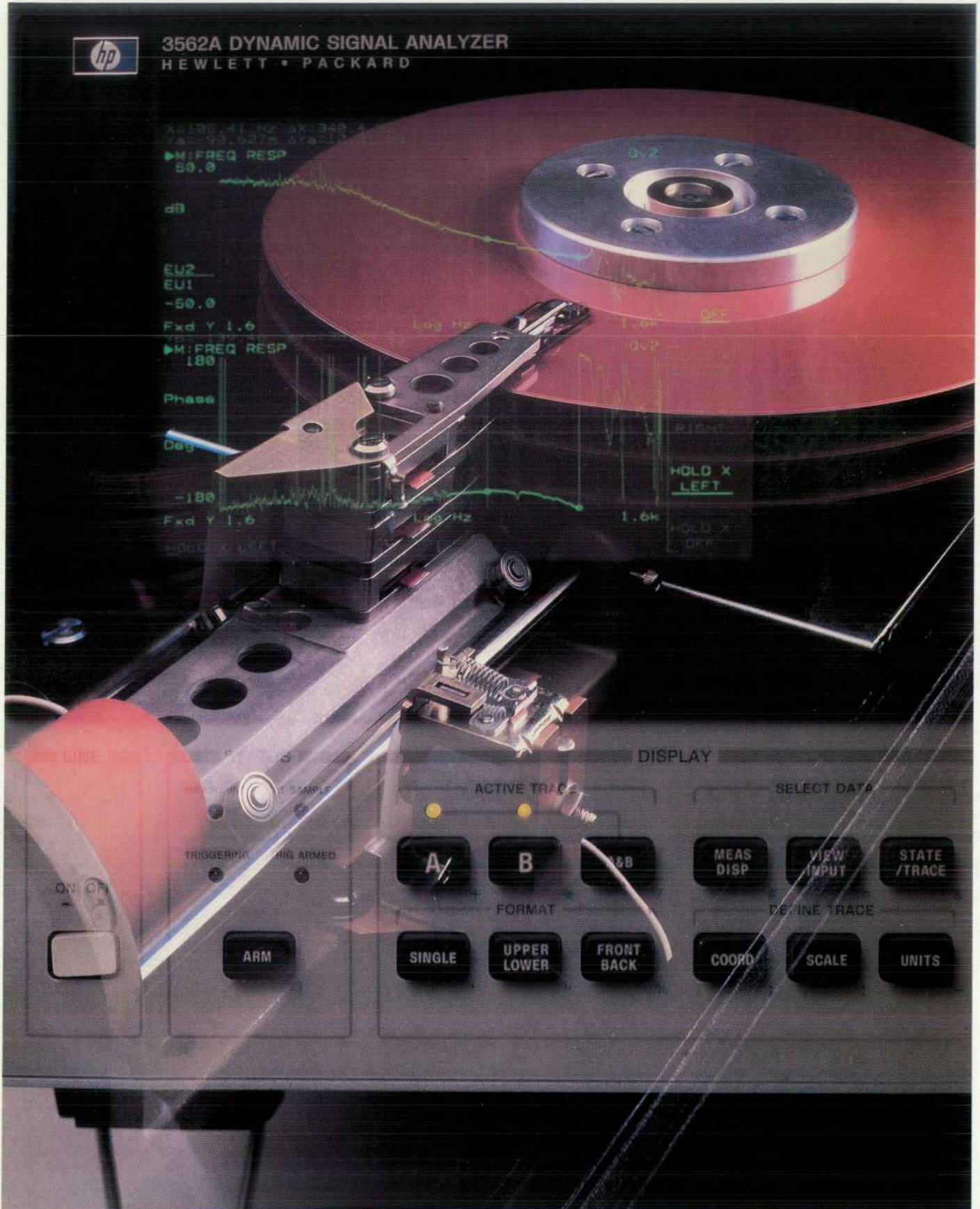


HEWLETT-PACKARD JOURNAL

JANUARY 1987



HEWLETT-PACKARD JOURNAL

January 1987 Volume 38 • Number 1

Articles

4 Low-Frequency Analyzer Combines Measurement Capability with Modeling and Analysis Tools, by Edward S. Atkinson, Gaylord L. Wahl, Jr., Michael L. Hall, Eric J. Wicklund, and Steven K. Peterson *Two-channel FFT analysis data can be stored directly on disc, curve fitted to find network parameters, or compared with synthesized responses.*

