SSTV Pictures from Your Microcomputer

Everyone asks, "How can microprocessors be used in Amateur Radio?" If slow-scan TV is your thing, here is a startling answer.

By Barry Sanderson*

Many amateurs have had a desire to get into slow-scan TV reception and transmission. More are interested in microprocessors and their role in Amateur Radio. This article puts the two together. Using the technique described here, you can generate simple graphic and alphanumeric characters without a camera! First, let's review the slow-scan signal, its makeup, and characteristics.

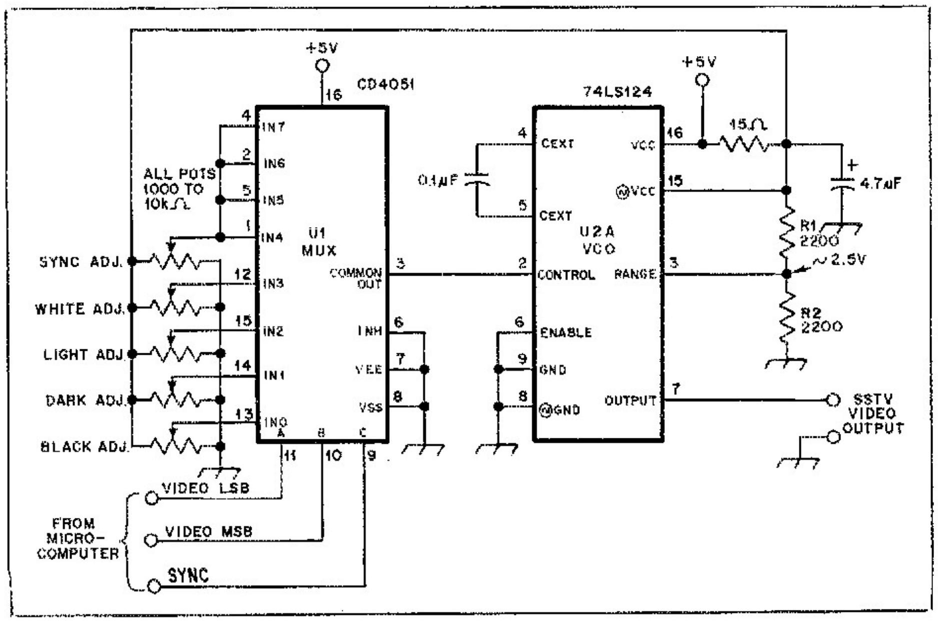
Slow-scan TV signals consist of two parts, the picture information and the synchronization information. The picture information is represented by a signal in the frequency range of 1500 Hz to 2300 Hz (1500 Hz corresponds to black and 2300 Hz corresponds to white). The frequencies in between correspond to shades of gray between black and white.

The synchronization information is represented by a frequency of 1200 Hz. The duration of the 1200-Hz signal distinguishes horizontal and vertical synchronization signals. The horizontal sync signal lasts for five ms and the vertical sync signal for 30 ms. The horizontal sync signal marks the boundary between the end of the picture information for one line and the start of the picture information for the next line. The vertical sync signal marks the boundary between the last or bottom line of one picture and the first or top line of the next picture.

At 15 lines per second, the time required by one line is 66.7 ms. Of this, five ms are required for the horizontal sync signal. The remaining 61.7 ms are filled with the picture information for that line. One entire picture is composed of 30 ms of sync signal (the vertical sync signal takes approximately one-half a line time) followed by 128 lines containing picture information and horizontal sync.[1]

*402 N. LaSalle, Indianapolis, IN 46201

Fig. 1 — A digital-to-analog-to-frequency converter for slow-scan television. No connections are made to IC pins not shown.
U1 — CMOS analog multiplexer, type CD4051.
U2 — TTL dual voltage-controlled oscillator, type 74LS124; one section unused.

