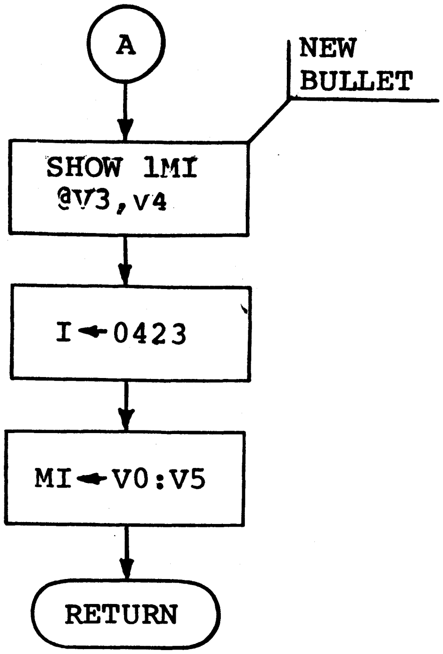


Figure 3. VIP flow chart showing use of all of the common flow chart symbols (cont'd)

may actually happen to you. But being a realist like I am, I realize that the true values of this technique are:

1. It helps organize things so that the program can be generated in minimum time.

2. It minimizes the inevitable bugs, thus minimizing the time required for de-bugging.

3. The detailed flow charts make it easier to do any required de-bugging.

```
02E4    A41D
02E6    F565
02E8    350F
02EA    1314
02EC    A3EA
02EE    D345
02F0    3200
02F2    1302
02F4    C103
02F6    A419
02F8    F11E
02FA    F065
02FC    8100
02FE    C20F
0300    7201
0302    236A
0304    A3EA
0306    6C03
0308    72FF
030A    6F00
030C    D345
030E    A41D
0310    F555
0312    00EE
```

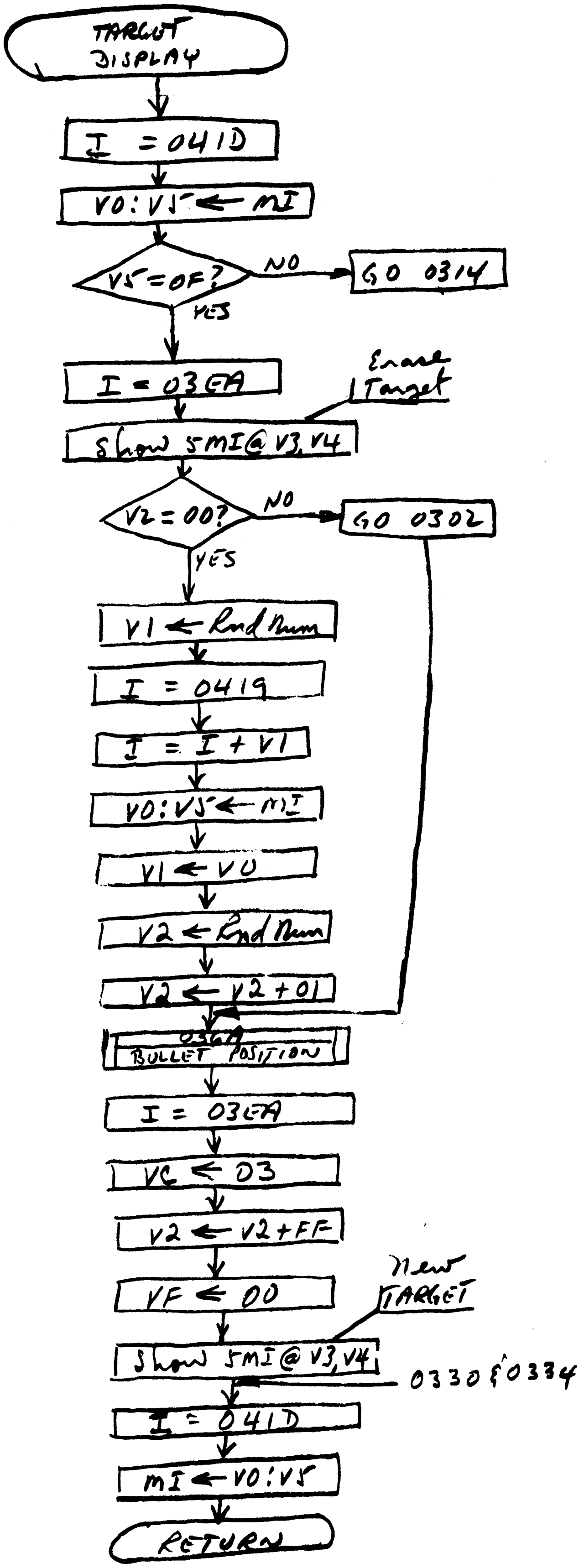


Figure 4. One of my typical flow chart worksheets

Bob Casey's** CHIP-8 with I/O Modifications: CHIP-8Y

Adding the following code will create two new CHIP-8 instructions without deleting any old ones. These new instructions deal with the I/O interface port of the VIP. To implement this modification:

1. Load CHIP-8
2. Insert the following code at 01F2:

```
01F2  00 E6 3F F4 6B D4
01F8  E6 63 D4
```

The two new instructions are:

FXF3 Get byte from input port when $\overline{\text{EF4}}$ is low and put it in VX

FXF8 Sent byte of VX to output port

Those familiar with CHIP-8X will note that these are similar to the I/O instructions of CHIP-8X. But here no old CHIP-8 instructions are changed (e.g. BMMM) and programs can still start at 0200 without having to modify them.

