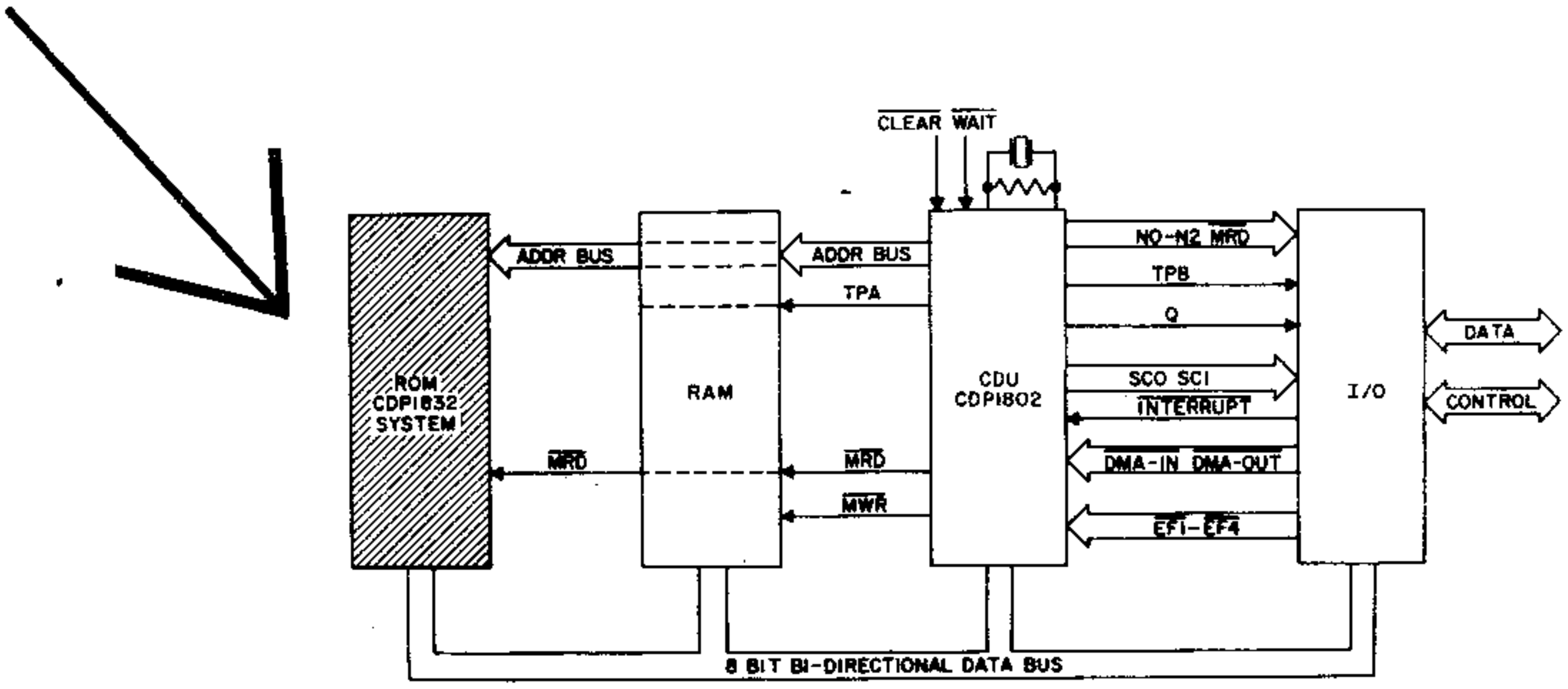
VIPER

VOLUME 1 SEPT 1978 ISSUE 3

ROM REVEALED!



PLUS:

CHIP-8I
HIGHER RESOLUTION DISPLAY

AND THAT AIN'T ALL!

$2.00

AN ARESCO PUBLICATION

EDITORIAL

Following hard on the heels of our expose of CHIP-8 in the last issue, we follow up this month with an equally in-depth look at the ROM operating system of the VIP. These were the two most-requested items in your letters over the last two months. Another frequently-requested item was information on using double-resolution display with the VIP. Jeff Windsor and Andy Modla of RCA's Sarnoff Labs put their heads together and have come up with two-page graphic display routines which work with CHIP-8. Their contribution is also in this issue. Because of the wealth of material in this issue, the second installment of Don Stein's VIP editor had to be delayed until next month. Next month's issue will also have the first information on RCA's new ROM Tiny BASIC for the VIP.

One look at the RCA ad in this issue will convince you that they are really dedicated to a full line of useful peripherals for the VIP. Make sure the computer store near you knows about the VIP and intends to carry all the new goodies!

SUBSCRIPTION RATES, ADVERTISING RATES AND 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, subscription price is $15.00 per year (all ten issues of one voiume.) Dealer prices upon request. Outside of Continental U.S. and Canada, add $10.00 per subscription for postage ($1.00 for single copy).

Readers are encouraged to submit articles of general interest to VIP owners. Material submitted will be considered free of copyright restrictions and should be submitted by the 1st day of the month in which publication is desired. Non-profit organizations (i.e., computer clubs) may reprint any part of the VIPER without express permission, provided appropriate credit is given with the reprint. Any other persons or organization should contact the editor for permission to reprint VIPER material.

Advertising rates are as follows:

1/4 page	–	25.
1/2 page	–	45.
3/4 page	–	65.
full page	–	85.

Less than 30% of the VIPER will be available for advertising. Please send camera ready copy in the exact page size of your ad on 8-1/2 x 11 white stock by the 1st day of the month in which you'd like the ad to appear. Photos should be glossy black & white in the exact size to be printed. Payment required with copy.

The VIPER is an Aresco Publication, edited by Terry L. Laudereau. For information contact Editor, VIPER, P.O. Box 43, Audubon, PA 19407.

The VIPER is not associated with RCA in any way, and RCA is not responsible for its contents. Inquiries should be directed to ARESCO at the address above or by telephone to (215) 631-9052.

LETTERS

We always enjoy hearing from VIP owners. Here are some of the letters we've received that we would like to share with you.

Dear VIPER,
Much congratulations and thanks for your excellent first issue.
Please don't dismiss us COSMAC machine language users as "hard core computer freaks." It's just that we want to use the VIP for some things that CHIP-8 can't do too well; like responding to the outside world.
I have had a great deal of trouble trying to get the VIP to handle external interrupts; and though the VIP User Manual suggests that the EF3 line can be used with care, I'm not convinced.
I suggest that part of each issue be devoted to some of these trickier aspects of the VIP. Maybe a problems write-in column could be incorporated. I'd be glad to help.

Peter Colin
Marine on St. Croix, MN

(Thanks, Peter. We will try to balance the VIPER content to satisfy both beginners and 'freaks'. I hope the article on CHIP-8I will help you interface to the real world. Now that single-chip A/D converters are available for $5, I hope some reader will send us a circuit. Also, I know that an R/2R ladder network could be hung right on the output port for D/A conversion - has anybody done it?

If you use the VIP video output you must give up the interrupt capability since the video output depends on it. The best you can do is hang in a loop waiting for EF3 or EF4 to change.)