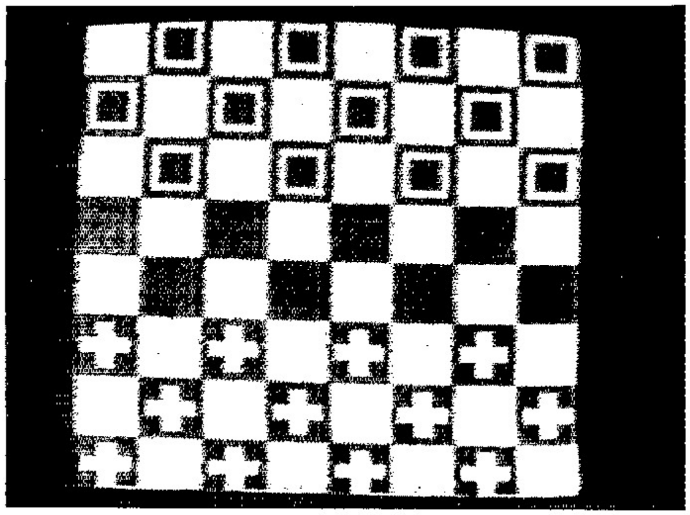
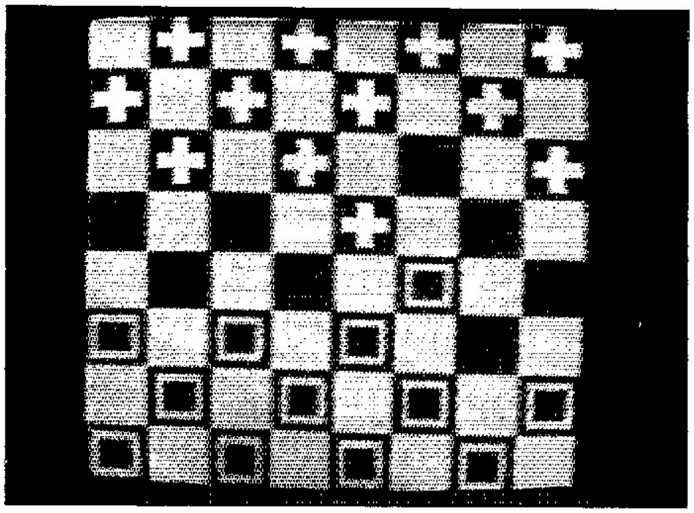


Picture Format

I made some changes in the picture format to simplify the signal-generation process. Rather than allowing 128 different lines in each picture, I allowed only 64 different lines in each picture. In order to make 128 total lines I repeated each line once so that the vertical dimension of each of my picture elements (pixels) is two of the 128 total lines or 1/64 of the total picture.

Also, I decided to make only 64 different pixels possible horizontally. Thus my pictures are an array of 64 by 64 squares or pixels. Each of these pixels is one of four shades of gray. In order of increasing brightness, these shades of gray are named: black, dark, light and white.

[1][Editor's Note: The amateur standard, adopted by gentlemen's agreement in 1968 when SSTV was first authorized for general use in the U.S. and Canada, calls for 120 lines per frame, requiring eight seconds to transmit a complete picture. In recent years the trend has been toward 128 lines per frame, especially when digital-logic circuitry is used to develop the sync signals, requiring 8.53 seconds to send a picture. With proper adjustment of the vertical sweep circuitry in a receiving monitor, either rate may be received satisfactorily.]

In the photo on the left is shown the screen at the starting point for a game of checkers. The white crosses and the light squares represent opposing sides in a game. At right, the board is shown as it might look after a couple of moves have been made.

I lumped the pixels together in groups of eight both vertically and horizontally. Each of these groups is called a character. Each character is a square of 64 pixels (eight pixels on a side). Thus the entire picture consists of a square of 64 characters. There are eight rows of characters with eight characters in each row.

There are no further limitations on what the characters look like. Besides the letters, numbers and punctuation marks that the word "character" connotes, characters can be made to symbolize anything you want. Some examples of things that characters could symbolize are chessmen, battleships, submarines, torpedoes, mines, tanks, artillery, shells, missiles, space ships, planets, stars, photon torpedoes and so on.

The hardware added to make the conversion from microcomputer logic output to the audio signal required for slow-scan is shown in Fig. 1. The conversion is done in two steps. First the logic levels are converted to an analog voltage by a multiplexer. Next, this analog voltage is converted to an audio frequency by a voltage-controlled oscillator (VCO).

The three microcomputer output bits (two bits representing one of four shades of gray, and one sync bit) are applied to the address inputs of an eight-channel CD4051 analog multiplexer. A separate pot is provided to set each of the five different levels (sync, black, dark, light and white). The selected signal is applied to the control input of half of a 74LS124 VCO. The slow-scan video signal is taken from the output of the 74LS124.

For this circuit to yield desirable results each pot must be adjusted so that the proper frequency is output by the VCO when that pot is selected to provide the control voltage to the VCO. The adjustment procedure consists of selecting each pot in turn and adjusting that pot until the VCO oscillates at the proper frequency for that pot. The proper frequencies are Sync — 1200 Hz; Black — 1500 Hz; Dark — 1767 Hz; Light — 2033 Hz and White — 2300 Hz.

