

David V. James  
Ampex Corporation  
Redwood City, CA

**Presented at  
the 69th Convention  
1981 May 12-15  
Los Angeles**



**AES**

*This preprint has been reproduced from the author's advance manuscript, without editing, corrections or consideration by the Review Board. The AES takes no responsibility for the contents.*

*Additional preprints may be obtained by sending request and remittance to the Audio Engineering Society, 60 East 42nd Street, New York, New York 10165 USA.*

*All rights reserved. Reproduction of this preprint, or any portion thereof, is not permitted without direct permission from the Journal of the Audio Engineering Society.*

**AN AUDIO ENGINEERING SOCIETY PREPRINT**

# **An Audio/Video Simulation Facility**

*David V. James*

Ampex Corp.  
Redwood City, California

## *ABSTRACT*

The digital audio/video simulation facility was constructed to aid in the development of new signal processing algorithms. A general purpose PDP-11/55\* computer is used to process data in non-real-time, but real-time evaluation of the results is supported. Software development was minimized by using the same signal processing modules for 1 dimensional (audio) and 2 dimensional (video) signal processing. All programs are written in a higher level language ('C') and can be transported to other computers using the UNIX† operating system.

The audio portion of the facility supports real time input/output of stereo audio channels at a 50kHz sampling rate. Interactive experiments involving A/B comparisons are available for evaluation of new algorithms. A tape recording illustrates the usefulness of the system.

## **1. Introduction**

Commercial audio laser disk recordings should be available to the general public within the next decade. The availability of inexpensive high quality data medium and the dramatic cost reductions in digital hardware have initiated an influx of digital signal processing technology into the audio marketplace. The change from analog to digital design technologies has created the need for a new set of design tools to assist the audio product engineer.

Computer simulations have always been useful for analog design, and extensive packages such as SPICE<sup>1</sup> have been developed for this purpose. These simulation tools have had limited success because of their complex human interface, the size of computers required for them, and the expense in CPU time to compute the results.

Digital simulation of digital signal processing algorithms is simpler and highly desirable. The equivalence between a hardware adder or register and computer operations is easily specified. The time to develop and test a complex signal processing algorithm in software is less than the time required to order, develop, and debug the hardware simulation. General purpose computers can be used to simulate a signal processing algorithms in non-real time; higher speed hardware can be built to simulate the algorithms in real-time.

The Audio/Video simulation facility at Ampex represents the first phase of tool building for the evolving digital decade. The benefits derived from this

\*DEC and PDP-11 are registered trademarks of Digital Equipment Corporation

†Unix is a Trademark of Bell Laboratories

facility include: 1. Performance evaluation of digital signal processing algorithms for products currently being designed. 2. Performance evaluation and feasibility studies for future products. The technical feasibility and practical value of noise reduction or signal modification algorithms developed by the DARPA speech community can be tested and evaluated for possible use in future product designs. 3. Software and hardware tools used in the development of the facility may be spun-off to other research and product development projects.

The Audio/Video Simulation Facility provides the power to process audio or video data. Most of the general purpose computational facilities (computer hardware and software) are shared by both applications, but only the hardware and software applicable for audio simulations are covered in this paper. More detailed coverage of the video aspects of the system is available elsewhere.<sup>2</sup>

## **2. Digital Audio Simulation Facilities**

### **2.1. Data Acquisition Hardware**

Peripheral equipment includes a high quality cassette and tuner to provide flexible input data sources, an Ampex ATR-102 audio tape recorder for high quality recording, an Apt-Holtman pre-amp, Audio Technology level meter, Brylston 2B amplifier, Stax SR-3 headphones, and Rogers LS3/5A speakers. Analog tapes and speakers are used for simple demonstrations; digital recordings are used for the most sensitive headphone based experiments.

A/D and D/A converters (CODECs) were obtained from the commercially available Ampex ADD-1 digital audio delay unit. Ampex compatible converters and backplane wiring minimized design time by maintaining compatibility with internally available hardware.

The interface hardware attached to the CODECs has programmable conversion rates, and may be operated in mono or stereo modes. The CODECs may be used for input/output to the computer, or can be used for local monitoring (A/D to D/A direct), placing no data transfer load on the host computer (a DEC PDP11/55 16 bit minicomputer). All modes are controlled by the host computer.