Dear VIPER,
CHIP-8 is too limited for my purposes. I use machine code and a Pascal-like language of my own invention.
I think to VIP is overall a better value than the other small machines. 2K of RAM is a good amount for a minimal machine, and the provision for easy and inexpensive expansion is most welcome. Unless all you want to do is time an egg, 256 bytes is useless. It's remarkable how frequently 1K of RAM seems to be not quite enough, but for the things I am doing now, 2K is ample.
The manual doesn't mention a couple of important facts about the ROM interrupt routine. (p.20) The routine increments R9 each time it is entered. To sound a tone n/60 seconds long, store n in R8.0. You can't use the Q as a program flag, because the interrupt routine may alter it at an unpredictable moment.

Norman Whaland
New Yorn, NY

(Pascal-like language?! Tell us more! Hope the information on the ROM program in this issue helps out in your understanding of the interrupt routine. The info on two-page display for CHIP-8 should also turn you on.)

Dear VIPER,
I have a copy of Tom Pittman's Tiny BASIC from Infinite, Inc. of Melbourne, FL. I use my old Model 19 TTY as an I/O device. I wrote a machine language program (about 500 bytes) to do the serial to parallel/ parallel-serial and Baudot to ASCII conversions. If you are interested, my software and hardware could probably be described in an article.

F.L. Shippey
Penfield, NY

(Anybody else have a Model 19? We'll be happy to forward your letter to Mr. Shippey if you'd like more info.)

Dear VIPER,
I have hooked up a Radio Shack keyboard. This led to a custom monitor with driver routines using the VIP tape routines. Of course relays were added to control the tape.
Currently I am wire-wrapping a VIP-to-S100 interface card. This will attach an 8K memory board and 4 ports to the VIP card. I am interested in sharing these ideas if they are original or have improved features the readers would want.

M.J. Steckmyer
Ft. Collins, CO

LETTERS (continued)

(One of the nice things about the 1802 is that it lacks the complex control line structure of the 8080. I don't know how you would simulate all those lines needed on the S-100 bus, but good luck! We sure would like to hear more about your keyboard monitor and tape routines.
The great thing about working with CMOS is that if you avoid TTL in your interface circuits and stick with MOS or CMOS, the noise problems and loading considerations which make interfacing so difficult just go away. Who's going to show us how they added more I/O using the RCA CDP1852 chip?)

Dear VIPER,
I am just learning to use the system. I have stored a complete alphabet in memory and on tape and can now create any message such as 'Happy Birthday'. I am working on a crap game but only have the dice roll programmed so far. Hope to get win and lose, with point display and then later a 'bank' display. I hope to get into analog to digital conversion and real time control. I appreciate your comments on CHIP-8. I've gotten started with CHIP-8 but still haven't learned to use all the instructions.

James Brooks
Albuquerque, NM

(Keep up the good work, Jim. I think the VIP is good for a wider variety of users than any other machine on the market today. The wide variety of interests shown by our readers proves it.)

Dear Editor,
Both my brother and I own VIP's! My brother lives in New York and I in Phoenix. He assembled both VIP's and any other hardware that we use. I write the software for both of us. The team is unbeatable! I find the machine language relatively easy to code in, although it is very frustrating to run into bottlenecks with the 'D' register.
I find CHIP-8 to be an amazing piece of software, considering that it resides in only ½K or RAM. I would like to see a BASIC interpreter for VIP, if only because of the numerous programs that are available in that language. The advent of an assembler would also be of great use to programmers of the VIP. I am currently working on four programs: an interactive command processor, a text editor, a tape data set search routine, and a tape data set label and write routine.
I'm interested in seeing 'VIPER' and the 'VIP' gain in popularity. Good Luck.

Gene Saadi
Phoenix, AZ

Dear Sir;
I was pleasantly surprised to receive my first copy of the VIPER. I would like to suggest three corrections to the tape write routine on page three of the first issue:

Location	was	should be
0L44	B3	D3
0L45	F9	F8
0L49	F9	F8

The D3 instruction (SEP R3) affects the return to the calling routine. The two F9 instructions (ØRI) would cause incorrect tape writes.

Henry Woodward
Kensington, MD

(We have received a number of comments on the tape read/write routines. Has anyone implemented them in a user program? Will you share your experience?)

CHIP-8I

A Modification of CHIP-8 to Provide I/O Instructions

by Rick Simpson

One of the few deficiencies of CHIP-8 is that it does not provide instructions to handle the I/O ports. This modified version removes the BMMM instruction (GO TO ØMMM+VØ) and replaces it with three new instructions:

```
BØKK - Output KK to port
B1XØ - Output contents of VX to port
B1X1 - Read input from port and place in VX
```

No provisions are made to flag the outside world that a new byte has been placed in the output port. The Q-line can be used for this purpose by outputting a tone of duration Ø1.

As an example, consider the Nucleonic Products Co. CRT-1ØØØ Video Interface card. This provides a 16 line by 64 character display and requires 7-bit ASCII code input and a positive-going strobe pulse after the data is stable. A CHIP-8I subroutine to output the ASCII character contained in V3 would be:

```
B13Ø   Output byte in V3
64Ø1   Set V4=Ø1
F418   Set tone to V4 - output strobe
ØØEE   Return
```

The Q-line must be connected to the STR line on the CRT-1ØØØ.

Since almost every input device requires a flag input to signal the VIP that data is ready to be read, strobing provision is incorporated in the B1X1 instruction. The instruction waits for the EF4 line to go low and then reads in the byte. It then waits until the EF4 line returns to a high state before completing the instruction.

To interface an ASCII keyboard such as the Southwest Technical Products Co. model KBD5Z, the ASCII output lines are connected to the parallel input port and the negative-going keyboard strobe line is connected to both the SET and EF4 lines (pins K and L on the VIP I/O connector).

To read the next character from the keyboard and place it in V8, for example, you would use the CHIP-8I instruction B181. No other code is required.

To create your own copy of CHIP-8I, load CHIP-8 on your VIP and change the following locations:

```
          start ↘            3A                  E7
Ø1A4    E2 86 FA Ø1 6E AC E5 63 D4 ^45 F7 FA Ø1      corrected version
Ø1FØ2   3A F2 63 FA 3F F2 6B 3F F5 D4                per VIPER #5
                     D4
```

Save the modified version as CHIP-8I. This version will work identically to CHIP-8 except for the deletion of the BMMM instruction and the addition of the three new I/O instructions.

... Two byte Hex Display, con'd.

```
address     instruction     remarks

200         6210            V2=10
202         6320            V3=20
204         F129            set I for V1
206         D235            display low order digit of V1
208         0220            go to subroutine
20A         F1F5            V1 is source, V5 is result
20C         F529            set I for V5
20E         72F9            V2=V2-5 (back up display pointer)
210         D235            display low order digit of V5
212         1212            endless loop for end

220 45 A6 06 F6 F6 F6 F6 AD 45 A6 8D 56 D4 (subroutine)
```

end.

Modifying a Pixie-Verter for use with the VIP

There is no doubt that a good video modulator is the best device to use for displaying the VIP video output, but it is often handy to be able to feed the VIP output to a standard TV set, especially if you need a large-screen display. Also, when the Color Board becomes available, few of us will be willing to buy color monitors.

