Slow Scan TV with the Sound Blaster

By Gene Harlan, WB9MMM
5931 Alma Dr
Rockford, IL 61108

Buying special boards and equipment is not the only way to do slow scan TV. Now you can copy SSTV using an IBM PC compatible and a Sound Blaster card!

The frugal nature of most hams, including me, leads us to look for better and less expensive ways to do things. I have had an interest in slow scan TV (SSTV) for many years, but I never wanted to spend the money to buy new equipment. It hasn't been cheap to get into SSTV, at least not until the last couple of years. Several low-cost units have come to the market recently, and interest has soared because of them. This is evident in the number of articles explaining SSTV such as those in the January and February 1993 issues of *QST*[1,2].

I started in slow scan about 1977 by starting to build my own display. I had picked up a P7 picture tube at a hamfest, built the surrounding electronics, and had it partly working when a Robot Model 70 monitor & Model 80 camera became available at a price I couldn't pass up. So much for building my own equipment! I'll never forget the excitement of my first SSTV contact with XE1JOF in Mexico City. While all I sent him was a picture of myself sitting in front of my equipment, he sent me a travelogue of the area around Mexico City. At that time the only mode that was used to any extent was 8-second black and white. To send color you needed to send 3 separate frames of 8 seconds, one each of red, blue, & green. It's bad enough to sit still for 8 seconds, much less sitting still for 24 seconds!

As I said before, hams are always looking for better or different ways of doing things. My wife, Shari, bought me a Creative Labs Sound Blaster card for a Christmas present two years ago.[3] (How she thought of it I'll never know. . .) Well, you can't have a programmable board that you cannot program, so I purchased the developers' kit as well (I mean, she did). Now I could have some fun!

What is the Sound Blaster?

The Sound Blaster is a card that plugs into a slot in an IBM PC compatible computer and provides sound input and output capabilities. It is used with computer games, multimedia applications, or just plain anything that uses sound. There are currently three versions being sold: the standard Sound Blaster V2.0, the Sound Blaster Pro, and the new 16-bit version for even fancier applications. I'm using the Sound Blaster Pro. It accepts sound input via a microphone or LINE IN jack and produces stereo sound output. It even has its own built-in 4-watt amplifier. Another connector provides an input for a joystick or MIDI (musical instrument digital interface) device (see Fig 1). The Sound Blaster comes with a collection of programs such as *VEDIT2* (records voice and sounds, allows editing, echo effects and more), *TALKING PARROT* (great for the kids as it will mimic what you say in a higher pitch and sometimes tells you that you have "bad breath"!), and other programs that play music or other sounds. A fun and enjoyable device all by itself!

After getting the Sound Blaster, I started thinking of programs I wanted to write. It didn't take longer than the next time I heard SSTV on 14.230 MHz that I realized I had a project. Thoughts of connecting the audio out of my receiver to the microphone input of the Sound Blaster and connecting the sound output of the Sound Blaster to the microphone input of my transceiver raced through my head. Let the programming begin!

It was very slow going at first as the initial software from the development kit was for Microsoft C and I was

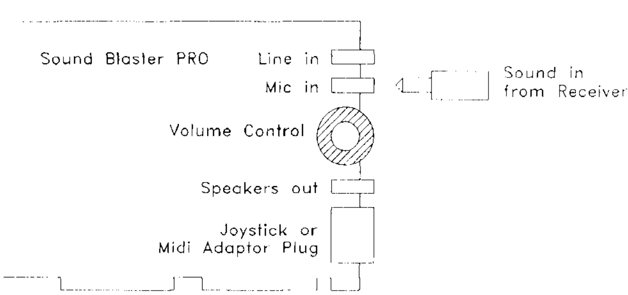

Fig 1—Diagram of the Sound Blaster's connections.