Design flexibility required the use of general purpose DMA interface hardware. A general purpose interface was obtained by adapting an MD3 supplied DEC equivalent DR11B DMA interface<sup>3</sup> to an adapter card with control circuitry, 128 words of first in first out (FIFO) data buffering, and line drivers and receivers for 200 foot line driving capability. This hardware adequately supports the current experiments being performed with stereo 50kHz signals.

### **2.2. Signal Processing Software**

The cost of one byte of hardware is 1-5 cents, the cost of the software to fill it is about one dollar, illustrating the importance of well planned software development. To minimize our internal software development costs, all digital audio software has been written to be transportable to other processors (such as the VAX11/780), is modular in design, and can be used for multiple channel one or two dimensional signals (audio and video).

The UNIX operating system<sup>4</sup> provided the desired tools for software development. The language "C"<sup>5</sup> is well supported, allows bit-type manipulations for detailed arithmetic simulations, and the compiler is supported on many processors. Modular programming styles are supported by the operating system, allowing a complex programming task to be broken down into a concatenation of several small simple programs.

For signal processing applications, data is transferred between signal processing tasks in signal segments consisting of a segment header followed by data (Figure 1). The header specifies the length of the data segment, number of channels, one or two dimensions, data type (8, 16, or 32 bit fixed point, 32 or 64 bit floating point, real or complex), and digital sampling frequency. This information allows modules such as the Fast Fourier Transform (FFT) to adapt to the format of the input data. This concept has functioned well; a significant number of modules have been produced for audio signal processing; many of these modules can also be used for filter design and video applications (Figure 2).

### 3. Applications

#### 3.1. Filter Design

The simulation facility provides the capability to design and evaluate digital filter structures. The first application of this capability was provided by engineering personnel investigating the effects of filter length and accuracy on the performance of the desired Finite Impulse Response (FIR) filter response. A proposed filter frequency response was as follows:

Filter Length	17
Filter Type	Band-Pass
Pass-Band Region	.16-.33 FS
Stop-Band Region	0.0-.1, .4-.5 FS
Error Weighting	Equal in all bands
(FS is the digital sampling frequency)	

Using one of the available text editors, these design parameters are encoded in the specified ASCII data format and stored in the temporary file *filspec*.

The modules *Seqfir* and *lpr* are available to solve this design problem. The module *Seqfir* uses the Parks and McClellan FIR filter design algorithm<sup>6</sup> to calculate the optimal filter parameters; the output ASCII text is normally directed to the user's computer terminal. The module *lpr* transfers the ASCII data input from the terminal to the line printer. By using the Unix "pipe" feature (the '|' symbol, these two modules can be directly connected on the command line, eliminating the intermediate terminal communication:

```
Seqfir filspec |lpr
```

This concatenation of commands produces data for evaluation (Figure 3).

Other signal module programs can be used to compute and plot the frequency response of the filter coefficients (Figure 4):

```
Seqfir -l1024 -d1.0 filspec | Sfft | Spproc -P -l -lo-40 | Splot | mp
```

In this case, the module *Seqfir* produces an output signal segment (1024 zero filled data points and digital sampling frequency normalized to 1.0). This signal segment is piped to *Sfft* (Fast Fourier Transform), a point by point processor *Spproc* (options specify power, logarithm, and a low limit of -40). The resultant data is piped next to the plot program *Splot* to calculate the graph coordinates (minimum and maximum *X* and *Y* coordinates are derived from the data and need not be specified) to produce a device independent graphic output specification. This final graphic output may be piped to the hardcopy plotter *mp* (as shown) or the graphics Tektronix terminal *tek*.

The effects of quantization of the filter parameters can be studied by rounding them before display (Figure 5):

```
Seqfir -l1024 -d1.0 flspec | Spproc -rnd16 | Sfft | Spproc -P -l -lo-40 | Splot | mp
```

In this case, the first *Spproc* module option specifies that the input should be rounded to the nearest 1/16'th fraction.

### 3.2. Analog Component Calibration

The initial CODECs were factory prototypes. Calibration of the hardware was performed using the available signal module software. The test data used for this measurement was a relatively pure digitally synthesized cosin-wave (1kHz, .5 times full scale) generated by the *Scos* module:

```
Scos -L16000 -l16000 -Ti 16384,1000,0 >d5-a
```

This generated a signal module header followed by 16000 samples of 16 bit integer cosin-wave data. The amplitude was .5 of full scale (32768), the frequency 1kHz, and the initial phase angle is zero. Output of the module was directed to the contiguous data file *d5-a*.