There are a large variety of RF modulators available, now that so many personal computers use video output. They range in price from $5 to $25. The one that has been around longest (it was available before there were any personal computers and was used by amateur TV operators) and is still the most widely available is the Pixie-verter, made by ATV Research, 13th and Broadway, Dakota City, NB, 68731. It costs about $7. The problem with the Pixie-Verter is that it was designed to use a -6V power supply. The following note describes how to modify the Pixie-Verter to use the VIP +5V supply.

First, observe all precautions in the Pixie-Verter assembly instructions to avoid violating FCC regulations. Then:

1. Omit R1 in the Pixie-Verter
2. Connect the VIP ground to the Pixie-Verter -6V terminal
3. Connect the VIP +5V supply to the Pixie-Verter +6V (ground)
4. Connect the VIP video output to Pixie-Verter video input at R2. Use a single, ungrounded wire.
5. Connect Pixie-Verter RF output (C6) to the VHF antenna terminal of your TV set. Use shielded cable and connect the shield braid to the Pixie-Verter -6V lead, not +6 as shown in the Pixie-Verter instructions.
6. Adjust the Pixie-Verter C3 and R2 for the best picture.

If you are going to ever need an RF modulator, buy it soon, as the FCC has announced that they intend to force all suppliers of RF modulators to withdraw their products because of possible interference problems.

Two-Byte Hex Display

A CHIP-8 Splinter

CHIP-8 provides facilities for doing hexadecimal arithmetic and allows you to convert a hex variable to a decimal representation spread over three bytes (FX33). The least significant digit of each byte can then be displayed using the FX29 and DXYN instructions. But CHIP-8 does not provide any way to display both digits of a variable in hex. The following subroutine provides this capability.

This routine takes the high (most significant) digit of variable X and stores that digit in the low (least significant) digit of variable Y. The subroutine can be stored anywhere in memory and is called by the CHIP-8 sequence:

```
2mmm        where mmm is the memory address of the
            subroutine
FxFy        where x and y are replaced by the variable
            numbers to be used
```

Thus the 2mmm calls the subroutine, and FxFy is really data passed to the subroutine - FxFy is never used as a program instruction.

R5 is used as the pointer to CHIP-8 instructions
R6 is a pointer to the variables to be modified
RD is used for temporary storage

```
machine language
instruction           assembly code        comments

45                    LDA R5               read first argument
A6                    PLO R6               store in R6.0
06                    LDN R6               get named variable
F6                    SHR                  shift it
F6                    SHR                  four
F6                    SHR                  times
F6                    SHR                  to the right
AD                    PLO RD               Put result in RD.0
45                    LDA R5               Get second argument
A6                    PLO R6               store in R6.0
8D                    GLO RD               get result back
56                    STR R6               store in result variable
D4                    SEP R4               all done - return
```

The following sample program in CHIP-8 displays the hex value of variable V1 at location V2, V3. V5 is used to store the high digit of V1 for display.

(go to p. 5)

COSMAC VIP

PRICE SCHEDULE — Effective September 1, 1978

RCA COSMAC VIP SYSTEM COMPONENTS

Type	Description	Optional User Price — $'s
CDP18S711	**VIP — Video Interface Processor** A completely assembled microcomputer. Features built-in cassette interface, video interface, 16-key keypad, 2K RAM, ROM operating system, CHIP-8 language and power supply. Output drives video monitor or rf modulator. Includes VIP-311 Instruction Manual with listing of 20 games.	249
VP-44 VP-45	**RAM On-Board Expansion Kits** Contain four type 2114 (VP-44) or four type 9131 (VP-45) RAM IC's for expanding the VIP on-board memory to 4K bytes. (Early kits used 9131 ICs - All others use 2114ICs)	36
VP-590	**VIP Color Board** — Available 10/78 Displays VIP output in color! Program control of four background colors and eight foreground colors in each of 64 picture areas of an 8 x 8 screen matrix. Higher resolution available under machine language control. CHIP-8C language adds color commands to CHIP-8 instruction set. Includes two sockets for VP-580 Expansion Keyboards.	69
VP-595	**VIP Simple Sound Board** — Available 10/78 Provides 256 different frequencies in place of VIP single-tone output. Ideal for use with VP-590 Color Board for simultaneous color and sound. Simple machine-language subroutine addition to CHIP-8 or CHIP-8C allows you to set the frequency and duration of the output tone. Great for simple music or sound effects! Includes speaker.	24
VP-550	**VIP Super Sound Board** — Available 11/78 Turn your VIP into a music synthesizer! Provides two independent sound channels (voices). Frequency, duration and amplitude envelope of each channel under program control. On-board tempo control. Provision for sync output for multi-track recording or slaving several VIP's for simultaneous play. Output drives audio preamp. Does not permit simultaneous video display.	49
VP-570	**VIP Memory Expansion Board** — Available 12/78 Adds 4K static RAM memory. Plugs into expansion connector. Jumper locates RAM in any of the first four 4K memory segments.	95
VP-580	**VIP Expansion Keyboard** — Available 11/78 Program your VIP for two-player video interaction games! 16-key keypad with cable connects to sockets provided on VP-590 Color Board or VP-585 Keyboard Interface Card. Instructions provided for use with CHIP-8 or CHIP-8C.	15

RCA

Type	Description	Optional User Price — $'s
VP-585	**VIP Keyboard Interface Board** — Available 11/78 Interfaces two VP-580 Expansion Keyboards directly to the VIP. Not required when VP-590 Color Board is used.	10
VP-560	**VIP EPROM Board** — Available 12/78 Allows two Intel 2716 EPROMs to be interfaced to the VIP. Has provisions for placing EPROMs anywhere in the VIP memory space. Can also re-allocate on-board RAM in VIP memory space.	34
VP-565	**VIP EPROM Programmer Board** — Available 12/78 Program Intel 2716 EPROMs with your VIP. Complete with software to program, copy, and verify EPROM. Features on-board generation of all required programming voltages. VP-560 required for use of EPROM in VIP memory space.	99
TC1210	**9" Video Monitor** An ideal, low-cost monochrome monitor for displaying the video output from your VIP — or other computers with video output.	175
TC1212	**12" Video Monitor** A large (74 sq. in. picture) monochrome monitor for use with your VIP or other computers with video output.	285
TC1217	**17" Video Monitor** A really BIG monochrome monitor for use with your VIP or other computers with video output. 148 sq. in. pictures. Ideal for two-player video interaction games or displays.	380
CDP18S731 CDP18S745	**RAM/IO Expansion Kits** Contain four type 9131 (18S731) or 2114 (18S745) RAM ICs plus other components for I/O expansion ports. Used only with 18S022 VIP kits.	69
VP-700	**VIP Tiny BASIC ROM Board** — Available early '79 Run Tiny BASIC on your VIP! All BASIC code is stored in ROM — no RAM is used for the interpreter. Includes Tiny BASIC programming manual. Requires separate external ASCII-coded alphanumeric keyboard.	29
VP-710	**VIP Game Manual** More exciting games for your VIP! Includes Blackjack, Biorythm, Pinball, Bowling and 10 others. Complete instructions and program listings are supplied for each game.	10
VIP-311	**VIP Instruction Manual**	5
VIP-320	**VIP User Guide Manual**	5
MPM-201B	**CDP1802 User Manual**	5

Prices and specifications subject to change without notice.