[1]Notes appear on page 6

Page 3, QEX, May 1993, published by The American Radio Relay League, Inc.

using Borland Turbo C. After about 9 months, I received an update disk that covered my needs. What I didn't know is that I *still* had a long road ahead. First I had to figure out the .VOC file format used to store sound, and I had to understand how to convert all those numbers in the file to frequencies. I decided the easiest way to begin would be by recording SSTV audio to a disk file then, when the sound stopped, read the file and display it on the screen.

Examining the .VOC File Format

The sound disk file has an extension of .VOC—in my case TEMP.VOC—which consists of a header and data.

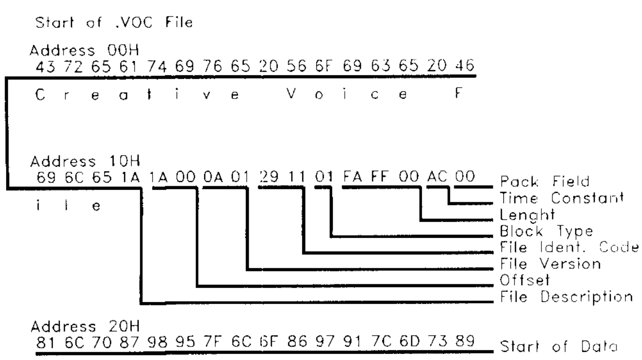

Fig 2—An example .VOC data file containing the data graphed in Fig 3. The format of the .VOC file is given below. Numbers of more than one byte are stored least-byte first, as is standard for Intel processors.

Bytes 00-13H	**File description ("Creative Voice File"+1AH).**
Bytes 14-15H	**Offset in the file of the first data block (normally 001AH).**
Bytes 16-17H	**File version (major, minor); currently 1.10**
Bytes 18-19H	**File identification. This is the complement of the version number added to 1234H; currently this results in 1129H.**

Each data block has a format as follows:

Byte 00H	**Data block type:** **00H = Terminator (EOF)** **01H = Sound data** **(Other types are not used by this program.)**
Bytes 01-03H	**Data block size in bytes (not used for terminator blocks).**

Type 1 data blocks include the following:

Byte 04H	**Sample rate constant = 256- (10^6/rate).**
Byte 05H	**Packing type; only type 0 (8-bit data) is used by this program.**
Bytes 06-?	**Sampled data bytes.**

The data portion consists of bytes of information with values from 0 to 255 decimal (00H to FFH hexadecimal). Fig 2 shows a set of numbers that make up one cycle of a given frequency. (Hexadecimal numbers are used for this example.) The numbers represent the input voltage, with the center point (zero volts) represented as the number 80H. The most positive amplitude possible is FFH and the most negative is 00H. The data in the example begins at location 20H in the file. The data, starting with 81H, 6CH, 70H, 87H, etc, are plotted in the graph of Fig 3 as *samples* of the input waveform.

The time between sample points is determined by the sample frequency. After trying sample rates from 8 kHz up to 22 kHz, a rate of 12 kHz was chosen. With this frequency, each sample has a time of $\frac{1}{12}$ kHz = 83.333 μs.

Calculating Frequency

Now we can begin calculating the input frequency. If we know the time of one cycle (the period), we can convert to frequency by taking the reciprocal of that number (1/period). Each input cycle extends over a number of samples. The tricky part is not in adding up the number of samples in the period, but in figuring the amount of time to allocate when the zero crossing occurs between samples. This effect is shown in Fig 3 at A and B. For my program, I assumed that the waveform was a straight line between points. Linear interpolation is then used to estimate the time of the zero crossing.

The calculations for the transition from sample 3 (70H) to sample 4 (87H) in Fig 3 are:

70H = 112 decimal,
87H = 135 decimal,
80H = 128 decimal
135 − 112 = 23 (this is the total swing from negative to positive)
135 − 128 = 7 (the amount on the positive side only)
Period = $\frac{1}{12000}$ = 83.333 μs
Time per 1-bit change in amplitude: 83.333 μs/23 = 3.623 μs
The time from the zero crossing until sample 4 is therefore: 3.623 μs × 7 = 25.62 μs to the period.

