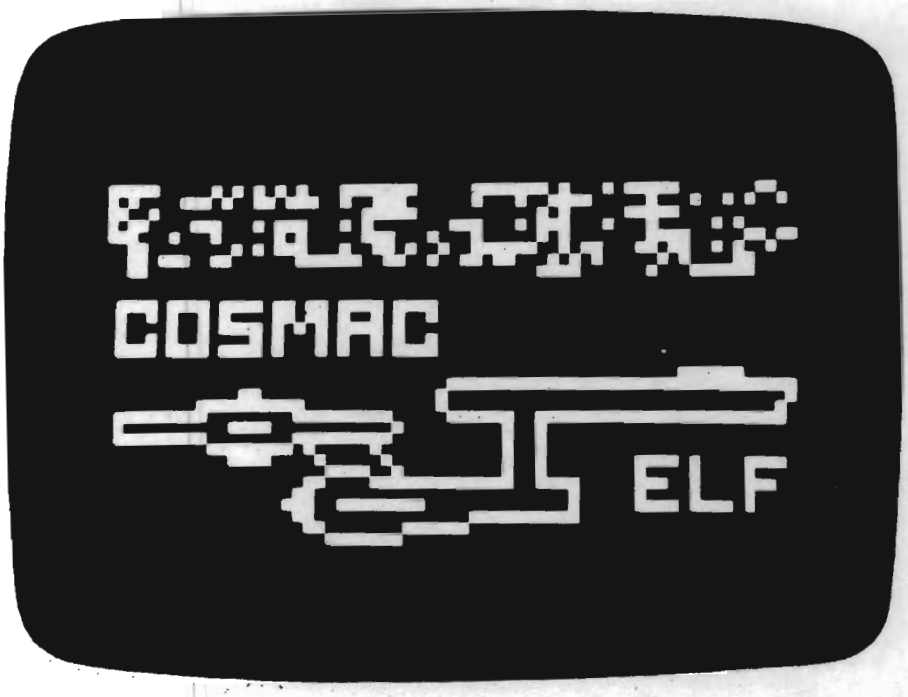
Popular Electronics
JULY 1977




PE TESTED
BREAKTHROUGH PROJECT

BY JOSEPH A. WEISBECKER

PART IV:

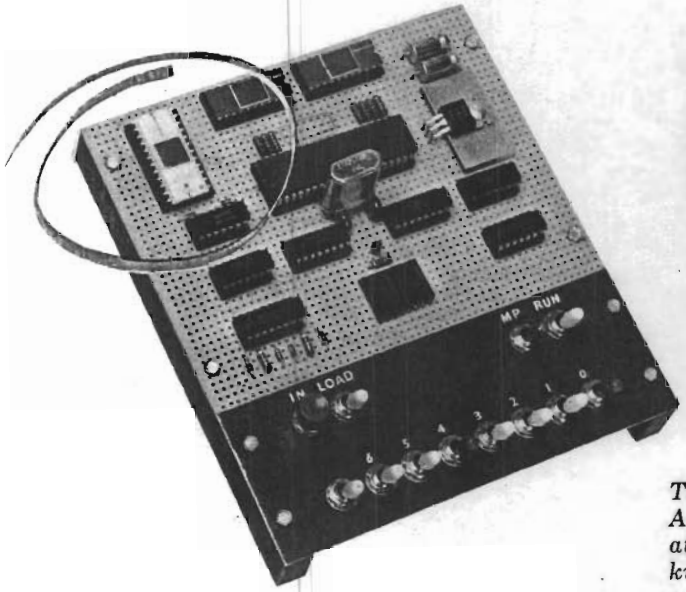
# Build the PIXIE Graphic Display

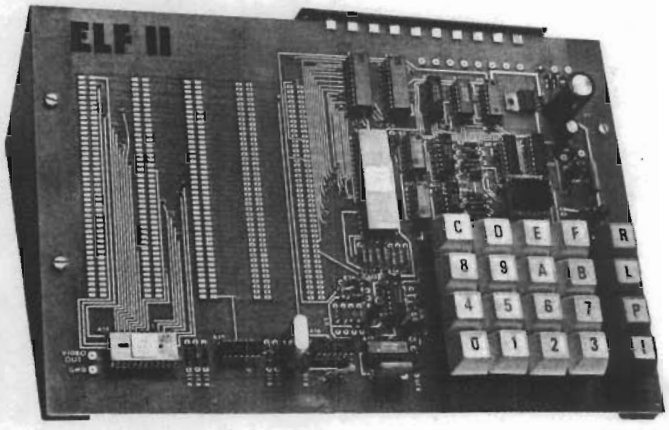
***Adding one chip to the Elf provides complete video interface and animated graphics capability for less than $25.***

If you own an Elf microcomputer (see POPULAR ELECTRONICS August 1976) or are planning to build one soon, the addition of a single IC and a handful of support components, and a change in the crystal frequency, can give you Pixie graphics. The entire graphics system is built into the new CDP 1861 LSI chip that sells for less than $20 from RCA parts distributors. (A complete kit is available; see Parts List.) The two other IC's in the optional add-on system are for a crystal oscillator that allows the graphics IC to generate the correct TV horizontal and vertical sync pulses.