RCA VIP Marketing
New Holland Avenue, Lancaster, PA 17604
Phone: (717) 397-7661

Information furnished by RCA is believed to be accurate and reliable. However, no responsibility is assumed by RCA for its use; nor for any infringements of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of RCA.

Printed in U.S.A./**9-78**
VIPP-2U

Order Form

COSMAC VIP

RCA encourages Computer and Electronics Dealers to stock VIP Products in order to serve you. However, in the event local sources are not available, RCA will accept direct orders. Please use this order form. Expect delivery of stock items within 3 to 4 weeks. A Certified Check or Money Order will expedite delivery. Purchases may also be charged to your VISA/BankAmericard or Master Charge account.

Complete this form, enclose a check or money order for the full amount (or add your signature and account number) and mail to:

RCA VIP Customer Service
New Holland Avenue
Lancaster, PA 17604

Please send me the RCA COSMAC VIP System Components indicated.

		Unit Price	Quantity	Total Price
CDP18S711	VIP Video Interface Processor	$249		
CDP18S731	RAM/IO Expansion Kit (Type 9131's)	69		
CDP18S745	RAM/IO Expansion Kit (Type 2114's)	69		
TC1210	9" Video Monitor	175		
TC1212	12" Video Monitor	285		
TC1217	17" Video Monitor	380		
VP-44	RAM "On-Board" Expansion Kit (Type 2114's)	36		
VP-45	RAM "On-Board" Expansion Kit (Type 9131's)	36		
VP-550	Super Sound Board	49		
VP-560	EPROM Board	34		
VP-565	EPROM Programmer Board	99		
VP-570	Memory Expansion Board	95		
VP-580	Expansion Keyboard	15		
VP-585	Keyboard Interface Board	10		
VP-590	Color Board	69		
VP-595	Simple Sound Board	24		
VP-700	Tiny BASIC ROM Board	29		
VP-710	VIP Game Manual	10		
VIP-311	VIP Instruction Manual	5		
VIP-320	VIP User Guide Manual	5		
MPM-201B	CDP1802 User Manual	5		
			Subtotal	
			Add: Applicable State and Local Taxes	
			Total	

I enclose a □ check or □ money order or, charge my □ VISA/BankAmericard □ Master Charge

Credit Card Account No. ______________________ Master Charge Interbank No. ____________

Signature (required for charge account orders) ______________________

Name (please type or print clearly) ______________________

Street Address ______________________

City ______________ State ______________ Zip Code ______________

Telephone: Area Code (   ) ______________________

Make checks payable to RCA Corp. Prices and Specifications are subject to change without notice.

RCA

Printed in U.S.A./**9-78**
VIPP-3

Trademark(s) ® Registered
Marca(s) Registrada(s)

2 Page Display for CHIP-8

by Andy Modla and Jef Winsor

There has been significant interest expressed in increasing the resolution of the video display on the VIP. The horizontal resolution of the display is fixed by hardware at 64 bits across the screen. It would require a hardware change to increase this amount. The vertical resolution, however, is variable and may be controlled by software, or the lack of it, in the display refresh routine. To generate a display of higher or lower resolution than that of the operating system of the VIP a new display refresh (interrupt) routine and a few register initialization instructions is all that is required. To implement this change and maintain compatibility between CHIP-8 and the new resolution display is more involved. The following listings may be used to generate a 2 page, 512 byte, display for CHIP-8 programs.

The most fundamental change is the interrupt or display refresh routine. The new interrupt routine begins at 0203. It includes the tone and timer control contained in the operating system display routine which is also normally used by CHIP-8. The routine is entered at location 0206 and exited at location 0205. This technique leaves the interrupt program counter (R1) pointing at the entry point when the next interrupt occurs.

The highest page in memory is usually the display page and the stack and variable storage for CHIP-8 is usually on the second highest page. The 2 page display system presented here uses the second highest page as the extra display memory so the stack, variables, and CHIP-8 work space must be moved 1 page lower than normal. Changing byte 0003 in the CHIP-8 code to 02 takes care of this.

Changing byte 000A to 02 and 000D to 06 initializes R1 to point to the new interrupt routine.

Contrary to popular belief the first CHIP-8 instruction is actually located at 01FC. Before the CHIP-8 program counter gets to 0200 the display page must be cleared to all zeros and the TV display turned on. The new display requires an additional CHIP-8 instruction prior to turning on the TV interface chip. Changing location 0018 to FA arranges for this extra instruction.

The extra preliminary CHIP-8 instruction at 01FA sets the start of the display page to the second highest page by calling the machine language routine at 0245. The instruction at 01FC clears the display memory by a new clear routine for two pages at 0230.

Location 0200 must still contain a CHIP-8 instruction so a branch is used to bypass the new interrupt and other related routines. This branch could go as low as 024A but 0260 is a nice round hexadecimal number.

CHIP-8 normally produces a wrap-around effect on the screen when an object moves more than 32 bit positions below the first display line. To increase this to a 64 bit vertical limit, corresponding to a 2 page display, locations 007E; 0084, 85; 00D7, D8; 00E0-ED; and 00FC-FF must be changed as listed.

The last point to consider is the amount of memory being used on board the VIP. When using this display system in a 2K VIP location 00EB must be set to 08. For a 3K VIP 00EB should be 0C and 10 in a 4K VIP.

```
TWO PAGE DISPLAY MODIFICATION TO CHIP-8 INTERPRETER

MAKE THE FOLLOWING CHANGES TO CHIP-8:

ADDR DATA
0003 02
000A 02
000D 06
0018 FA
007E 3F
0084 30E0
00D7 30E6
00E0 9B7C 00BC 3086 9C7C 00BC FB08 30FC
00FC 3AB3 30D9
01FA 0245 0230

NOTE: LOCATION 00EB IS SET BASED ON MEMORY AVAILABLE
      IN YOUR VIP AS FOLLOWS:
      4K    10
      3K    0C
      2K    08
```

```
0200 1260 GO 0260
0202 017A INTERRUPT SUBROUTINE
0204 4270
0206 2278
0208 2252
020A C419
020C F800
020E A09B
0210 FA0E
0212 B0E2
0214 E280
0216 E220
0218 A0E2
021A 3C15
021C 80E2
021E 20A0
0220 341C
0222 9832
0224 29AB
0226 2B8B
0228 B888
022A 3203
022C 7B28
022E 3004
0230 F802 CLEAR TV DISPLAY 2 PAGES
0232 AE9B
0234 BFF8
0236 00AF
0238 F800
023A 5F1F
023C 8F3A
023E 382E
0240 8E3A
0242 38D4
0244 019B ADJUST TV MEMORY POINTER
0246 FF01
0248 BBD4

TEST PROGRAM
0260 6000 V0=00
0262 6100 V1=00
0264 A273 I=0273
0266 D014 SHOW 4MI@V0V1
0268 7004 V0+04
026A 7104 V1+04
026C 3140 SKIP;V1 EQ 40
026E 1266 GO 0266
0270 1270 GO 0270
0272 01F0
0274 F0F0
0276 F0D4
```

YOUR CHIP-8 PROGRAMS START AT 0260.
THE ABOVE CODE FROM 0200 TO 0249 MUST BE INCLUDED
TO USE THE TWO PAGE DISPLAY MODIFICATION.