The Program

My program is based on two things. First, a hardware timer that generates an interrupt at a regular interval, and second, a routine that is executed in response to the interrupt that is under program control and changes as the program executes.

The hardware timer is set to generate interrupts at the pixel-output rate. The interval is calculated from the slow-scan specification by dividing the time allotted for the picture portion of a line (approximately 61.7 ms) by the number of pixels per line (64). The result is 963.5 microseconds.

One of three interrupt service routines is executed in response to each interrupt from the interval timer. The three routines are named VSYNC, PICT and HSYNC. VSYNC is executed during vertical sync time. It inserts the vertical sync signal and changes the interrupt response to PICT at the end of the vertical sync signal. This means that the next interrupt will result in PICT being executed instead of VSYNC.

PICT fetches the picture information stored in a portion of memory and outputs it to the proper port. After 64 pixels have been output, it changes the interrupt response to HSYNC. Thus HSYNC will be executed in response to the next interrupt.

HSYNC activates the sync signal for five pixel times and then changes the interrupt response to either PICT or VSYNC. If 128 lines have been output, VSYNC is executed in response to the next interrupt; otherwise PICT is executed in response to the next interrupt.

Since the generation of a slow-scan TV signal corresponding to the stored picture is done by interrupt service routines, the leftover time is available to the main-line program to perform other functions.

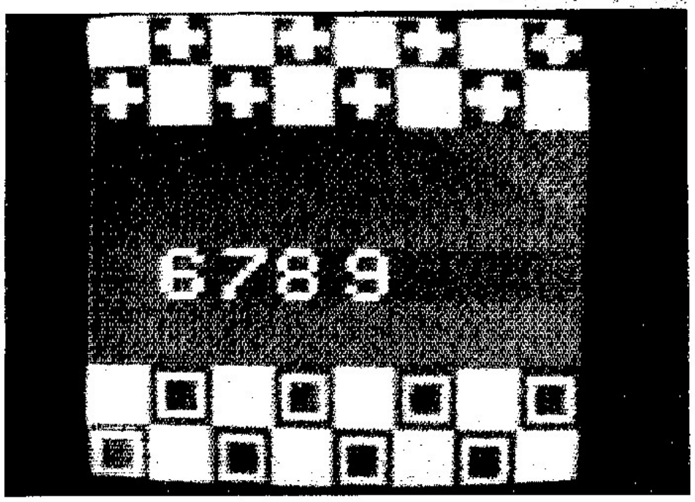
Picture Updating

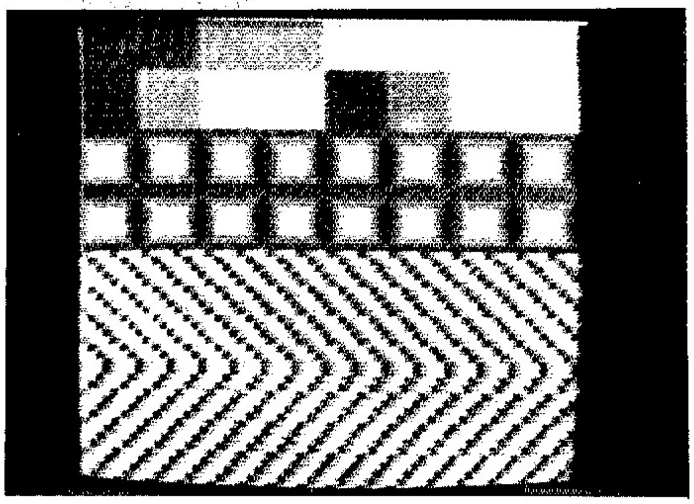
One thing that should be included in the main-line program is a way to change the contents of the memory locations that hold the picture. One approach is to allow a new character to be written into any of the 64 character positions on command from an input device (typically a keyboard). Both the new character and its position in the picture are required each time this command is executed. This type of picture updating is useful in game-playing situations, such as checkers or chess.

Another way to update the picture is to have the position in the picture of the new character incorporated into the program. Thus the input device just supplies a string of new characters and the program places them in the picture. This type of picture updating is useful for alphanumeric pictures or fixed pictures (CQ, QSL, etc.).

Still another way to update the picture is to have both the new characters and their positions in the picture incorporated into the program. This allows a predetermined sequence of pictures to be output.