7 Applications

17 Measurement Modes and Digital Demodulation for a Low-Frequency Analyzer, by Raymond C. Blackham, James A. Vasil, Edward S. Atkinson, and Ronald W. Potter *Swept sine and linear and logarithmic resolution modes and demodulation using digital signal processing algorithms provide greater accuracy and measurement convenience.*

22 Demodulation Example

25 Analyzer Synthesizes Frequency Response of Linear Systems, by James L. Adcock *Pole-zero, pole-residue, or polynomial models can be synthesized for comparison with measured responses of low-frequency filters and closed-loop control systems.*

33 Curve Fitter for Pole-Zero Analysis, by James L. Adcock *This curve fitting algorithm can derive the poles and zeros of measured frequency responses.*

38 Performance Analysis of the HP 3000 Series 70 Hardware Cache, by James R. Callister and Craig W. Pampeyan *State-of-the-art measurements and modeling led to improved performance.*

Departments

- 3 In this Issue
- 3 What's Ahead
- 16 Correction
- 36 Authors

Editor, Richard P. Dolan • Associate Editor, Business Manager, Kenneth A. Shaw • Assistant Editor, Nancy R. Teater • Art Director, Photographer, Arvid A. Danielson
Support Supervisor, Susan E. Wright • Administrative Services, Typography, Anne S. LoPresti • European Production Supervisor, Michael Zandwijken

In this Issue



Computers controlling instruments isn't a new story anymore, nor is built-in instrument intelligence. But few who don't use them know how very intelligent some of our electronic instruments have become. Our cover subject, the HP 3562A Dynamic Signal Analyzer, is an example of this trend. It's a measuring instrument, designed for low-frequency network and spectrum analysis, but you can use it to do computer aided design (CAD) without a computer. It not only measures and analyzes, but also synthesizes and models, all by itself—no mainframe, no workstation, no PC. With just this instrument, you can do a whole linear network design:

- Decide what shape you want the network response to have and synthesize it using the HP 3562A's built-in synthesis capabilities. The instrument will fit a rational polynomial to the response curve and compute the roots of the denominator and numerator polynomials—that is, the poles and zeros of the response. From these you can choose a network topology and component values.
- Build the prototype and measure its response in any of the analyzer's three measurement modes. Compare it with the response you wanted.
- Extract the prototype's actual poles and zeros and modify the design to get closer to the desired result.
- Repeat as necessary.

On pages 4 to 35 of this issue, the HP 3562A's designers explain how it works and what it can do. Its basic functions are described in the article on page 4, and the details of its measurement modes are in the article on page 17. Unusual is the analyzer's digital demodulation capability. Give the analyzer a modulated carrier, and if the modulation is within its frequency range, it can extract and analyze it. It doesn't matter if you don't know the carrier frequency or whether the modulation is amplitude, phase, or a combination (as long as the two modulating waveforms don't have overlapping spectra). The article on page 33 reveals the theory of operation of the curve fitter and tells how so much computing power was made to fit in the available memory. The article on page 25 walks us through several examples of the use of the HP 3562A to solve realistic analysis and design problems.

The HP 3000 Series 70 Business Computer is the most powerful of the pre-HP-Precision-Architecture HP 3000s. The objective for its design was to upgrade the Series 68's performance significantly in a short time. Measurement, modeling, and verification were used to identify and evaluate possible design changes. The paper on page 38 describes the methods and how they were applied to the Series 70's cache memory subsystem, its major improvement. According to the authors, the design of a cache provides a severe test for any estimation methodology. They feel they have advanced the state of the art in cache measurement and prediction.

-R. P. Dolan

Cover

A principal use of the HP 3562A Analyzer is the design of servo systems such as the head positioning mechanisms for disc drives.

What's Ahead

The February issue will feature several articles on the design of a new family of fiber optic test instruments. The family includes three LED sources, an optical power meter with a choice of two optical heads, two optical attenuators, and an optical switch. Microwave transistor measurements using a special fixture and de-embedding techniques will also be treated.

The HP Journal encourages technical discussion of the topics presented in recent articles and will publish letters expected to be of interest to our readers. Letters must be brief and are subject to editing. Letters should be addressed to: Editor, Hewlett-Packard Journal, 3200 Hillview Avenue, Palo Alto, CA 94304, U.S.A.