CODING FORM FOR RCA COSMAC PROGRAMS

Title J. W. Wentworth's Interpretation of VIP Operating System

MEMORY PAGE 80 Programmer

Address for first Fetch cycle in RUN mode is 8000 because RUN switch resets FF U6A. R0 is both PC and X Register at start.

00	F8	
01	80	START INITIALIZATION
02	B2	Set R2 to 8008
03	F8	
04	08	
05	A2	
06	E2	2 → X
07	D2	2 → P
08	64	Output 00 to Bus (signal on bus is ignored, but operation of N2 line sets U6A).
09	00	
0A	62	Output 0C to Keyboard Latch
0B	0C	DETERMINE HIGHEST MEM. PAGE
0C	F8	
0D	FF	Set R1 to 0FFF (highest RAM address that may be used in any VIP system)
0E	A1	
0F	F8	
10	0F	
11	B1	
12	F8	
13	AA	Store AA via R1
14	51	
15	01	Load via R1, XOR with AA (i.e., check to see if storage was successful)
16	FB	
17	AA	
18	32	If D = 0 (storage successful), go to 8022
19	22	
1A	91	Get R1.1, subtract 04
1B	FF	
1C	04	
1D	3B	Branch if minus to 8022
1E	22	
1F	B1	Put in R1.1,

20	30	Branch back to 8012 (Finds highest page and sets R1 to its highest address)
21	12	
22	36	If EF3 = 1 (meaning that "C" is pressed) branch to 8028
23	28	
24	90	If "C" is not pressed, Get R0.1(=00) and put in R0.0, thereby setting R0 to 0000, then 0 → X, 0 → P (Go to M. L. 0000 to run a program)
25	A0	
26	E0	
27	D0	ERASE DISPLAY PAGE FROM OXFF TO 0XBO (X is highest page
28	E1	1 → X
29	F8	Load Imm. 00, store via X and decrement
2A	00	
2B	73	
2C	81	Get R1.0, XOR with AF
2D	FB	
2E	AF	
2F	3A	If D ≠ 0, branch back to 8029
30	29	STORE CONTENTS OF REGISTERS
31	F8	Store D2 (as a RETURN instruction) in cell 0XAF and decrement
32	D2	
33	73	
34	F8	
35	9F	Store 9F via R1 (in cell 0XAE) (a 2-byte subroutine has now been formed at 0XAE)
36	51	
37	81	Set R0 to same as R1 (=0XAE)
38	A0	
39	91	
3A	B0	
3B	F8	Set R1.0 to CF (R1 now OXCF)
3C	CF	
3D	A1	
3E	D0	0 → P (calls subroutine at 0XAE to get RF.1 and returns)
3F	73	Store via X (=R1) and decrement

CODING FORM FOR RCA COSMAC PROGRAMS

Title J. W. Wentworth's Interpretation of VIP Operating System

MEMORY PAGE 80 Programmer

40	20	Decrement R0 twice (return to 0XAE)
41	20	
42	40	Load via R0 and advance
43	FF	Subtract 01, cancel advance in R0, and store via R0 (i.e., in 0XAE)
44	01	
45	20	
46	50	
47	FB	XOR with 82--if the result is not zero, branch back to 803E. (When the program "falls through" this point, register contents from RF.1 down through R3.0 will have been stored.)
48	82	
49	3A	
4A	3E	

INITIALIZE REGISTERS FOR REST OF OPERATING SYSTEM

4B	92	Set R3 to 8051
4C	B3	
4D	F8	
4E	51	
4F	A3	
50	D3	3 → P (R3 becomes PC)
51	90	
52	B2	0X → R2.1
53	BB	0X → RB.1
54	BD	0X → RD.1
55	F8	
56	81	81 → R1.1
57	B1	81 → R4.1
58	B4	81 → R5.1
59	B5	81 → R7.1
5A	B7	81 → RA.1
5B	BA	81 → RC.1
5C	BC	
5D	F8	R1 = 8146 (points to Interrupt routine)
5E	46	
5F	A1	

60	F8	R2 = 0XAF (stack pointer)
61	AF	
62	A2	
63	F8	R4 = 81DD (points to display control routine)
64	DD	
65	A4	
66	F8	R5 = 81C6 (points to subroutine for 5-byte display of hex digits)
67	C6	
68	A5	
69	F8	R7 = 81BA (points to subroutine for assembly of two keyboard entries into a byte)
6A	BA	
6B	A7	
6C	F8	RC = 81A1 (points to keyboard debounce routine)
6D	A1	
6E	AC	
6F	E2	2 → X (designate stack pointer as X)
70	69	Enable display and interrupts--see Interru Routine at 8146.

ENTER ADDRESS

71	DC	Call debounce routine at 81A1 (for "C")
72	D7	Enter high address byte via subroutines at 81BA, 81BB and 81C1
73	D7	
74	D7	
75	B6	Put high address byte in R6.1
76	D7	Enter low address byte via subroutines at 81BA, 81BB and 81C1
77	D7	
78	D7	
79	A6	Put low address byte in R6.0
7A	D4	Call display routine at 81DD

ENTER AND DECODE CONTROL DIGIT

7B	DC	Call keyboard scan routine at 8195
7C	BE	Put control digit in RE.1
7D	32	If control digit is 0 (Memory Write), branch to 80F4
7E	F4	
7F	FB	

CODING FORM FOR RCA COSMAC PROGRAMS

Title J. W. Wentworth's Interpretation of VIP Operating System

MEMORY PAGE 80 Programmer

80	0A	If control digit is A (Memory Read), branch to 80EF
81	32	
82	EF	
83	DC	Call keyboard scan routine at 8195 for entry of number of pages
84	AE	Put entry (no. of pages) in RE.0
85	22	Decrement stack pointer (compensation for unwanted advance of next step)
86	61	Turn TV display off (advances stack pointer)
87	9E	Get E.1 (control digit)
88	FB	If control digit is B (Tape Read), branch to 80C2
89	0B	
8A	32	
8B	C2	
8C	9E	Get control digit (from RE.1)
8D	FB	If control digit is not F (Tape Write), branch to 8F. (Forms endless loop, requiring RESET--invalid code digit.)
8E	0F	
8F	3A	
90	8F	

TAPE WRITE ROUTINE

91	F8	Set RC.0 to 6F (RC now points to 816F)
92	6F	
93	AC	
94	F8	40 → R9.1 (Timer for tape leader)
95	40	
96	B9	
97	93	80 → D
98	F6	Shift Right (assures that DF = 0)
99	DC	Do subroutine at 816F (write one cycle of tape signal)
9A	29	Decrement R9
9B	99	Get R9.1; if not zero, branch back to 8097. Program "falls through" this point after leader has been written--approximately 4 seconds.
9C	3A	
9D	97	
9E	F8	10 → R7.0 (Presets parity counter to an even number)
9F	10	