Since the method of picture updating depends on the situation and the hardware available, I have not included a discussion of my main-line program. However, the three interrupt service routines and a fourth subroutine will actually do all of the work, generating the slow-scan TV signals from stored picture information. With a thorough understanding of the functions performed by these subroutines, you should be able to incorporate them into a program of your own that generates a slow-scan TV signal of pictures of your




At left is an example of 5 × 7 dot matrix numbers fit into characters and displayed on the screen. In the right photo, the top two rows of characters are gray scales. The next two rows consist of a square character that is black around the perimeter, white in the center, and light and dark in between. The bottom half of the picture consists of a sequence of gray scales where the shade of gray changes with every pixel. Each line of pixels is jogged one space to the right for the first 16 lines, then the direction of sliding of gray scales is reversed.

own choosing. This article will provide you with that necessary understanding.

The rest of the article discusses some counters which are used to control program execution, the way the picture is stored in memory, the GET C subroutine which fetches the next pixels to be output, and the three interrupt service routines.

Three eight-bit counters are used to control program execution. These are the X counter, the Y counter, and the PIXEL counter. The X counter is incremented each time a new pixel is output, and is reset to zero at the end of each horizontal sync period. The Y counter is incremented each time the X counter is reset to zero. The Y counter counts the number of picture lines that have been output. It is reset to zero during each vertical sync period. When taken together, the X and Y counters define the position of the pixel on the TV screen at any given moment. The PIXEL counter is used differently by the three interrupt service routines. These uses will be explained separately as the three routines are discussed.

Character Storage

Each pixel requires two bits to specify one of the four shades of gray. An 8-bit word can hold four consecutive pixels. Two consecutive 8-bit words hold eight consecutive pixels, or one line of a character. Thus for each character, 16 consecutive 8-bit memory locations are required to store the two-bit codes specifying the shade of gray for each of the 64 pixels.

In my model, the two video bits required by my interface circuit are connected to the highest order bits of an output port (bit 7 to the video most significant bit, MSB, and bit 6 to the video least significant bit, LSB). Therefore, bits 7 and 6 of the first word of memory for a particular character indicate the shade of gray for the first pixel (upper left corner) of that character. Bits 5 and 4 of that same word correspond to the second pixel of that character. The next two bits correspond to the next pixel, and so on.

Bits 1 and 0 of the second word correspond to the upper right-hand pixel. Bits 7 and 6 of the third word of memory for a particular character correspond to the first (left-most) pixel in the second line of the character. Bits 7 and 6 of the 15th word correspond to the first pixel in the eighth line of the character, and so on. Finally, the lower right-corner pixel is stored in bits 1 and 0 of the 16th word for that particular character.

As a specific example, consider the character diagrammed in Fig. 2A. The cross-hair lines indicate pixel boundaries. This character is a white cross on a black background. Fig. 2B lists the 2-bit codes required for this character according to the following convention: 00 = black, 01 = dark, 10 = light, 11 = white. The solid lines represent microcomputer word boundaries. The table in Fig. 2C shows a hex listing of the contents of the 16 consecutive memory locations required to store this character. A starting address of N is assumed.

Picture Storage

Since a picture is composed of 64 characters, I decided to store a 7-bit value in the first 64 locations on page one of memory to represent each character of the picture. (A page is 256 8-bit words.)

I made these the upper 7 bits of the 8-bit word and forced the LSB to be a zero. The resulting 8-bit word is used to fetch the low order half of a 16-bit address from page two of memory. The high-order half of the address is fetched from the location just above the one containing the low order half of the address. The upper seven bits of the location on page two where the low and high order halves of the address are stored are the same. The LSB is a zero for the low-order half and the LSB is a one for the high-order half. This 16-bit address specifies where the word containing the first four pixels of the character is stored. The remaining pixels are stored in the 15 words following the word in which the first four pixels are stored.

Bits in the X and Y counters specify which of the 16 memory locations associated with each character contains the pixels to be output next. These bits are used to calculate the offset from the word containing the first four pixels to the word containing the pixels to be output next.