The photo at the top of this page illustrates what can be done with the original 256 bytes of memory in the Elf when the Pixie graphics system is added. In this article, we will show you how to install and program the Pixie system to produce this type of graphics.

**Some Details.** The unique Pixie graphics system employs the direct memory access (DMA) capability built into the 1802 microprocessor in the Elf




*The basic Elf project orginally published in the August 1976 issue of* POPULAR ELECTRONICS *is shown at left with Pixie components added. Elf II is a complete kit including a pc board, hexadecimal keypad, Pixie graphics components and expansion bus (see Parts List).*

| 0000 | 0001 | 0002 | 0003 | 0004 | 0005 | 0006 | 0007 |
|---|---|---|---|---|---|---|---|
| 0008 | 0009 | 000A | | | | 000E | 000F |
| 0010 | 0011 | 0012 | | | | 0016 | 0017 |
| | | | | | | | |
| 00F0 | 00F1 | 00F2 | | | | 00F6 | 00F7 |
| 00F8 | 00F9 | 00FA | 00FB | 00FC | 00FD | 00FE | 00FF |

*Fig. 1. Memory addresses of bytes mapped onto TV screen in sample program.*

to work in conjunction with the new graphics IC. This allows you to display any 256-byte segment of memory on a CRT monitor or TV receiver. The output of the new chip is a 1-volt composite video/sync signal.

The selected segment of memory appears on-screen as an array of small squares that represent individual memory bits. If a memory bit is a 1, the appropriate square will be white, while if a bit is a 0, the square will be dark. Changing the bit pattern within the memory will change the pattern that appears on-screen. You can store several different bit patterns (pictures) in memory and,

**TABLE I—TEST PROGRAM**

| Label | M | Bytes | Comments |
|---|---|---|---|
| Start | 0000 | 90 B1 B2 | R1.1,R2.1=00 |
| | 0003 | B3 B4 | R3.0,R4.0=00 |
| | 0005 | F8 2D A3 | R3.0=(main) |
| | 0008 | F8 3F A2 | R2.0=(stack) |
| | 000B | F8 11 A1 | R1.0=(interrupt) |
| | 000E | D3 | P=3 (go to main) |
| Return | 000F | 72 | restore D, R2+1 |
| | 0010 | 70 | restore XP,R2+1 |
| Interrupt | 0011 | 22 78 | R2-1, save XP @ M2 |
| | 0013 | 22 52 | R2-1, save D @ M2 |
| | 0015 | C4 C4 C4 | no-op (9 cycles) |
| | 0018 | F8 00 B0 | |
| | 001B | F8 00 A0 | R0=0000(refresh ptr) |
| Refresh | 001E | 80 E2 | D=R0.0 |
| | —— | —— | 8 DMA cycles (R0+8) |
| | 0020 | E2 20 A0 | R0-1,R0.0=D |
| | —— | —— | 8 DMA cycles (R0+8) |
| | 0023 | E2 20 A0 | R0-1,R0.0=D |
| | —— | —— | 8 DMA cycles (R0+8) |
| | 0026 | E2 20 A0 | R0-1,R0.0=D |
| | —— | —— | 8 DMA cycles (R0+8) |
| | 0029 | 3C 1E | go to refresh (EF1=0) |
| | 002B | 30 0F | go to return (EF1=1) |
| Main | 002D | E2 69 | X=2, turn TV on |
| | 002F | 3F 2F | wait for IN pressed |
| | 0031 | 6C A4 | set MX,D,R4.0=toggles |
| | 0033 | 37 33 | wait for IN released |
| | 0035 | 3F 35 | wait for IN pressed |
| | 0037 | 6C | set MX,D=toggles |
| | 0038 | 54 14 | set M4=D, R4+1 |
| | 003A | 30 33 | go to M33 |

# PIXIE ANIMATION PROGRAM

BY EDWARD C. DEVEAUX

THE PROGRAM given here can be used with the Pixie version of the Elf microcomputer to create animation graphics using only the original 256 bytes of memory. The interrupt routine uses the same timing as described in previous Elf articles. However, a counter has been added to this routine, and we load the refresh address into R0 from R4. The main line of the program has been completely rewritten and contains shift, roll, and INPUT switch read routines.