In the example of Fig 3, the cycle of the waveform consists of 5 sample periods of 83.333 μs each plus the time from the positive going zero crossing (A) which we calculated as 25.62 μs plus the time at the end of the waveform (B) which, using the same technique as above, is 61.591 μs. Adding these together gives a total period for that cycle of 503.876 μs which is 1984 Hz ($1/(503.876 \times 10^{-6})$). I found that I had to use a full cycle of audio to make the calculations rather than a half cycle because there could be a dc offset on the input which would make the positive side of the signal appear to be a very high frequency and the negative side a very low frequency, or vice versa.

One of the many good points of the Sound Blaster card is that it has its own clock so the sample rate does not depend on the host computer. An 8-kHz sample rate did not provide a satisfactory image, whereas 12 kHz did. Going to 22 kHz did not give much improvement so 12 kHz was chosen. This allows the use of the lower priced Sound Blaster with its maximum sampling rate of 12 kHz.

The SSTV Software

The choice of video mode was based on the fact that I haven't seen anyone else using the 640 × 480, 256-color VGA mode as yet. When I get to displaying color, I think it will look great. This mode is harder to work with, though, because when you use a 256-color video mode with more resolution than 320 × 200, there is no standard among manufacturers of video cards. It is necessary to detect which video card you are using, and then set up the proper mode. I purchased drivers from Peter Jones to overcome this problem.[4] Peter's documentation is well done and I used his fax machine a couple of times to get help! He was very prompt and helpful.

There are two versions of the software currently available. One is a shareware version which gives a taste of what SSTV is all about.[5] The only mode is the 8-second black and white mode. The second version includes Robot 8-, 12-, 24-, and 36-second black and white, and Robot 36- and 72-second color modes (displayed in black and white), as well as Scotty 1 and Scotty 2 (also displayed in black and white).[6,7] I plan a third version which will include color and add transmit capability to each of the previous versions. Hopefully it will be available by the end of 1993, depending on the work load at my "real" job.

During the development of the program, I needed a method for varying the sync, black, and contrast frequencies. These adjustable parameters have been left in the program for experimentation.

The sync frequency of SSTV is 1200 Hz, but I found that the program works well using anything below 1350 Hz.

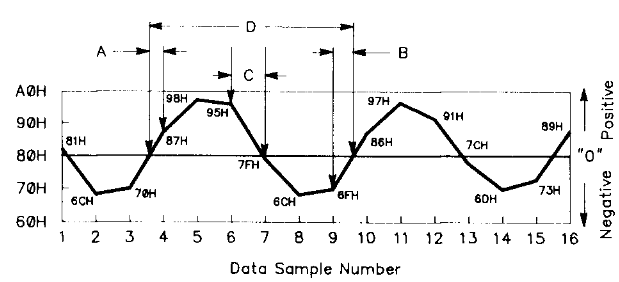

Fig 3—A graph of the example data. A sample occurs at each division of the horizontal axis. The vertical axis is the amplitude, from 00H to FFH, where 80H represents zero. The part of the graph labeled A is the time from the positive-going zero crossing to the next sample. This period is calculated by the program (see text). Likewise, B is the period from a sample to the negative-going zero crossing. C is the period between samples, and D is the period of the input cycle.

Alt-F1 and Alt-F2 make this adjustment. The black frequency for SSTV is 1500 Hz. If you change this frequency by use of Alt-F3 and Alt-F4, you will shift the whole video range up or down. In other words, the range is defined to be from 1500 Hz (black) to 2300 Hz (white). Changing black to 1400 Hz changes the range to 1400-2200 Hz. The range (1500-2300 Hz) is 800 Hz. The program divides this range into 64 levels of gray scale, resulting in 12.5 Hz per step of gray scale. By adjusting the number of hertz per step with Alt-F5 and Alt-F6, you can experiment with the contrast to see the difference it makes.