The following program will (with a simple sound board) demonstrate FXF8. It's a modification of a program that came with the simple sound instructions. The program on the right shows FXF3.

```
0200  6008 V0=08                    0200  6100 V1=00
  02  6100 V1=00                      02  6208 V2=08
  04  F1F8 V1=out port                04  F3F3 V3=Input when EF4 low
  06  F015 V0=timer                   06  F329 display LSD of V3
  08  F018 V0=tone                    08  D215 show
  0A  F207 timer=V2                   0A  8336 move MSD of V3
  0C  3200 skip if V2=00              0C  8336 to LDS of V3
  0E  120A branch to 020A             0E  8336   "     "
0210  7101 increment V1             0210  8336   "     "
  12  1204 branch to 0204             12  F329  display new LSD
                                      14  D1[illegible]5  of V3
                                      16  1216 stop
```

Load the FXF3 program and run it. Momentarily short $\overline{\text{EF4}}$ to ground. "FF" should appear on the screen. Now tie <u>the</u> MSB (D7) of the input port low. Run the program again and short $\overline{\text{EF4}}$ to ground. Now, "7F" should appear on the screen. The input port has pull-up resistors on the VIP.

With these two new instructions if should be possible to tie the VIP to another computer and write programs that exchange bytes between each other. I've built a small computer using the SC/MP microprocessor and included an I/O port like the VIP's. I wrote a monitor program similar to the VIP's for the SC/MP, but without the tape functions. I have written programs for both machines to exchange bytes with each other. It should be possible to write a program that splits progessing between the two machines. I don't know for sure, but you could probably tie the VIP to a TRS-80, Apple or PET.

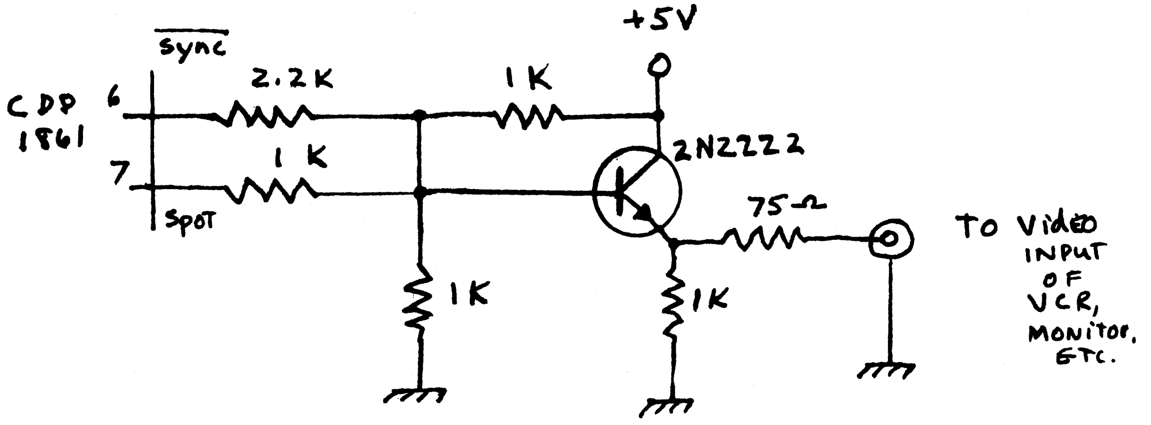
**Robert Casey 855 Oradell Ave., Oradell, NJ 07649

75 Ohm Video output for the VIP

By Bob Casey*

The circuit diagram below shows a modification to the video output of the VIP. It enables the VIP to drive 75 Ohm loads: long lengths of cable or several monitors connected in a "loop-through" method or feeding a video tape deck. The original VIP circuit is high impedance which can't drive 75 Ohn loads.

The DC bias on the output should have no effect on the video monitors or VCR's as their inputs are capacitively coupled internally after the 75 Ohm termination. This circuit should produce about 1 Volt peak-to-peak when terminated in 75 Ohms.



VIP Tape Copy Program

from
Glen C. Merritt

```
0000  3500 B200  If EF2=1. branch to 00, else R(p)+1
0002  7B   SEQ   Q=1
0003  3D03 BN2 03 If EF2=0, branch to 03. else R(p)+1
0005  7A   REQ   Q=0
0006  3000 BR00  Branch to 00
```

This program was supplied by Dale Barker as a means of copying digital programs from one cassette recorder to another. It uses the VIP as an intermediate data reader and pulse shaper. The program allows direct recording from one recorder to another by re-generating the VIP tape write tones. Since the program being duplicated is not first stored in memory, there is no limit to the size of program that is being copied. And the time to copy the program is reduced since it does not have to be loaded into memory.