The shift routine shifts 16 lines of the display to the right one bit at a time; bits shifted off the rightmost byte are shifted back onto the display in the

```
LOC  COSMAC CODE  LNNO  SOURCE LINE

                    1   .. AN 1802 ANIMATION PROGRAM by E. DEVEAUX
                    2   ..
78                  3   BEGSFT=#78  .. ADDRESS OF FIRST LINE SHIFTED.
                    4   ..
                    5   .. THIS PROGRAM PROVIDES VARIABLE SPEED
                    6   .. ANIMATION OF THE IMAGE LOCATED AT #78 to
                    7   .. #F7 IN MEMORY.
                    8   .. SPEED CONTROL IS PROVIDED BY INPUT SWITCHES.
00   90             9            GHI R0       ..ZERO HIGH ORDER OF
01   B1            10            PHI R1       ..R1 R2 R3.
02   B2            11            PHI R2
03   B3            12            PHI R3
04   B4            13            PHI R4       ..R4 POINTS TO REFRESH
05   A4            14            PLO R4         ADDRESS
06   F816          15            LDI A.0(INTRPT)
08   A1            16            PLO R1
09   F813          17            LDI A.0(STACK)
0B   A2            18            PLO R2
0C   F831          19            LDI A.0(MAIN)
0E   A3            20            PLO R3
0F   D3            21            SEP R3       ..GO TO MAIN_LINE
10   01020300      22            DC#01020300 ..STACK AREA
13                 23   STACK =*-1
                   24   ..
                   25   ..THIS PROGRAM USES A MODIFIED VERSION
                   26   ..OF THE INTERRUPT ROUTINE THAT APPEARED
                   27   ..IN COSMAC ELF PART 4.
                   28   ..
                   29   ..A SHIFT ROUTINE HAS BEEN ADDED THAT MOVES THE
                   30   .. STARSHIP FROM LEFT TO RIGHT ACROSS THE CRT.
                   31   ..
14   72            32      RETURN:LDXA
15   70            33             RET          ..CYCLES
16   22            35      INTRPT:DEC  R2       .. 2
17   78            36             SAV          ..4 R5 COUNTS REFRESH
18   22            37             DEC  R2      ..6 CYCLES, USED TO
19   52            38             STR  R2      ..8 DETERMINE WHEN TO
1A   15            39             INC  R5      ..10 SHIFT /ROLL.
1B   C4            40             NOP          ..13
1C   94            41             GHI  R4      ..15 R4 TO R0
```

using software, display them successively onscreen to produce animation effects. Low-resolution alphanumerics can also be created.

Since the basic Elf has only 256 bytes of memory, we will show how to display the entire memory on the screen. The memory is mapped as shown in Fig. 1, in an array of 64 spots wide (eight bytes with eight bits/byte) by 32 spots high to make a total of 256 bytes.

The byte at M(0000) is displayed at the upper-left of the screen; each row on the screen is equivalent to eight memory bytes. Byte M(00FF) appears at the bottom-right of the screen.

**Circuit Operation.** The entire schematic diagram for the Pixie graphics display system is shown in Fig. 2A. It consists of five components: the 1861 chip, a phono jack for the video output, and three resistors. The circuit shown in Fig. 2B may be used to replace the original crystal used in the Elf microcomputer. This is necessary because, to use the graphics display, the original crystal frequency must be changed to approximately 1.760640 MHz to generate the correct TV horizontal and vertical sync pulses. Crystals of this frequency may be expensive. The Fig. 2B circuit uses a