Low-Frequency Analyzer Combines Measurement Capability with Modeling and Analysis Tools

HP's next-generation two-channel FFT analyzer can be used to model a measured network in a manner that simplifies further design.

by Edward S. Atkinson, Gaylord L. Wahl, Jr., Michael L. Hall, Eric J. Wicklund, and Steven K. Peterson

THE NAME FFT ANALYZER has been applied to a category of signal analysis instruments because their dominant (in some cases, their only) analysis feature has been the calculation of the fast Fourier transform of the input signals for spectrum and network response measurements. These analyzers produce an estimate of a network's frequency response function at equally spaced frequency intervals.

These FFT analyzers have justified their use for low-frequency signal analysis mainly because of their higher measurement speed when compared to conventional swept frequency response analyzers. However, proponents of swept sine analyzers are quick to point out their instruments' wider dynamic range and ability to characterize nonlinearities in a network. Proponents of 1/3-octave analyzers jump into the fray by extolling the advantages of logarith-

mically spaced spectrum analysis. These debates about which is the "best" analyzer can never really be won, since, in reality, all of these measurement techniques have their advantages depending on the application.

This fact was recognized in the design of the HP 3562A Dynamic Signal Analyzer (Fig. 1), which provides three different measurement techniques for low-frequency analysis within one instrument:

- FFT-based, linear resolution spectrum and network analysis
- Log resolution spectrum and network analysis
- Swept sine network analysis.

These measurement techniques use advanced digital signal processing algorithms that result in more accurate and more repeatable measurements than previously available with conventional analog implementations.



Fig. 1. The HP 3562A Dynamic Signal Analyzer performs fast, accurate network, spectrum, and waveform measurements from dc to 100 kHz. Measurements include power spectrum, histogram, frequency response, and cross-correlation. These can be performed in real time or on stored data. Built-in analysis and modeling capabilities can derive poles and zeros from measured frequency responses or construct phase and magnitude responses from user-supplied models. Direct control of external digital plotters and disc drives allows easy generation of hard copy and storage of measurement setups and data.

The HP 3562A has two input channels, each having an overall frequency range from 64 μ Hz to 100 kHz and a dynamic range of 80 dB.

Measurement Modes

Linear Resolution. In the linear resolution (FFT) mode, the HP 3562A provides a broad range of time, frequency, and amplitude domain measurements:

- Frequency domain—linear spectrum, power spectrum, cross spectrum, frequency response, and coherence functions
- Time domain—averaged time records, autocorrelation, cross-correlation, and impulse response functions
- Amplitude domain—histogram, probability density function (PDF), and cumulative distribution function (CDF).

A special feature in the linear resolution mode is the ability to perform AM, FM, or PM demodulation on each input channel. Traditionally, demodulation has been performed using separate analog demodulators whose outputs are connected to the test instrument. The digital demodulation technique in the HP 3562A has the advantages of higher accuracy and greater dynamic range than analog demodulation, and it is built into the test instrument.

The type of demodulation is independently selectable for each channel. For example, a frequency response measurement can be performed using AM demodulation on Channel 1 and PM demodulation on Channel 2. As another example, a two-channel power spectrum measurement can be set up in which AM demodulation is specified for Channel 1 and no demodulation is selected for Channel 2. One can easily perform two types of demodulation (e.g., AM and FM) simultaneously on the same input signal by connecting it to both channels.

A preview mode allows the user to view (and modify) the modulated input signal before the demodulation process is invoked. There is also a special demod-polar display that shows the locus of the carrier vector as its amplitude and phase vary for AM and PM signals. This display is very useful for observing possible interrelationships between AM and PM signals.

Log Resolution. In many applications, the network or signals of interest are best characterized in terms of logarithmic or proportional frequency resolution. The HP 3562A provides a true proportional resolution measurement—not simply a linear resolution measurement displayed on a log frequency scale.

In this mode, the user can make the following frequency domain measurements: power spectrum, cross spectrum, frequency response, and coherence functions. For log resolution measurements, the average quantity is always a power quantity. Stable, exponential, and peak hold averaging modes are available and offer the same benefits as in the linear resolution mode.