A0	A7	
A1	F8	08 → R9.0 (Preset bit counter)
A2	08	
A3	A9	
A4	46	Load via R6 (contents of entered address) and advance
A5	B7	Put in R7.1
A6	93	80 → D
A7	FE	Shift left (Sets DF = 1)
A8	DC	Do subroutine at 816F (writes start bit)
A9	86	Get R6.0 (address of next byte)
AA	3A	If D ≠ 0, branch to 80AD
AB	AD	
AC	2E	Decrement RE (No. of pages)
AD	97	Get R7.1 (byte being recorded), shift right and return to R7.1 (DF set to 1 or 0 depending on bit)
AE	F6	
AF	B7	
B0	DC	Do subroutine at 816F (record bit)
B1	29	Decrement R9 (bit counter)
B2	89	If R9.0 is not zero, branch back to 80AD. Program "falls through" this point when all 8 bits have been recorded
B3	3A	
B4	AD	
B5	17	Increment R7 (establishes odd parity)
B6	87	Get R7.0, shift right (moves parity bit to DF)
B7	F6	
B8	DC	Do subroutine at 816F (record parity bit)
B9	8E	Get RE (no. of pages remaining); if not zero, branch back to 809E. Program "falls through" when the last byte of the final page has been written.
BA	3A	
BB	9E	
BC	DC	Do subroutine at 816F (repeats last parity bit)
BD	69	Turn display on.
BE	26	Decrement R6 (points to address of last byte recorded)
BF	D4	Do subroutine at 81DD (displays address and contents of last byte recorded)

CODING FORM FOR RCA COSMAC PROGRAMS

Title J. W. Wentworth's Interpretation of VIP Operating System

MEMORY PAGE 80 Programmer

C0	30	Go to 80C0 (endless loop to await RESET)
C1	C0	
		TAPE READ ROUTINE
C2	F8	Set RC.0 to 83 (RC now points to 8183)
C3	83	
C4	AC	
C5	F8	0A → R9.1 (Timer for leader search)
C6	0A	
C7	B9	
C8	DC	Do subroutine at 8183 (Read leader)
C9	33	Branch if DF=1 to 80C5 (re-initiate the search for a proper leader if a "1" bit is detected early in search)
CA	C5	
CB	29	Decrement R9
CC	99	Get R9.1; if not zero, branch to 80C8 (Reads leader--must have enough consecutive "0's" to allow 9.1 to reach zero.)
CD	3A	
CE	C8	
CF	DC	Call subroutine at 8183 (continue reading)
D0	3B	If DF = 0, branch to 80CF (loops until first start bit is detected).
D1	CF	
D2	F8	09 → R9.0 (to count bits plus parity bit)
D3	09	
D4	A9	09 → R7.0 (parity counter pre-set with an odd number)
D5	A7	
D6	97	Get R7.1, ring shift right, put back in R7.1 (Moves DF into MSB position)
D7	76	
D8	B7	
D9	29	Decrement R9
DA	DC	Call subroutine at 8183 for next bit
DB	89	Get R9.0, branch if not zero to 80D6. (Assembles 8 bits into a byte, reads parity bit on final pass)
DC	3A	
DD	D6	
DE	87	Get R7.0 (parity counter), shift right (Places LSB in DF)
DF	F6	

E0	33	If DF = 1 (i.e., if no parity error), branch to 80E3
E1	E3	
E2	7B	Set Q (turns on tone generator)
E3	97	Get R7.1 (assembled byte)
E4	56	Store via R6
E5	16	Increment R6
E6	86	Get R6.0; if not zero, branch back to 80CF to read in next byte.
E7	3A	
E8	CF	
E9	2E	Decrement RE (No. of pages)
EA	8E	Get RE.0; if not zero, branch back to 80CF to commence reading another page. Program "falls through" when full no. of pages has been read in.
EB	3A	
EC	CF	
ED	30	Branch to 80BD (to display final address and contents, then stop to await RESET)
EE	BD	
		MEMORY READ ROUTINE
EF	DC	Call keyboard scan routine at 8195 (any key)
F0	16	Increment R6
F1	D4	Call display subroutine at 81DD
F2	30	Branch back to 80EF to read next M. L.
F3	EF	
		MEMORY WRITE ROUTINE
F4	D7	Call subroutines at 81BA, 81BB and 81C1 to enter two hex digits and assemble them into abyte.
F5	D7	
F6	D7	
F7	56	Store via R6
F8	D4	Call display subroutine at 81DD
F9	16	Increment R6
FA	30	Branch to 80F4 (for repeated cycles)
FB	F4	
FC	00	fillers
FD	00	
FE	00	
FF	00	

CODING FORM FOR RCA COSMAC PROGRAMS

Title J. W. Wentworth's Interpretation of VIP Operating System

MEMORY PAGE 81 Programmer

00	30	START ADDRESSES FOR HEX DIGIT DISPLAYS	20	F0	Start "5"
01	39		21	80	
02	22		22	F0	Start "2"
03	2A		23	10	
04	3E		24	F0	Start "6"
05	20		25	80	
06	24		26	F0	Start "8"
07	34		27	90	
08	26		28	F0	Start "9"
09	28		29	90	
0A	2E		2A	F0	Start "3"
0B	18		2B	10	
0C	14		2C	F0	
0D	1C		2D	10	
0E	10		2E	F0	Start "A"
0F	12	5-BYTE DIGIT DISPLAY PATTERNS	2F	90	
10	F0	Start "E"	30	F0	Start "0"
11	80		31	90	
12	F0	Start "F"	32	90	
13	80		33	90	
14	F0	Start "C"	34	F0	Start "7"
15	80		35	10	
16	80		36	10	
17	80		37	10	
18	F0	Start "B"	38	10	
19	50		39	60	Start "1"
1A	70		3A	20	
1B	50		3B	20	
1C	F0	Start "D"	3C	20	
1D	50		3D	70	
1E	50		3E	A0	Start "4"
1F	50		3F	A0	

CODING FORM FOR RCA COSMAC PROGRAMS

Title J. W. Wentworth's Interpretation of VIP Operating System

MEMORY PAGE 81 Programmer

40	F0	Last 3 bytes of 5-byte display pattern for "4"
41	20	
42	20	INTERRUPT ROUTINE (DISPLAY REFRESH)
43	7A	Reset Q
44	42	Load by R2 (restores D) and advance
45	70	Return and enable interrupt
46	22	ENTRY POINT--Decrement R2 (stackpointer)
47	78	Save (T → Stack)
48	22	Decrement stack pointer
49	52	Store D in stack (NOTE: DF is not altered by this routine)
4A	C4	No Operation (3-cycle time compensation)
4B	19	Increment R9 (Random number pointer)
4C	F8	Set R0 (DMA pointer) to 0X00
4D	00	
4E	A0	
4F	9B	
50	B0	
51	E2	2 → X (twice--time fillers only)
52	E2	
53	80	Get R0.0
54	E2	2 → X (time filler) ← 8 DMA cycles occur here
55	E2	2 → X (kill time)
56	20	Restore start address (decrement R0 to cover case of last four lines in display when R0.1 advances to 0X + 1)
57	A0	← 8 DMA cycles occur here
58	E2	2 → X (kill time)
59	20	Restore start address
5A	A0	← 8 DMA cycles occur here
5B	E2	2 → X (Kill time)
5C	20	Restore start address
5D	A0	← 8 DMA cycles occur here
5E	3C	If EF1 = 0, return to 8153. (R0.0 will have advanced by 8 thru DMA action; program "falls through" after final line of display.)
5F	53	