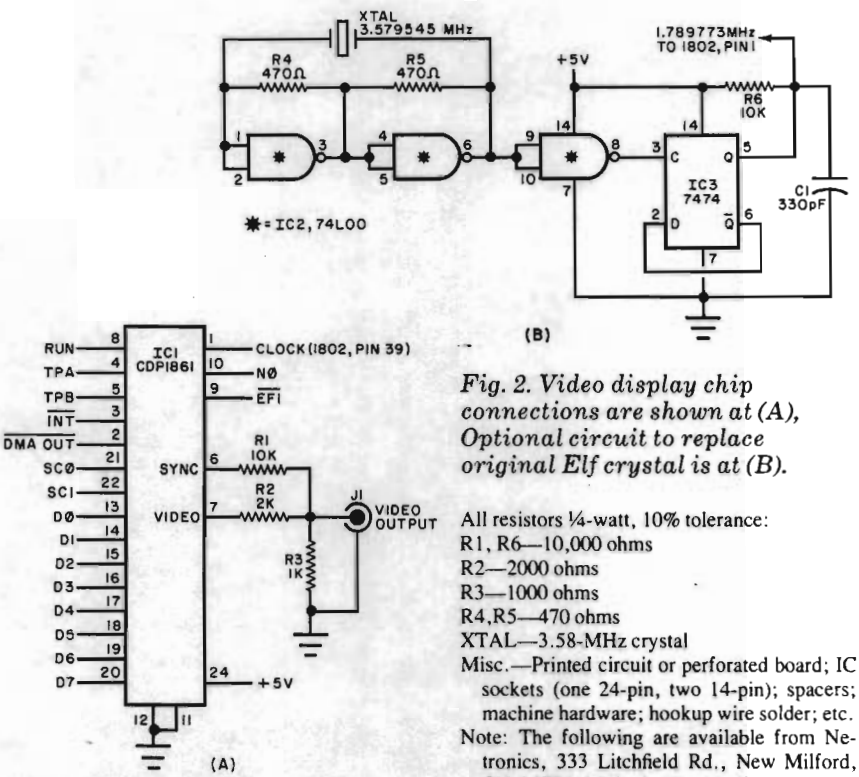



*Fig. 2. Video display chip connections are shown at (A), Optional circuit to replace original Elf crystal is at (B).*

## "PIXIE" PARTS LIST

C1—330-pF disc capacitor
IC1—CDP 1861 video IC (RCA)
IC2—74L00 low-power quad 2-input NAND gate IC
IC3—7474 dual-D flip-flop IC
J1—Phono jack
All resistors ¼-watt, 10% tolerance:
R1, R6—10,000 ohms
R2—2000 ohms
R3—1000 ohms
R4,R5—470 ohms
XTAL—3.58-MHz crystal
Misc.—Printed circuit or perforated board; IC sockets (one 24-pin, two 14-pin); spacers; machine hardware; hookup wire solder; etc.
Note: The following are available from Netronics, 333 Litchfield Rd., New Milford, CN 06776: kit including all of above Pixie components except those under "Misc." at $24.95; complete Elf II kit (basic Elf plus Pixie components and hexadecimal keyboard), including pc board, keyboard support IC's and expansion bus at $99.95, plus $3.00 shipping. Connecticut residents, add 7% sales tax.

high-order position of the first byte on the line.

The 32 lines of the display can be moved up one line by incrementing the starting refresh address by eight between refresh cycles. Decrementing register 4 (R4) allows the display to be rolled down. Hence, varying the frequency of shifts or rolls varies the animation speed of the displayed image.

Control of the speed is via the Elf's conventional INPUT switches. Setting all switches to zero and depressing the INPUT pushbutton causes a hex 00 to be read into location 13 (stack), in which case, there will be no movement of the displayed image. Loading any nonzero bit through the INPUT switches will animate the image. Any bits loaded are compared to the bits in the low-order byte of register 5 (R5). A shift or roll routine is initiated whenever there is a match between the bits of the low-order byte of R5 and the bits in the byte read into location 13. Register 5 is used to count the refresh cycles and is incremented by one every interrupt cycle.

```
1D  B0          PHI  R0      ..17 REFRESH ADDRESS
1E  84          GLO  R4      ..19
1F  A0          PLO  R0      ..21
                             ..
20  80          GLO  R0      ..23
21  80          GLO  R0      ..25
22  80    REFRESH:GLO  R0     .. 27
23  E2          SEX  R2      .. 29  8 DMA CYCLES
                             ..
24  E2          SEX  R2      ..
25  20          DEC  R0
26  A0          PLO  R0      ..      8  DMA CYCLES
                             ..
27  E2          SEX  R2
28  20          DEC  R0
29  A0          PLO  R0      ..      8 DMA CYCLES
                             ..
2A  E2          SEX  R2
2B  20          DEC  R0
2C  A0          PLO  R0      ..      8 DMA CYCLES
                             ..
2D  3C22        BNI  REFRESH .. ON EF1 REFRESH
2F  3014        BR  RETURN   .. IS OVER.
31  E2      MAIN:SEX  R2     .. RX=2
32  69          INP  1       .. TELL 1861 TO
                             ... TURN ON CRT.
..SFREAD READS INPUT SWITCHESTO CONTROL
..SPEED OF SHIFTS/ROLLS.
..INPUT SWITCH IS STORED AT STACK M(R2).
..
..INITIAL VALUE OF STACK IS ZERO AND THERE IS
..NO MOVEMENT OF STAR SHIP UNTIL A NON ZERO BIT
..IS INPUT.
33  3F38    SFREAD:BN4   CKSHIF   .. IF NO INPUT GO SEE
35  3735    WTREAD:B4    WTREAD   .. IF TIME TO SHIFT.
37  6C          INP      4        .. READ INTO STACK.
..                                ..
38  85      CKSHIF:GLO   R5       .. GHI R5 VARY/SPEED
39  F2          AND               .. OF STAR SHIP.
3A  3233        BZ       SFREAD   .. SHIFT/ROLL BIT MATCH.
3C  F800        LDI      A.1(BEGSFT)   ..BR ROLL 3061
3E  B9          PHI      R9            ..ROLL NO SHIFT.
```