The pure cosin-wave can be used to calibrate the CODECs by connecting the analog output of the D/A converter to the analog input of the A/D converter.<sup>7</sup> The digital data in the computer file *d5-a* is updated with the corrupted loop-back data by typing the command line:

```
Sconvert -io d5-a
```

The option *-io* specifies that data is to be input and output simultaneously.

Conceptually, the data could be passed through the spectrum analysis programs discussed in the introduction. In practice, the convert module and the hardware insert a delay in the returned data. An option for processing the input data is:

```
Shdin -s12000 -l2048 -L2048 | Sfft | Spproc -P -l -lo-160 | plot -q | mp
```

The initial *Shdin* module seeks past the first unsettled data locations to produce a signal segment of length 2048. The resulting plot (Figure 6) illustrates the magnitude and location of the noise and harmonic distortion (about 60dB) of the experimental CODECs.

### 3.3. Algorithm Evaluation

The special hardware interfaces to the existing hardware were provided to allow real-time evaluation of various signal processing algorithms. A simple illustration of this capability is the audio demonstration comparing the perceptual errors introduced by three data compression algorithms.

To perform the demonstration, 20 seconds of audio data is converted and stored on a contiguous data file *d20-a*:

```
Sconvert -i20 d20-a
```

This data is processed by three different algorithms:

```
Spproc -rnd-256 d20-a > d20-b
```

```
Spproc -ran-256 -rnd-256 d20-a > d20-c
```

```
Spproc -Aout -Ain d20-a > d20-d
```

The first data file *d20-a* is the original data. The second data file *d20-b* contains the original data quantized to the 8 most significant bits by rounding. The third data file *d20-c* contains the original data dithered with additive white noise and rounded to the 8 most significant bits. The fourth data file *d20-d* contains the original data after encoding and decoding using the logarithmic based 8 bit A-law data compression standard.

All of the above data files are evaluated in an interactive fashion by typing the command line:

```
Sconvert -o -c d20-a d20-b d20-c d20-d
```

The options for Sconvert specify data output (*-o*), and continuous operation (*-c*). The audio data will be continuously output from the file *d20-a* repeating every 20 seconds. The source of the output data can be changed by typing one of the four keys [0,1,2,3] for selection of one of the four data files specified on the command line (*d20-a*, *d20-b*, *d20-c*, or *d20-d*). The data in the file *d20-b* contains perceptual noise and distortion due to quantization, *d20-c* has increased noise levels but no apparent distortion, *d20-d* illustrates the effectiveness of logarithmic encoding for data quantization. All of these evaluations have been performed elsewhere, but are included here for illustrative purposes.

#### 4. Summary

The facility described in this paper emphasized the development of tools for digital audio product development. The general nature of the design tools has resulted in a wide range of applicable uses including audio and video signal processing, algorithm development, and digital filter design.

The facility is adequate for preliminary investigations; the addition of a higher performance signal processor is planned. Based on our past experiences, the ideal real-time processor will have the computational bandwidth and data accuracy of the Systems Concepts processor<sup>8</sup> while maintaining the generalized architecture philosophy of the DVI<sup>9</sup> or LDSP<sup>10</sup> signal processors.

#### References

1. R. Dowell, A. R. Newton, and D. O. Pederson, *Spice VAX Version 2X.x User's Guide*, University of California, Berkeley, CA (Dec. 1979).
2. Steve Noble, "Real-Time Digital Image Simulation Facility with Application for Evaluation of Image Based Missile Guidance Systems," *Proc. of the Open Sessions of the Workshop on Imaging Trackers and Autonomous Acquisition Applications for Missile Guidance*, pp.436-450 (Nov. 19-20 1979).