CUDDLY SOFTWARE
157 CHARTER ROAD
WETHERSFIELD, CT. 06109
UNITED STATES OF AMERICA
(203) 529-3414

April, 1981

VIP Hobby Computer Association:

Cuddly Software extends Congratulations and Best Wishes on the formation of your new group and the revival of the VIPER!

Regards,

P. V. Piescik, Prop.

Notice:
As of July 1, 1981
Cuddly Software discontinued
its business operation.

P.S. If your software isn't Cuddly, it's second best!

Advertisement

For those of you who have written about the availability of the PIPs for VIPs material, this letter should interest you:

Dear Raymond---ARESCO, as you probably know by now, is no longer in business. However, I will be selling what is left of the PIPs for VIPs series as long as the supply lasts. As of now (Feb 28) I have several copies of Volume II and III along with tapes. We have sold all of Volume I, and I have no plans to reprint more of them. Instead, I am selling the Volume I tape with instructions but no program documentation. If you would like to place an order, the prices are:

Volume I tape only	$10
Volume II, book and tape	$15
Volume III, book and tpae ...	$15

Please make your check or money order payable to Tom Swan, not ARESCO. I cannot accept credit card orders at this time.

Tom Swan Box 1014 Columbia, MD 21044

Advertisement

Grapevine information

For those of you who might need a 5V 3A power supply, Radio Shack has a nice KIT supply (catalog #28-5015) which has been discontinued and may still be available at a close-out price (less than $20.) Check around and have the salesperson check the close-out listing from Ft. Worth to get the price break. The kit is easy to build with Heathkit grade instructions. The supply has an ampmeter on front panel.

READER I/O

Dear Mr. Sills....It's great that someone is keeping up the VIPER. I was so sad to see it go. At last, maybe some of those promised articles in the last issue of VIPER will be seen. After the demise of the VIPER, ther became such a lack of things to read about the VIP, we bought another computer. Alas and alack! The new system can do super word processing, database management, run CP/M or HDOS operating systems, but only the VIP can play Animal Race (RCA Game book #2) with any degree of skill.

My father and I are currently working on an RS-232 post for the VIP which would allow our Heathkit H-89 computer and our printer to communicate with the VIP. We currently have our H-9 connected to our VIP, with working character IN/OUT interfaces. I am tentatively wroking on a Text Editor for the VIP, but it will probably go the same way as my Forth compiler/interpreter and machine languare assembler...our the window. It would be <u>much</u> easier to write programs were there an Editor/assembler.

Tom Swan's CHIP-8 Assembler (PIPS II) is a step in the right direction, but I have never written any great programs in CHIP-8 other than some trivial games. What is needed is a machine language assembler, something like RCA's Level I Assembler on the high-priced COSMAC DevelopmentSystem.

Another thing that is needed is some sort of standard Disk Operating System. Programs would be much easier to save, load, and manipulate were there a disk. One does not have to rewind a disk to the beginning to gain access to the first file, or fast-forward to the end to use the last one. Your simply say, "get me this file," and it does. The disk, however should ABSOLUTELY NOT be 8-inch! Both the drives themselves and the disks are two to three times the cost of five-inch. Besides five-inchers take up less room, and are easier to obtain....

William Lindley 21 Hancock St. Bedford, MA 01730

Dear Ray----In my many readings of the VIPER Vol. 2, I came across some references to an article in Vol. 1 that gave a breakdown of CHIP-8. I was wondering if that could be reprinted or copies of Vol. 1 be made available for sale. My second desire would be for some very basic tutorial in machine language. My last request would be for some information: I've tried to set up the high resolution to use the simple sound board. So far, all attempts have ended in failure. Any help in this last area would be greatly appreciated.

Gary Cordes 1100 Childers N.E. Albuquerque, NM 87112

----Well, Gary, I'm not sure if the CHIP-8 breakdown can be reprinted, but you might consider getting the RCA publication VIP-320, "RCA COSMAC VIP User's Guide" if you don't already have it. The VIP-320 book, however, does not have the line-by-line analysis of the CHIP-8 interpreter that was in John Wentworth's article in Vol. 1. The machine language tutorial idea would interest a great many people. And one of our members has indicated that he might be able to provide that sort of material. I hope so--but we will have to wait and see!--RS

VIP Hobby Computer Association
32 Ainsworth Avenue
East Brunswick, NJ 08816

A final word:

Preparing this issue of the VIPER was a very interesting experiance! And I would welcome your suggestions and criticisms: after all, this newsletter is supposed to serve your interests. Please feel free to send in your articles or programs. Anything you can send would be helpful. Of course, material that is neatly typed and camera ready is even more helpful, but anything would be welcome.

ARTICLES PLANNED FOR COMING ISSUES:

1. Simple Music Program Part 2 by Udo Pernisz
2. COSMAC VIP Autocall by George S. Gadbois
3. VIP Operating System for Elf by Leo F. Hood
4. CHIP-8 in EPROM by Bob Casey