readily available 3.58-MHz color-TV crystal and frequency divider to generate 1.789773 MHz, which is close enough for the 1861 chip to perform properly.

The 1861 chip uses the same clock as the 1802 μP chip to trigger internal counters to provide the TV-like composite sync at pin 6. The graphics display is directly refreshed from the memory 60 times each second, accomplished by an interrupt request sent to the 1802 at the same rate.

When the 1802 receives the interrupt request, it temporarily stops the program it is executing and immediately branches to the interrupt routine previously stored in memory. This branch occurs when P is automatically set to 1 and X is set to 2. The interrupt routine program counter is always R1, which must be set to the address of the interrupt routine before the 1861 is activated and starts sending interrupts to the 1802. A pulse from NO is sent to pin 10 of the 1861, permitting this chip to start sending interrupts. A 69 instruction can be used to generate the 1861 activation pulse. The 1861 is always turned off when the Elf is stopped with the RUN switch down.

In the program shown in Table I, R1 is set to the address of the interrupt routine at M(0011), R2 is set to the address of the work area (or stack) used subsequently for byte storage, R3 is set to the main program starting at M(002D), and setting P=3 causes a branch to M(002D) with R3 as the program counter. The main program permits entry of the bytes at any time via the Elf's toggle switches. This permits you to see what is happening to the CRT screen as memory bytes are changed. The program loops on itself until an interrupt signal is generated by the 1861, activated by the 69 instruction at M(002E).

Exactly 29 machine cycles after the initiation of the interrupt routine, the 1861 requests eight sequential memory bytes by pulling down the DMA-OUT (pin-2) request line for eight bytes (eight machine cycles). This automatically causes eight memory bytes, addressed by R0, to be sequentially fetched and transferred to the 1861 via the data bus. Note that the C4 instructions at M(0015) are special no-op instructions that re-

**TABLE II—SPACESHIP PROGRAM**

| M | Byte Sequence |
|---|---|
| 0040 | 00 00 00 00 00 00 00 00 |
| 0048 | 00 00 00 00 00 00 00 00 |
| 0050 | 7B DE DB DE 00 00 00 00 |
| 0058 | 4A 50 DA 52 00 00 00 00 |
| 0060 | 42 5E AB D0 00 00 00 00 |
| 0068 | 4A 42 8A 52 00 00 00 00 |
| 0070 | 7B DE 8A 5E 00 00 00 00 |
| 0078 | 00 00 00 00 00 00 00 00 |
| 0080 | 00 00 00 00 00 00 07 E0 |
| 0088 | 00 00 00 00 FF FF FF FF |
| 0090 | 00 06 00 01 00 00 00 01 |
| 0098 | 00 7F E0 01 00 00 00 02 |
| 00A0 | 7F C0 3F E0 FC FF FF FE |
| 00A8 | 40 0F 00 10 04 80 00 00 |
| 00B0 | 7F C0 3F E0 04 80 00 00 |
| 00B8 | 00 3F D0 40 04 80 00 00 |
| 00C0 | 00 0F 08 20 04 80 7A 1E |
| 00C8 | 00 00 07 90 04 80 42 10 |
| 00D0 | 00 00 18 7F FC F0 72 1C |
| 00D8 | 00 00 30 00 00 10 42 10 |
| 00E0 | 00 00 73 FC 00 10 7B D0 |
| 00E8 | 00 00 30 00 3F F0 00 00 |
| 00F0 | 00 00 18 0F C0 00 00 00 |
| 00F8 | 00 00 07 F0 00 00 00 00 |

```
3F  F878        84              LDI   A.0(BEGSFT)
41  A9          85              PLO   R9          ..R9=FIRST LINE
42  F810        86              LDI   16          ..TO SHIFT.
44  A6          87              PLO   R6          ..SHIFT 16 LINES.
45  99          88    NXTLNE:GHI   R9
46  BA          89              PHI   RA        ..SAVE ADDRESS OF 1st
47  89          90              GLO   R9        ..ON LINE IN RA
48  AA          91              PLO   RA
49  F807        92              LDI   7         ..R7=BYTES TO SHIFT-1.
4B  A7          93              PLO   R7
4C  09          94              LDN   R9
4D  B8          95              PHI   R8        ..SAVE 1ST BYTE ON
4E  76          96              SHRC            ..LINE IN R8.1
4F  19          97    NXTBYT:INC   R9          ..POINT R9 TO NEXT BYTE.
50  09          98              LDN   R9        ..LOAD NEXT BYTE.
51  76          99              SHRC            ..SHIFT RIGHT.
52  59          100             STR   R9        ..STORE BYTE
53  27          101             DEC   R7
54  87          102             GLO   R7        ..CHECK IF ALL BYTES
55  3A4F        103             BNZ   NXTBYT    ..SHIFTED.
57  98          104             GHI   R8        ..PUT BIT 0 of 8TH
58  76          105             SHRC            ..BYT ON BIT 7 OF
59  5A          106             STR   RA        ..1ST BYT ON LINE.
5A  19          107             INC   R9        ..R9=BYTE 0 NXT LINE.
5B  26          108             DEC   R6
5C  86          109             GLO   R6        ..CHECK IF 16 LINES
5D  3A45        110             BNZ   NXTLNE    ..SHIFTED.
5F  3033        111             BR    SFREAD    ..SKP 38 ROLL AND SHIFT.
61  84          112   ROLL:GLO    R4            ..INCREMENT R4 ONE LINE
62  FC08        113             ADI   8         ..ROLL SCREEN UP.
64  A4          114             PLO   R4
65  94          115             GHI   R4        ..CHANGE LNNO 116 TO
66  F800        116             LDI   00        ..ADCI 0 7C00 IF MORE
68  B4          117             PHI   R4        ..THAN 256 BYTES.
69  3233        118             BZ    SFREAD
6B  84          119             GLO   R4
6C  B4          120             PHI   R4
6D  3033        121             BR    SFREAD
6F  00          122             DC    #00
                123 ..ENTER IMAGE TO BE SHIFTED IN LOCATIONS
                124 ..X'78' - x'F7'.
                125             END
```