3. MDB Systems Inc., *MDB DR11B Direct Memory Access Module Instruction Manual*.
4. D. M. Ritchie and K. Thompson, "The UNIX Time-Sharing System," *Communications of the ACM* 17(7), pp.365-375 (July 1974).
5. Brian W. Kernighan and Dennis M. Ritchie, *The C Programming Language*, Prentice-Hall, Inc. (1978).
6. J. H. McClellan, T. W. Parks, and L. W. Rabiner, "FIR Linear Phase Filter Design Program," pp. 5.1-1 - 5.1-13 in *IEEE Programs For Digital Signal Processing*, ed. Digital Signal Processing Committee, IEEE Press, New York, New York (1980).
7. Chin-Moh Tsai, "A Digital Technique for Testing A-D and D-A Converters," Master of Science Thesis in Computer Science, University of Utah, Salt Lake City, Utah (1973).
8. Peter R. Samson, "A General-Purpose Digital Synthesizer," *Journal of the Audio Engineering Society* 28(3) (March 1980).
9. R. I. Tucker, "LDVT Programmers Manual," Lincoln Manual 119, Lincoln Laboratory, Lexington, MA (Jan. 20, 1976).
10. P. Blankenship and V. Sferrino, "Succinct LDSP User's Guide," Internal Memorandum, Lincoln Lab, Lexington, MA (1978).

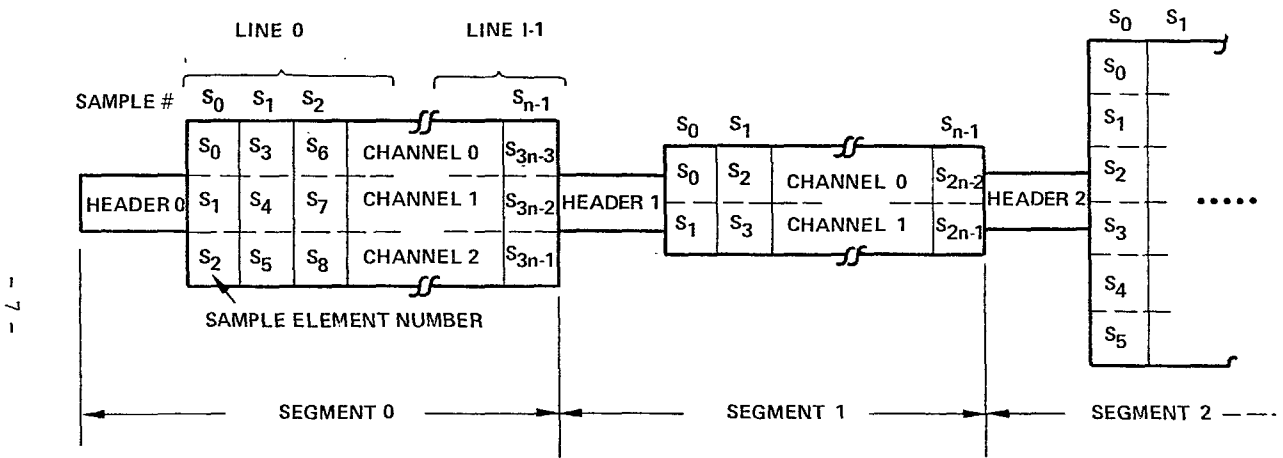


Figure 1: A Sequence of Signal Segments

### General Purpose Signal Modules (Audio/Video)

Satosig	Convert ASCII data to signal module format
Sfft	One/Two dimensional Discrete Fourier Transform
Sframe	Zero fill or data crop
Shdin	Insert header on raw data
Shdrem	Remove header from raw data
Spproc	Point by point processor
Sprint	Print signal module data
mp	Versatek hardcopy output
tek	Tektronix terminal graphical output

### One Dimensional Signal Modules (Audio)

Sconvert	A/D and D/A conversion of audio data (hardware dependent)
Sdrbtest	Test DMA interface (hardware dependent)
Seqfir	Parks and McClellan FIR filter design
Sfilt	IIR or FIR filter
Sharm	Harmonic distortion (sin-wave removal)
Snorm	Normalize signal data
Splot	Plot signal data
Swind	Window data (Hamming, Hanning, Blackman, etc)

### Two Dimensional Signal Modules (Video)

Sunlace	Separates an interlaced frame into fields
Slace	Interlaces 2 fields into a frame
Sht	Computes Hadamard transform on arbitrary subpictures
Siht	Computes inverse Hadamard transforms
Sct	Computes Cosine transforms on arbitrary subpictures
Sict	Computes inverse Cosine transforms
Sdpcmcod	Performs DPCM encoding with specified linear predictor and quantizer
Sdpcmdec	Performs DPCM decoding with specified linear predictor
Scode	Performs specified quantization scheme for each channel
Sdecode	Inverse of Scode
Sbsc	Simulates binary symmetric channel with given error rate
Shalftone	Display image on dot matrix plotter
Stv	Display image on TV monitor
Smse	Computes NMSE between two images
Shist	Computes estimate of probability density function (histogram)
Splt	Make vector plot with axes
Slaplace	Laplacian edge enhancement
Sgamma	Alter gamma
Sinterp	Zooms a decimate image with 2-D interpolation and filtering
Strans	Transpose large arrays
Smix	Mixes signal streams