60	98	Get R8.1 (timer); if zero, branch ahead to 8167
61	32	
62	67	
63	AB	Put D (contents of R8.1) in B.0, decrement B, get B.0 and put in R8.1. (This brief routine decrements R8.1 without disturbing the contents of DF)
64	2B	
65	8B	
66	B8	
67	88	Get R8.0 (tone timer)
68	32	If D = 0, branch to exit via "Reset Q" instruction at 8143
69	43	
6A	7B	(If D is not zero), Set Q, decrement R8 and branch to exit at 8144 (bypassing "Reset Q" instruction)
6B	28	
6C	30	
6D	44	SUBROUTINE FOR EACH CYCLE OF TAPE WRITE SIGNAL (At entry, DF indicates if "0" or "1 is to be written)
6E	D3	Return to main pgm.
6F	F8	0A → D (timing constant for "0" cycle)
70	0A	
71	3B	If DF = 0, branch to 8176
72	76	
73	F8	20 → D (timing constant for "1" cycle)
74	20	
75	17	Increment R7 (parity counter)
76	7B	Set Q (Output of Q is tape write signal)
77	BF	Put D (0A or 20) in RF.1
78	FF	Subtract 01; if result is not zero, branck back to 8178. (This is a delay loop to establish duration of a half cycle--ten loops for "0", thirty-two loops for "1".
79	01	
7A	3A	
7B	78	
7C	39	If Q = 0 (i.e., a full cycle completed) branch to 816E for exit to main prograr
7D	6E	
7E	7A	Reset Q
7F	9F	Get RF.1 (contents 0A or 20)

CODING FORM FOR RCA COSMAC PROGRAMS

Title J. W. Wentworth's Interpretation of VIP Operating System

MEMORY PAGE 81 Programmer

80	30	Branch to 8178
81	78	

SUBROUTINE FOR READING EACH CYCLE OF TAPE PLAYBACK SIGNAL

82	D3	Return to main program
83	F8	ENTRY POINT-- Load Immediate 10 (A number larger than the 0A but smaller than the 20 used in recording.)
84	10	
85	3D	If EF2 = 0, branch to 8185 (Wait for start of tape signal)
86	85	
87	3D	If EF2 = 0, branch to 818F (EF2 is "1" when this instruction is first encountered)
88	8F	
89	FF	Subtract immediate 01; if D ≠ 0, branch to 8187. (If the program "falls through" this point because D reaches zero before EF2 = 0, it means bit in playback signal is a "1".)
8A	01	
8B	3A	
8C	87	
8D	17	Increment R7 (Parity counter)
8E	9C	81 → D
8F	FE	Shift left (DF = 1 if bit was 1, DF = 0 if bit was 0)
90	35	If EF2 = 1, branch to 8190 (Waits for end of high half cycle in event bit was "1")
91	90	
92	30	Branch to 8182 for exit to main program
93	82	

KEYBOARD SCANNING SUBROUTINE

94	D3	Return to main program
95	E2	2 → X (this subroutine uses the stack)
96	9C	81 → RF.0
97	AF	
98	2F	Decrement RF
99	22	Decrement stack pointer
9A	8F	Get RF.0, store in stack
9B	52	
9C	62	Output to keyboard latch; advance stack pointer
9D	E2	2 → X twice (kill time to allow for propagation delays thru latch and keyboard to EF3 input)
9E	E2	
9F	3E	If EF3 = 0 (addressed key not pressed), branch to 8198
A0	98	
A1	F8	(ALTERNATIVE ENTRY POINT FOR KEY DE-BOUNCE ONLY) 04 → R8.0 (De-bounce timer and tone control).
A2	04	
A3	A8	
A4	88	Get R8.0; if not zero, branch back to 81A4. (Wait for 4 TV fields-- de-bounce delay.)
A5	3A	
A6	A4	
A7	F8	04 → R8.0 (additional timing cycle for de-bouncing of end of key-press operation)
A8	04	
A9	A8	
AA	36	If EF3 = 1 (key matching digit in keyboard latch still pressed), branch to 81A7
AB	A7	
AC	88	Get R8.0 (filler--no useful function)
AD	31	If Q = 1 (R8.0 timer still running) branch to 81AA ("Falls through"4 TV fields after last detection of key press)
AE	AA	
AF	8F	Get RF.0 (counter used for scanning)
B0	FA	AND with 0F (save 2nd digit only)
B1	0F	
B2	52	Store in stack (key entry is also in D upon exit from this subroutine)
B3	30	Branch to 8194 for exit to main program
B4	94	
B5	00	fillers
B6	00	
B7	00	
B8	00	

SUBROUTINE FOR ASSEMBLY OF TWC KEYBOARD ENTRIES INTO A BYTE

B9	D3	Return to main program
BA	DC	Call subroutine at 8195 (keyboard scan)
BB	FE	Shift left 4 times
BC	FE	
BD	FE	
BE	FE	
BF	AE	Put in RE.0 (high digit)

CODING FORM FOR RCA COSMAC PROGRAMS

- Title J. W. Wentworth's Interpretation of VIP Operating System

MEMORY PAGE 81 Programmer

C0	DC	Call keyboard scan subroutine at 8195
C1	8E	Get RE.0 (high digit in proper position)
C2	F1	OR with top of stack (latest entry)
C3	30	Branch to 81B9 for exit to main program (with assembled byte in D)
C4	B9	SUBROUTINE FOR 5-BYTE DISPLAY OF HEX DIGITS
C5	D4	Return to control subroutine
C6	AA	Put digit to be displayed in RA.0 (A=810H) H=hex digit
C7	0A	Load via A (start address)
C8	AA	Put in RA.0 (RA now points to start address)
C9	F8	05 → RF.0 (byte counter)
CA	05	
CB	AF	
CC	4A	Load by A and advance (display byte)
CD	5D	Store via RD (which points to destination address on display page)
CE	8D	Get RD.0, add 08 and replace in RD.0 (prepare for next line of digit display)
CF	FC	
D0	08	
D1	AD	
D2	2F	Decrement RF
D3	8F	Get RF.0; if not zero, branch back to 81CC. (Program "falls through" this point when 5 bytes have been written)
D4	3A	
D5	CC	
D6	8D	Get RD.0, add D9, put back in RD.0 (at exit, this subroutine leaves RD pointing to a new location on the display page one byte to the right of the start address of the digit just written.).
D7	FC	
D8	D9	
D9	AD	
DA	30	Branch to 81C5 for exit
DB	C5	SUBROUTINE TO CONTROL DISPLAY OF ADDRESS AND M.L. CONTENTS
DC	D3	Return to m ain program
DD	22	Decrement stack pointer
DE	06	Load by R6 (byte at addressed location)
DF	73	Store in stack and decrement stack pointer