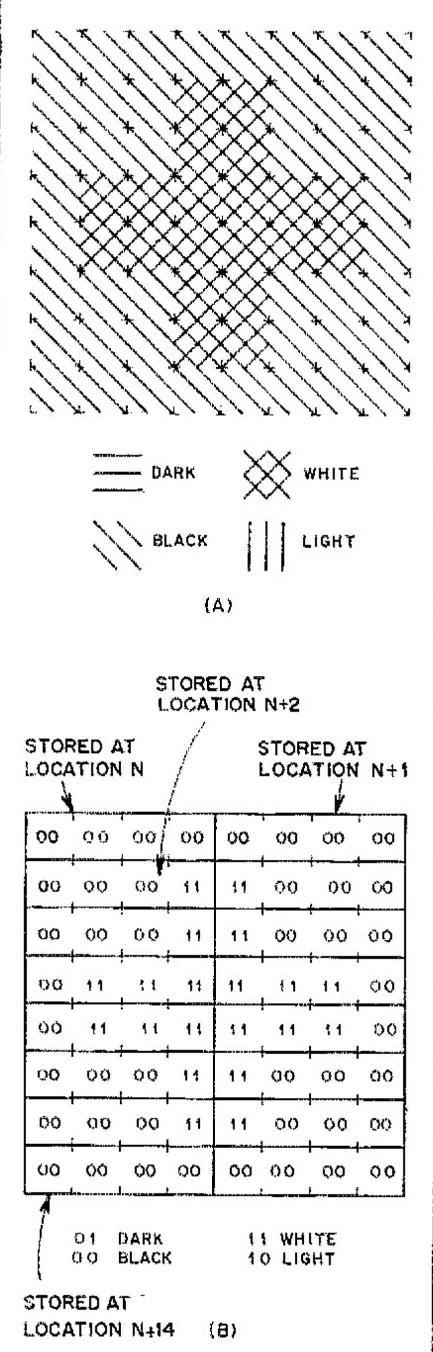
The GET C Subroutine

In order to understand the GET C routine, you should first realize what each bit in the X and Y counters corresponds to with respect to the TV picture. The individual bits are referred to by a letter and a number, the letter being X for the X counter and Y for the Y counter. X7 is the MSB of the X counter, XØ is the LSB of the X counter, Y6 is the next to the MSB of the Y counter, and so on.

X2, X1 and XØ specify which of the eight pixels on a given line of a given character is to be output next. The next three bits (X5, X4 and X3) specify which of the eight characters on a given line is to be output next. X6 goes from a 0 to a 1 when all 64 pixels on a line have been output.

Y3, Y2 and Y1 specify which of the eight lines of a character are currently being output. Y6, Y5 and Y4 specify which of the eight rows of characters is currently being output. Y7 goes from a 0 to a 1 after all 64 lines have been output twice (each line is repeated to make 128 lines). YØ indicates whether a particular line is being output for the first or second time.

As you can see in Fig. 3, the GET C



Address	Data (Hex)
N	00
N + 1	00
N + 2	03
N + 3	C0
N + 4	03
N + 5	C0
N + 6	3F
N + 7	FC
N + 8	3F
N + 9	FC
N + 10	03
N + 11	C0
N + 12	03
N + 13	C0
N + 14	00
N + 15	00

(C)

Fig. 2 — Example of data to represent a character. The cross-hair lines at A indicate pixel boundaries; a complete SSTV frame is 64 × 64 pixels. The 8 × 8-pixel character is stored in 16 bytes of computer memory.

subroutine first assembles the bits from the X and Y counters, specifying one of the 64 character positions, into an 8-bit number. This number is in the range of 0-63 inclusive. Next the number of the character currently in that position is fetched from that memory location in page one. This character number is multiplied by two and the result used to fetch a 16-bit address from two consecutive memory locations on page two.

This 16-bit address gives the location of the word containing the first four pixels of the character. An offset of 0000$Y_3Y_2Y_1X_2$ is added to this 16-bit address. The result is an address of the memory location containing the next four pixels to be output. These four pixels are placed in the temporary video storage location. The GET C subroutine then returns control to the routine that called it.

Subroutines

The PICT subroutine (Fig. 4) uses the pixel counter to determine which of the four pixels in an 8-bit word have not been output yet. Each time this subroutine is executed a pixel is transferred from the temporary video storage location to the proper output port. The data in the temporary video storage location are rotated two bits to the left so that the next pixel is in the two highest order bit positions.

After every four pixels have been output, the GET C subroutine is called to update the temporary video storage location with the next four pixels to be output. After 64 pixels have been output, the interrupt response is changed to HSYNC and the pixel counter is loaded with 5. Therefore, the HSYNC subroutine will be executed in response to the next interrupt instead of the PICT subroutine.