Figure 2: Signal Modules for Signal Processing

```

deviation = -.001709139
deviation = -.042537073
deviation = -.060050058
deviation = -.065490037
deviation = -.066195274
deviation = -.066229255
deviation = -.066232981

```

\*\*\*\*\*

FINITE IMPULSE RESPONSE (FIR)  
Linear Phase Digital Filter Design  
Remez Exchange Algorithm

Bandpass Filter

Filter Length = 17

\*\*\*\* Impulse Response \*\*\*\*

```

H( 0) = -1.60175208e-02 = H(16)
H( 1) =  6.22392618e-05 = H(15)
H( 2) =  8.97138640e-02 = H(14)
H( 3) =  1.44461246e-05 = H(13)
H( 4) =  3.34788859e-02 = H(12)
H( 5) = -1.52493512e-05 = H(11)
H( 6) = -3.06645989e-01 = H(10)
H( 7) =  3.58045581e-05 = H( 9)
H( 8) =  4.64980036e-01 = H( 8)

```

	BAND 0	BAND 1	BAND 2
Lower Band Edge	0.0000000	0.1666700	0.4000000
Upper Band Edge	0.1000000	0.3333000	0.5000000
Desired Value	0.0000000	1.0000000	0.0000000
Weighting	1.0000000	1.0000000	1.0000000
Deviation	0.0662330	0.0662330	0.0662330
Deviation in dB	-23.5785142	0.5570422	-23.5785142

EXTREMAL FREQUENCIES--Maxima of the Error Curve

0.0000000	0.0694445	0.1000000	0.1666700	0.1944478
0.2500034	0.3055587	0.3333000	0.4000000	0.4312499