Saving To .VOC & .PCX Formats

The received image is saved to the TEMP.VOC file. Most of the time, due to the size of this file (about 100k of data for every 8 seconds), you would not want to save the data in the .VOC format. (TEMP.VOC can be renamed if desired to save it for later playback in that format by pressing F7 from the main menu, and then F4. The program will ask for a name at that point.) Most of the time you will want to save the pictures in .PCX format. This is done by pressing F7 from the main menu, and then F1. The .PCX picture format will save the picture in a much smaller file than the .VOC format does. The pictures can be recalled later in this program or in many of the paintbrush or picture viewing programs that are available. Also, when future versions of the program allow you to send pictures, you will be able to load pictures created in the same paintbrush programs or by using one of the video capture boards on the market.

Conclusion

It has been very rewarding doing this project. I have had phone calls and letters from people who are enjoying using it. I didn't expect many people to actually send money for shareware, but they *really do!* I have even received a registration from the Canary Islands. This is my first attempt at shareware, so the results are exciting! I have many other ideas for the Sound Blaster and plan to pursue them as time permits. Some of these include weather satellite pictures, RTTY, and many of the other communication means we use with ham radio.

Thanks to Thomas P. Myers, N9LHK, for his proofreading efforts.

About the Author

Gene Harlan received his license in 1973 after many years of interest and not doing anything about it. Only after meeting Ray, W9VTL, did he get inspired and do something about it. Getting the Amateur Radio license also started a chain reaction with his wife Shari, now WB9SFT (ARRL Section Manager for Illinois), and son Shawn, KA9BXA, also getting licensed. Gene's amateur activities created enough interest in electronics for him to enroll in Rock Valley College where he received his Associate in Electronics degree. He is now Manager of

Technical Support Services at Arachnid, Inc in Rockford, IL. Gene is a past president of the Rockford Amateur Radio Association and the Experimental Amateur Radio Society.

Notes

[1] J. Langner, "Slow Scan TV—It Isn't Expensive Anymore," *QST,* Jan 1993, pp 20-30. R. Booth, "The Beat of a Different Drum: The Cop MacDonald Story," *QST,* Jan 1993, p 31.

[2] R. E. Taggart, "A New Standard for Amateur Radio Analog Facsimile," *QST* Feb 1993, pp 31-36.

[3] Creative Labs, Inc, 2050 Duane Ave, Santa Clara, CA 95054. USA Technical Support Line 408 982-9226.

[4] VGA BGI drivers for Borland *Turbo C* available from Jones Computer Supplies, RR#3, Perth, Ontario, Canada, tel 613 267- 6704.

[5] The program *SSTV for the Sound Blaster* (8-second black and white version) is available on Compuserve and other bulletin boards as shareware. On Compuserve it is located in HAMNET, Library 6 (SLOWSC.ZIP). It is also available from the author (see below) for $20 (Illinois residents add 6.25% sales tax).

[6] The program *SSTV II for the Sound Blaster* (receives Robot 8-, 12-, 24-, 36-second black and white and Robot 36- and 72-second color [displayed in black and white] and Scotty 1, Scotty 2 [displayed in black and white]) is available from the author for $39.95 (Illinois residents add 6.25% sales tax). Shipping charges on foreign orders $15, USA free. Specify 5¼" or 3½" disk. Order from Gene Harlan, WB9MMM, 5931 Alma Dr, Rockford, IL 61108, tel 815 398-2683.

[7] Requirements include an IBM PC AT or compatible with a VGA monitor capable of 640 × 480, 256 colors, minimum of 640K memory, hard drive with at least one Mbyte free, and using PC/MS DOS V3.3 or higher. A numeric coprocessor will speed up the display but is not necessary. □□