The HSYNC subroutine (Fig. 5) uses the pixel counter to make the horizontal sync signal five pixel times long. This results in a horizontal sync signal about four percent shorter than the SSTV specification requires; however, it is close enough. After the pixel counter has been decremented to zero, the X counter is reset to zero and the Y counter is incremented. If all 128 lines have not yet been output, GET C is called (to update the contents of the temporary video storage location). The pixel counter is then set to 4 and the interrupt response is changed to PICT. If, however, all 128 lines have been output, the pixel counter is set to 31 and the interrupt response is changed to VSYNC.

The VSYNC subroutine (Fig. 5) uses the pixel counter to make the vertical sync signal 31 pixel times long. After the pixel counter has been decremented to zero, the Y counter is reset to zero. Then GET C is called (to update the contents of the temporary video storage location), the pixel counter is set to four and the interrupt response is changed to PICT.

Since a slow-scan TV signal is an audio waveform, it can be transcribed on any tape recorder with low noise, flutter and distortion characteristics. I made up some characters, and then some pictures from those characters. I used my program to generate the slow-scan TV signal corresponding to those pictures and recorded it. I then took this tape to someone who has a commercially available scan converter. This scan converter accepts a slow-scan video signal as an input and outputs a "normal" or fast-scan video signal which can be displayed on a TV monitor. The pictures in this article are ones I took of the TV monitor while it was hooked to the output of a scan converter receiving my taped slow-scan signals.

This particular scan converter did something to pictures with sharp high-contrast edges, causing a decrease in picture quality. The converter superimposed a checkerboard pattern of black dots on the picture. Instead of each pixel being the proper width, each pixel is only half as wide as it should be with the other half being a black dot. The black dot appears either to the left or right of the half-size