The numbers in the program flow chart (right) refer to the line numbers in the program. The program can be set up to shift or roll, or shift and roll. The program is loaded into locations 78 through F7. (Try using the program for the starship shown in Table II of the Pixie article.) Only the data loaded into 78 through F7 is shifted, but the entire area from 00 through FF is rolled.

Loading the program exactly as it is listed here will enable the shift routine only. Loading a 38 (SKP instruction) in location 5F (line 111) will enable both shift and roll routines. Loading 30 61 (BR ROLL) in locations 3C and 3D (line 82) will enable only the roll routine.

After loading and running the program, animation of the display will begin after any nonzero byte is loaded via the INPUT switches and operation of the INPUT pushbutton. By varying the INPUT bit pattern, you can control the speed of the animation.

If you have never seen a stack in "motion" when a program is running, take a look at displayed location 13. Then vary the speed. ◇

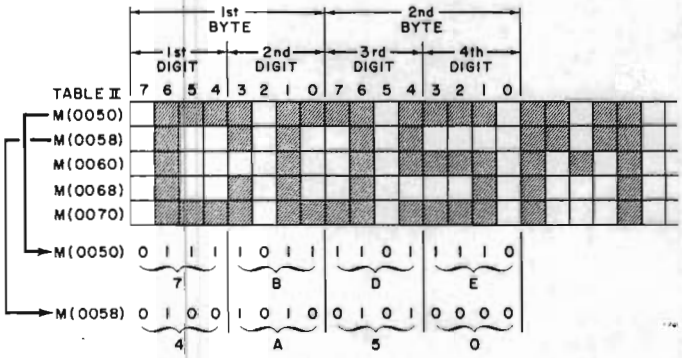



*Fig. 3. Diagram showing how to create your own display. This one is for parts of five lines of Spaceship Program.*

quire three cycles for each execution. These are used only to provide the delay required between the beginning of the interrupt routine and the first eight-byte DMA request generated by the 1861 display circuits.

Each of the eight display refresh bytes requested by the 1861 is internally converted to a bit serial form and used to provide the luminance (brightness) pulses that come out of the 1861 at pin 7. The actual raster display consists of 262 horizontal lines for each frame, and there are 60 frames per second. Each display spot is four raster lines high, which means that each eight-byte display row must be repeated four times. With the interrupt routine, R0 is initially set to M(0000), which means that the first DMA request causes the eight bytes from M(0000) to M(0007) to be fetched and displayed. The time of each raster line is exactly 14 machine cycles to permit the transfer of eight bytes (eight cycles) plus the execution of three two-cycle instructions during each raster line time. Following the eight DMA cycles required to refresh the first eight bytes, R0 is restored to its original value so that it remains pointing at the same eight bytes.