\*\*\*\*\*

Figure 3: Printout Produced by Filter Design Module Seqfir

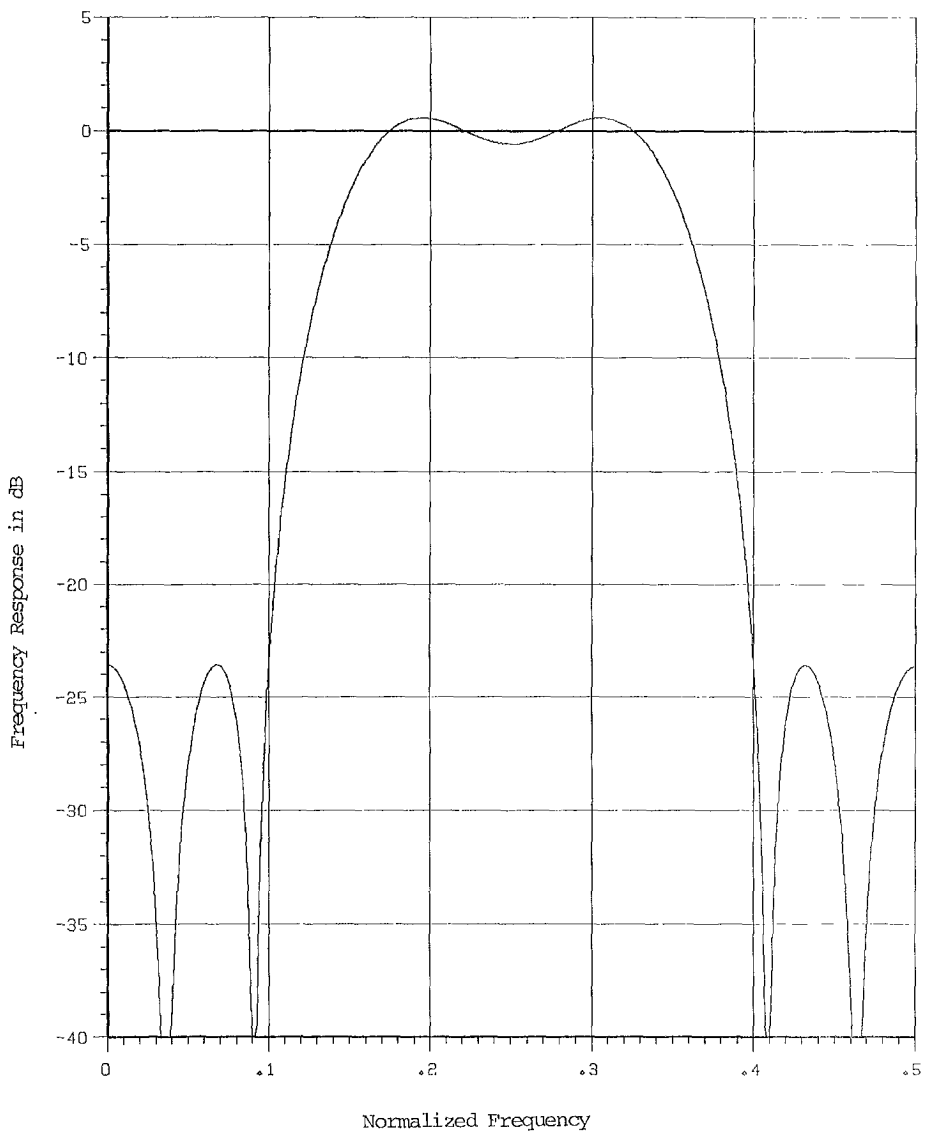


Figure 4: FIR Filter Frequency Response (Exact Coefficients)

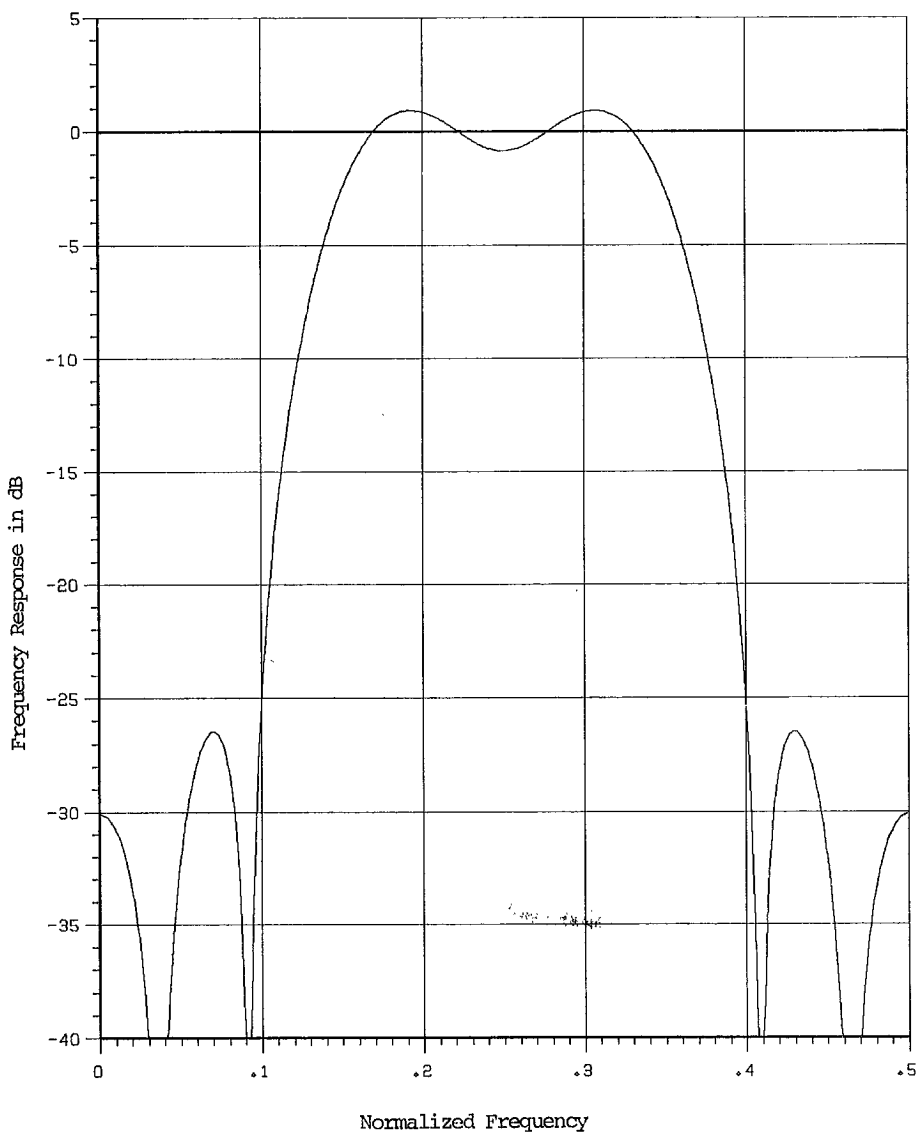


Figure 5: FIR Filter Frequency Response (Quantized Coef.)