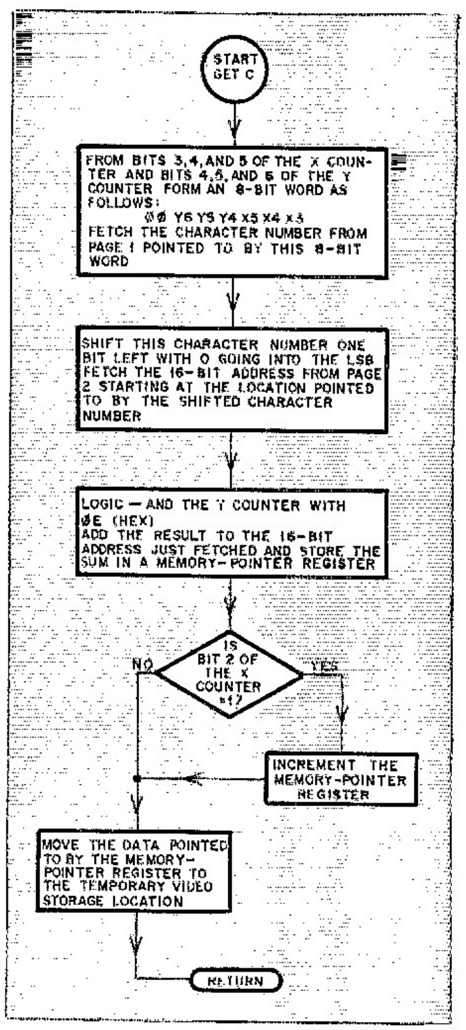
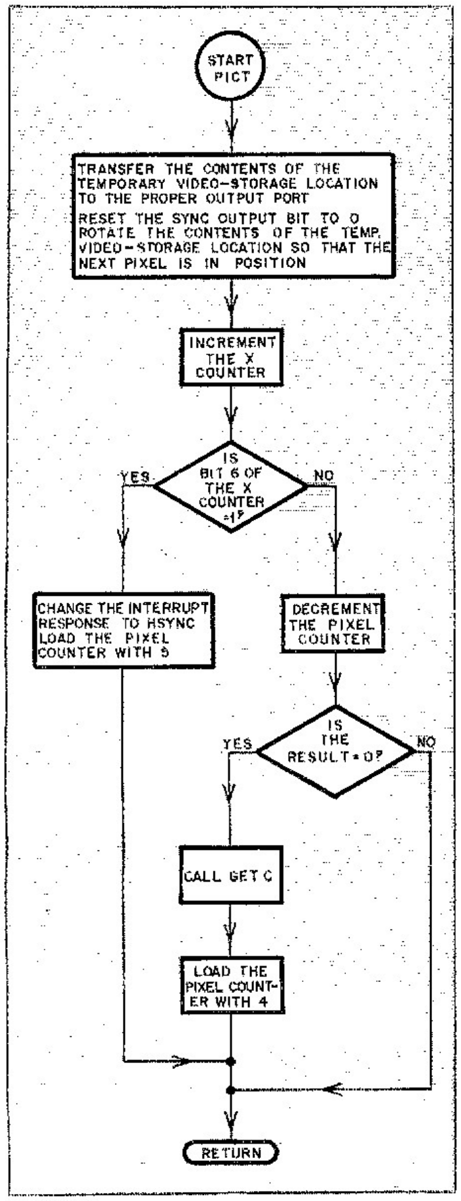
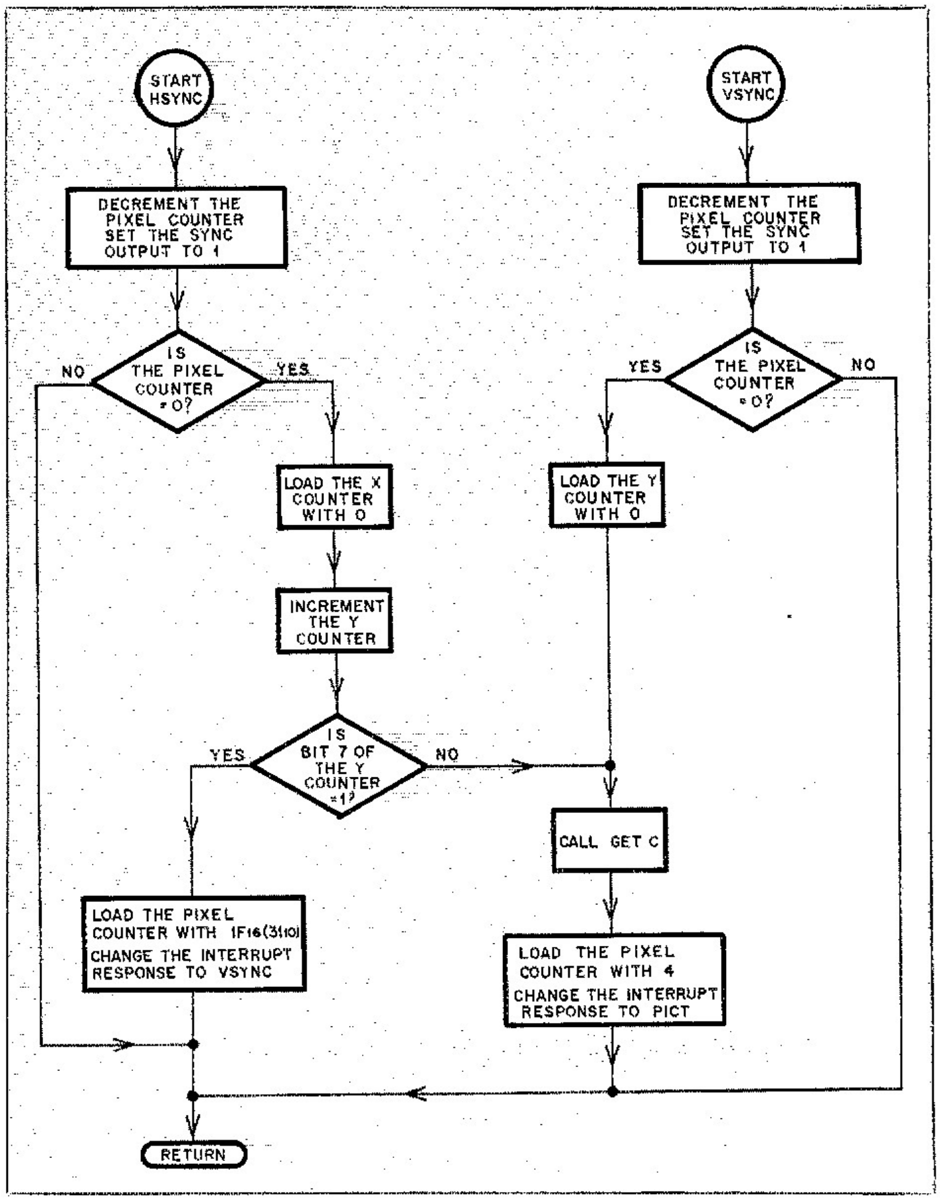