E0	86	Get R6.0 (low address byte), store in stack and decrement
E1	73	
E2	96	Get R6.1 (high address byte), store in stack (no decrement)
E3	52	
E4	F8	06 → RE.0 (special control byte)
E5	06	
E6	AE	
E7	F8	D8 → RD.0 (start address for high address byte display on display page)
E8	D8	
E9	AD	
EA	02	Load from stack
EB	F6	Shift right 4 times (put MSD in LSD position)
EC	F6	
ED	F6	
EE	F6	
EF	D5	Call digit display SR at 81C6
F0	42	Load from stack and advance; AND with 0F to save LSD only
F1	FA	
F2	0F	
F3	D5	Call digit display SR at 81C6
F4	8E	Get RE.0, shift right and put back in RE.0
F5	F6	
F6	AE	
F7	32	If D= 0 (all digits written on display page), branch to 81DC for exit
F8	DC	
F9	3B	If DF = 0 (after writing high address byte only), branch to 81EA
FA	EA	
FB	1D	(If DF ≠ 0) increment RD twice (provide space between address display and M.L. contents display)
FC	1D	
FD	30	Branch to 81EA
FE	EA	
FF	01	filler and check byte

SUMMARY OF REGISTER FUNCTIONS IN VIP OPERATING SYSTEM

R0 DMA Pointer (Pre-set at start of Interrupt Routine)

R1 Interrupt Routine Program Counter (Pre-set to 8146)

R2 Stack Pointer (Pre-set to start stack at 0XAF where X is highest page)

R3 Main Program Counter

R4 PC for display control subroutine (Pre-set to 81DD)

R5 PC for 5-byte digit display subroutine (Pre-set to 81C6)

R6 Pointer for addressed byte as entered by operator (later incremented during Memory Read and Memory Write routines)

R7 PC for subroutine commencing at 81BA for assembly of two keyboard entries into a byte. (Routine must be called three times in succession because keyboard scanning routine returns control to main program, not to this subroutine.)

R7. 1 stores bytes for processing in tape write and read routines.

R7. 0 is parity counter for tape write and read routines.

R8 R8. 1 is a timer used in Chip 8 programs. Although not used directly by the Operating System, it is tested during each Interrupt cycle (TV scan) and decremented once if it is not already at zero.

R8. 0 is also a timer that is tested and decremented once per TV scan if it is not zero. In the Interrupt routine, this timer is related to the Q output in such a way that a tone is developed when R8 is not at zero. In the Operating System, this timer controls de-bounce delays and acoustic feedback for the keyboard.

R9 Special pointer used by the random number generator in the CHIP 8 Interpreter, incremented once per TV scan by the Interrupt Routine in the Operating System

R9. 1 serves as a timer for the tape leader in Tape Write and Tape Read routines (during which the display is turned off, so there are no interrupt cycles).

R9. 0 serves as a bit counter in Tape Write and Tape Read routines.

RA Data pointer used in 5-byte hex digit display subroutine--points to source data for display. (Pre-set high byte to 81)

RB RB. 1 stores display page number (Pre-set to 0X) for use during Interrupt Routine.

RB. 0 used during Interrupt Routine to accomplish decrementing of R8. 1 in a manner that does not disturb Data Flag.

SUMMARY OF REGISTER FUNCTIONS (CONTINUED)

RC PC for keyboard scanning subroutine at 8195 (with alternative entry point at 81A1 for key de-bouncing only). Also PC for subroutine at 816F that forms each cycle of Tape Write signal, and PC for another subroutine at 8183 that interprets each cycle of Tape Read signal.

RD Destination pointer for data to be displayed on display page. (Pre-set high byte to 0X).

RE RE.1 stores Control Digit (A, 0, B or F) used to distinguish between Memory Read, Memory Write, Tape Read and Tape Write operations.

RE.0 stores the most-significant-digit in the subroutine which combines two keyboard entries into a byte. Stores a special control byte in the display control subroutine. Stores the number of pages to be read or recorded for Tape Read and Tape Write routines.

RF RF.1 stores timer constant for Tape Write subroutine.

RF.0 used for keyboard scanning, also as a byte counter for the 5-byte display subroutine.

COMMENTS ON GENERAL-PURPOSE USE OF CERTAIN SUBROUTINES

Several of the subroutines in the VIP Operating System may be useful in machine-language programs or in machine-language subroutines within CHIP 8 programs. The following comments describe some of the possibilities.

For keyboard scanning to enter a hex digit, pre-set RC to 8195, call with DC instruction. At entry, stack pointer (R2) must point to an available cell, and there must be at least one-cell "head room" for the stack. An E2 instruction is included in this routine, but there is no provision for restoring an original value for X. At exit, the subroutine returns via a D3 instruction (leaving RC pointing to 8195), and the hex digit entered from the keyboard is in both D and the top of the stack. RF and R8.0 are altered.

For key de-bounce only, pre-set RC to 81A1, call with DC. This cause entry to the keyboard scanning routine after the actual scanning process. At exit via D3, RC will be left pointing at 8195 and the LSD (hex) of RF.0 will be in both D and top of the stack. (This digit may, of course, be ignored, but the user should be aware that both D and the stack are altered, as is R8.0.)

GENERAL-PURPOSE USE OF CERTAIN SUBROUTINES (Continued)

For assembly of two successive keyboard entries into a byte (high digit entered first), pre-set RC to 8195, pre-set any available register (N) to 81BA, and call by three consecutive DN instructions. The three calls are necessary because the keyboard scan routine called within this one returns via D3, not DN. At final exit (via D3), RN points to 81BA, assembled byte is in D, and least-significant hex digit is at the top of stack; RF, RE. 0 and R8. 0 are altered.

For 5-byte display of hex digits, first set up a display-control subroutine using R4 as PC. (This is necessary because the 5-byte display SR in the Operating System returns via D4, not D3.) The control subroutine may be nothing more than a DN instruction (where N is PC for the 5-byte display SR) followed by D3, but in many cases it is helpful to use a subroutine to make initial settings for RA or RD. Pre-set RD to the destination address on the display page for the first byte of the display, pre-set the high byte of RA to 81, pre-set any available register (N) to 81C6, place the hex digit to be displayed in D, and call with a DN instruction. (The subroutine itself will determine the start address for the five display bytes and place this address in RA.) At exit, RN will point to 81C6, RA will point to an address five greater than the start address of the source data for the display just written, and RD will have been incremented (such that the next call of this SR without changing RD will cause another digit to be written one byte position to the right on the display page). This subroutine alters RF. 0.

For general-purpose use of 5-byte display subroutine (not necessarily of hex digits), set up a control subroutine using R4 as PC (same as above), pre-set RA to first of five consecutive bytes in memory to be displayed one above the other (these could be on any page), pre-set RD to destination address of first byte on display page, pre-set any available register (N) to 81C9, and call via DN. Exit conditions are the same as those cited for the 5-byte hex digit display. (The writer has found this general-purpose 5-byte display subroutine very useful in writing labels for simulator programs.)