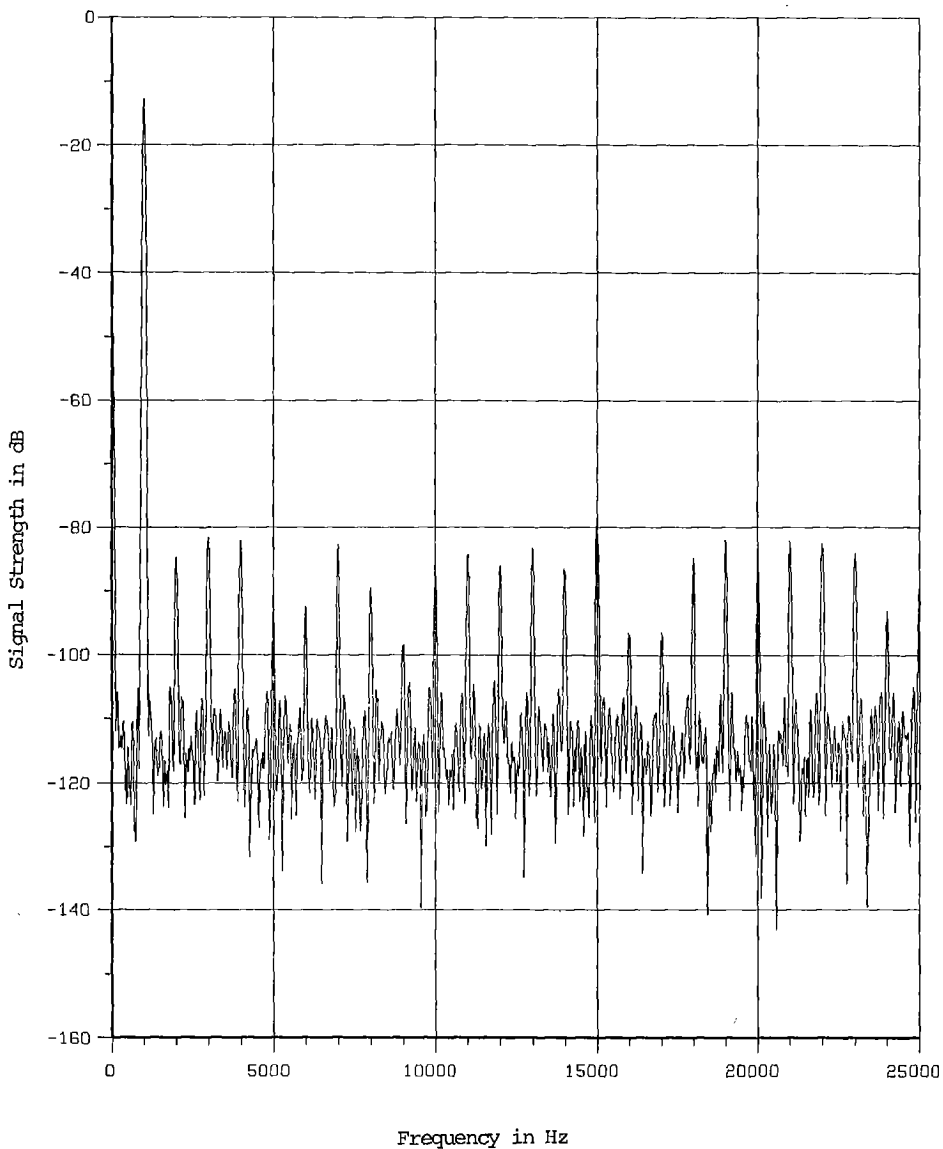


Figure 6: Spectrum of a 1kHz Sin-wave Containing Noise and Distortion