START GET C
FROM BITS 3,4, AND 5 OF THE X COUNTER AND BITS 4,5, AND 6 OF THE Y COUNTER FORM AN 8-BIT WORD AS FOLLOWS:
ØØ Y6 Y5 Y4 X5 X4 X3
FETCH THE CHARACTER NUMBER FROM PAGE 1 POINTED TO BY THIS 8-BIT WORD
SHIFT THIS CHARACTER NUMBER ONE BIT LEFT WITH 0 GOING INTO THE LSB
FETCH THE 16-BIT ADDRESS FROM PAGE 2 STARTING AT THE LOCATION POINTED TO BY THE SHIFTED CHARACTER NUMBER
LOGIC — AND THE Y COUNTER WITH ØE (HEX)
ADD THE RESULT TO THE 16-BIT ADDRESS JUST FETCHED AND STORE THE SUM IN A MEMORY-POINTER REGISTER
IS BIT 2 OF THE X COUNTER =1?
NO
YES
INCREMENT THE MEMORY-POINTER REGISTER
MOVE THE DATA POINTED TO BY THE MEMORY-POINTER REGISTER TO THE TEMPORARY VIDEO STORAGE LOCATION
RETURN

Fig. 3 — Flow diagram of the GET C subroutine.



START PICT
TRANSFER THE CONTENTS OF THE TEMPORARY VIDEO-STORAGE LOCATION TO THE PROPER OUTPUT PORT
RESET THE SYNC OUTPUT BIT TO 0
ROTATE THE CONTENTS OF THE TEMP. VIDEO-STORAGE LOCATION SO THAT THE NEXT PIXEL IS IN POSITION
INCREMENT THE X COUNTER
IS BIT 6 OF THE X COUNTER =1?
YES
NO
CHANGE THE INTERRUPT RESPONSE TO HSYNC
LOAD THE PIXEL COUNTER WITH 5
DECREMENT THE PIXEL COUNTER
IS THE RESULT=0?
YES
NO
CALL GET C
LOAD THE PIXEL COUNTER WITH 4
RETURN

Fig. 4 — Flow diagram of the PICT subroutine.



START HSYNC
DECREMENT THE PIXEL COUNTER
SET THE SYNC OUTPUT TO 1
IS THE PIXEL COUNTER =0?
NO
YES
LOAD THE X COUNTER WITH 0
INCREMENT THE Y COUNTER
IS BIT 7 OF THE Y COUNTER =1?
YES
NO
LOAD THE PIXEL COUNTER WITH 1F16(3110)
CHANGE THE INTERRUPT RESPONSE TO VSYNC
RETURN
START VSYNC
DECREMENT THE PIXEL COUNTER
SET THE SYNC OUTPUT TO 1
IS THE PIXEL COUNTER =0?
YES
NO
LOAD THE Y COUNTER WITH 0
CALL GET C
LOAD THE PIXEL COUNTER WITH 4
CHANGE THE INTERRUPT RESPONSE TO PICT

Fig. 5 — Flow diagram of the HSYNC, VSYNC subroutine.

pixel alternating from one line to the next. This accounts for the fuzzy vertical edges in these pictures.

I have purposely not included a listing of my program, because it is written in assembly level language and I attempted to use as many special Z-80 instructions (which are not a part of the 8080 instruction set) as practical. I also freely used special hardware features which I designed into my microcomputer. Thus, a significant number of the instructions in my program involve some special feature of my microcomputer. Rather than "sidetrack" to explain these, I have discussed the algorithm I used and hope you can adapt this algorithm to your microcomputer and application.

Strays

"ROADEO" HAMS

□ Backing through obstacles, steering around tennis balls and negotiating a serpentine course that would raise the hair of a Grand Prix driver were just some of the difficulties that 279 school bus drivers faced during the recent fifth annual Wisconsin School Bus Safety "Roadeo" in Madison, WI. Devoted to safety for the school-age rider, the event was made possible by the 21 hams who assisted. It was surely a communication and organizational challenge not seen since Field Day!

The Madison Area Repeater Association, Four Lakes Amateur Radio Club and Yellow Thunder Amateur Radio Club undertook the task of providing mobile, portable and fixed 2-meter equipment and personnel to keep track of the drivers, their scores and equipment.

Headquarters for the event was the Madison-Sheraton Hotel, where K9VAL acted as net control. Twenty other hams sporting an assortment of gear and antennas were scattered throughout the road course and elsewhere at the Dane County Exposition Center. K9BIL acted as field coordinator for these locations.

A bus equipped with 2-meter gear was used to shuttle entrants and spectators to and from various events. WR9ABT, the MARA repeater, provided clear, concise communications. — K9ZZ