The E2 20 A0 instructions at M(0020), M(0023), and M(0026) are used to occupy six machine cycles between the DMA requests and to restore R0 to its initial value before incrementing it by eight during the eight-byte DMA request. The 20 instruction decrements R0.1 back to its initial value if a 256-byte page boundary was crossed during the preceding eight DMA cycles.

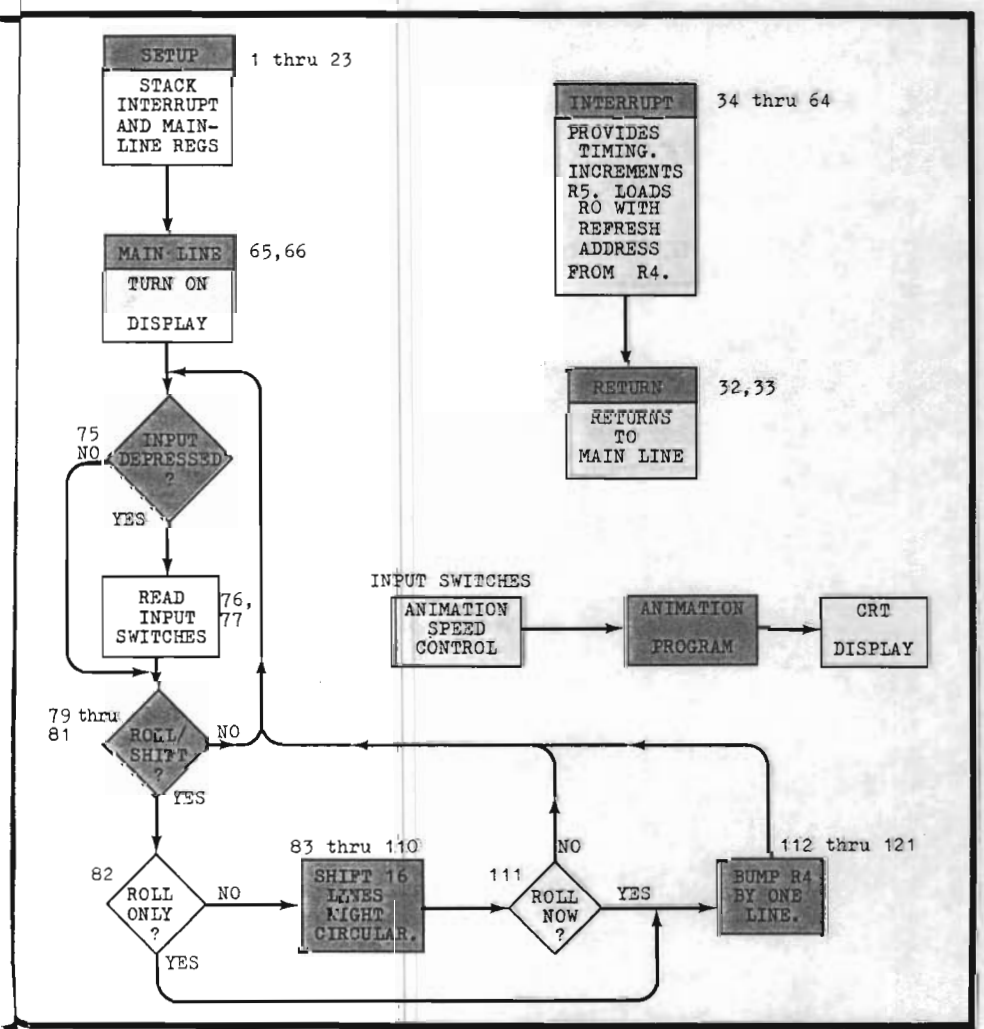
After the first group of eight bytes has been displayed for four raster line times, R0 is permitted to advance to the next group of eight bytes to be displayed. This process is continued until 32 groups of eight bytes each (256 total) have been displayed. At this time, the circuits in the 1861 chip cause line EF1=1 (at pin 9) and the interrupt routine terminates.

**Other Considerations.** The raster refresh involves the display of 32 groups of eight bytes, and each row of eight bytes is repeated on four raster line scans. This means that the display refresh ties up the 1802 μP for slightly more than 128 raster lines (32 × 4). Since there are 262 raster lines per frame, the μP spends about 50% of its time performing the display-refresh function.

Since the 1802 and 1861 clocks must remain synchronized, none of the three-cycle instructions described in the 1802's user's manual should be used in programs that run concurrently with this display. The only exception is the use of the C4 instruction in the interrupt routine.

The sample program given in Table I was designed to run in expanded-memory systems as well as in the basic 256-byte Elf. In the expanded system, just change the bytes at M(0019) and M(001C) so that R0 initially points to any 256-byte segment of the memory you wish to display on the raster. You can write any other main program to run concurrently with this interrupt routine.

The 1861 chip can also be used to display any number of memory bytes from eight to 1024 by rewriting the interrupt routine. For example, change the byte at M(0024) from 20 to 80, and you will see 512 bytes displayed on the CRT screen as 64 spots horizontally by 64 spots vertically. If you have only 256 bytes of memory in your system, you will see the same 256 bytes repeated twice on the screen. When displaying 512 bytes, each spot represents half the



height of those displayed when 256 bytes are displayed.

One of the main advantages of mapping main memory directly into the monitor or TV raster is the ability to manipulate the display using the normal instruction set. In systems that employ an external frame buffer for refresh, specialized instructions are required to change buffer contents. The buffer memory also costs more money. With the refresh buffer approach toward animation, you must store two picture patterns in memory and alternately transfer them to the buffer memory. Using the Pixie graphics display described here, you store the same two-picture patterns in memory but you need only change the initial value of R0 to alternately display them. Not only do you save the cost of a refresh buffer, you can greatly simplify the programming.

**Construction.** The Pixie circuit can be mounted on the original Elf board by relocating the crystal and two capacitors to the center of the board. Now, the 1861 IC goes on the upper left of the board, the resistors on the bottom of the board, and the output jack on the rear apron of the chassis.

Remove the crystal from the Elf and wire the Fig. 2B frequency divider to pin 1 of the 1802 μP. Then interconnect the two boards exactly as shown in Fig. 2A and B, including the power lines. Jack *J1* can be mounted on a small metal bracket and secured to the add-on board with No. 4 machine hardware. Also, mount *R1* and *R2* on the add-on board via "flea" clips because they may have to be changed for different-value resistors to suit the modulation requirements of the particular monitor you are using.

**Sample Display Program.** To test the Pixie, load the program given in Table I, starting at location M(0000). When this program is run, a random spot pattern should be displayed on-screen. At this time, you may have to alter the values of *R1* and *R2* to produce a tight sync lock and the desired modulation level of the spots. These are only level-adjust resistors and play no role in the actual sync or video production. The displayed pattern represents whatever is stored in the Elf's memory. The top eight rows represent the program given in Table I.

You can familiarize yourself with the new graphics ability of your computer if you visualize a grid of 64 boxes wide by 32 boxes deep, assuming a 256-byte memory. Bear in mind that the operating program given in Table I occupies the top eight lines. Since the program ends at memory location M(003B), load 00 into memory location M(003F) to complete that line.

Now, to display the spacecraft shown in the lead photo, load the programs given in Tables I and II in that order, starting the Table II program at memory location M(0040). Reset and switch to RUN.

If you wish to create your own display, Fig. 3 illustrates how to arrive at the correct hex digits. (In this case, the example used is for a small area of the program in Table II.) Use graph paper to "draw" your picture, shading in the "spots" you want to be white on the CRT screen. Then transfer the line bit pattern into the eight hex bytes per line as shown in Fig. 3.

**Conclusion.** The Pixie system described here adds video graphics to your Elf microcomputer at very low cost. So far, we have described how the Pixie system can be used to put simple, stationary images on-screen. Accompanying this article is a program that will put the graphics in motion. ◇

# Electronic "Bell" for a TVT-II

***Lets you know when you are near the end of a line on a TV typewriter.***

BY DENNIS J. DEUTSCH

Here is an add-on circuit for the computer hobbyist that will give his setup the effect of a bell ringing near the end of a line as it does on a typewriter. The circuit, as shown in the diagram, is for use with the Southwest Technical Products CT-1024 TVT-II terminal.

The CT-1024 produces 32 characters per line, for which access is required to bits 1, 2, 4, 8, and 16 on the CT-1024. These are located at IC35 and IC42.

The circuit as shown is set up to produce the tone on character 27. (Bit 4 is inverted in the 7404 IC so that it is "NOT'ed".) The character number trap consists of an 8-input NAND gate in the 7430 and the single inverter (which can be a single transistor if desired). If you want to stay at character 27, eliminate the inverter and bit 4.

Once the character is counted, the resulting pulse turns on the 74121 one-shot for a short period of time. The timing values of the one-shot can be altered by changing the circuit's time constant.

The one-shot triggers a 555 timer used as a tone generator to drive a small 8-ohm speaker. To alter the tone, change the value of the capacitor between pin 6 of the 555 and ground. ◇