The user can select a frequency span from one to five decades with a fixed resolution of 80 logarithmically spaced spectral lines per decade. For example, if the start frequency is set to 1 Hz and the span is set to five decades, the frequency range of the measurement will be from 1 Hz to 100 kHz with 400 lines of resolution. Both random noise and fixed sine source outputs are available in this mode.

Swept Sine. When swept sine mode is selected, the HP 3562A is transformed into an enhanced swept frequency response analyzer. In this mode the user can make the following frequency domain measurements: power spectrum, cross spectrum, frequency response, and coherence functions. The user can select either linear or log sweep with a full range of sweep controls, including sweep up, sweep down, sweep hold, and manual sweep. During the sweep, the HP 3562A's built-in source outputs a phase-continuous, stepped sine wave across the selected frequency span and a single-point Fourier transform is performed on the input signal.

There are four key setup parameters associated with the sweep for which the user can either set fixed values or specify an automatic mode of operation. These parameters are input range, integration time, source output gain, and frequency resolution.

By judiciously selecting these automatic sweep features, the user can perform a measurement in which the sweep adapts dynamically to meet the requirements of the device under test.

A discussion about the technical basis behind the HP 3562A's measurement modes and digital demodulation capability is given in the article on page 17.

Advanced Data Acquisition Features

Normally, measurements are made on-line using the data currently being acquired from the input channels. However, the HP 3562A provides two modes in which data can be acquired and then processed off-line at a later time.

Time Capture. In this mode, up to ten time records (20,480 sample points) from either channel can be stored in a single buffer. This data can be acquired in real time for any frequency span up to 100 kHz. The user can display a compressed version of the entire time buffer or any portion of it in the time or frequency domain. Furthermore, the time capture buffer can be used as an input source for any of the single-channel measurements available in linear resolution mode.

If a measurement is set up with the same frequency range as the time capture data, then up to ten averages can be done. At the other extreme, a measurement with one average can be made with a frequency span that is a factor of 10 narrower than the original time capture frequency range. For example, if a time capture is performed with a span from dc to 10 kHz, the user can perform a power spectrum measurement on this data within a 1-kHz span centered anywhere from 500 Hz to 9.5 kHz. Several display options (e.g., expansion, scrolling, etc.) are available for manipulating time capture data.

Time Throughput. The HP 3562A is special among low-frequency analyzers in offering the capability to throughput time data directly from the input channel(s) to an external disc drive. No external HP-IB (IEEE 488/IEC 625) controller is needed for this operation.

The instrument can throughput data to an HP CS/80 hard disc drive (e.g., the HP 7945) at a real-time measurement span of 10 kHz for single-channel operation and 5 kHz for dual-channel operation. The throughput data session can be used as an input source for both linear resolution (including demodulation) and log resolution measurements.

As in the time capture mode, zoomed measurements can be made on real-time throughput files. However, much narrower zoom spans are possible since a throughput file can be much larger than the internal time capture buffer.

The user can conveniently view the data in the throughput file by stepping through the file one record at a time. Measurements do not have to start at the beginning of the file—an offset into the file can be specified for the measurement start point.

Modeling and Data Analysis Capabilities

Given the extensive measurement capability described above, the design of the HP 3562A could have stopped there—as a versatile test instrument. However, combining this measurement performance with equally extensive built-in analysis and design capabilities turns the product into a one-instrument solution for many applications.

Flexible Display Formats. Generally, the result of a measurement process is the data trace on the screen. Graphical analysis is simplified by the HP 3562A's full complement of display formats. The independent X and Y markers (both single and band) and the special harmonic and sideband markers make it easy to focus on important regions of data. The annotated display and the special marker functions (e.g., power, slope, average value, THD, etc.) allow simple quantitative analyses to be performed directly on the displayed data.

Waveform Calculator. The instrument includes a full complement of waveform math functions including add, subtract, multiply, divide, square root, integrate, differentiate, logarithm, exponential, and FFT. The operands (either real or complex) for the math functions can be the displayed trace(s), saved data traces, or user-entered constants. A powerful automath capability allows the user to specify a sequence of math operations to be performed on any of the standard data traces while the measurement is in progress. The user's custom measurement result can then be displayed at any time by simply pressing the automath softkey, which can be relabeled to indicate the name of the function. Some examples include group delay, Hilbert transform, open-loop response, and coherent output power (COP). Automath is discussed in more detail later in this article.

S-Plane Calculator. The HP 3562A can synthesize frequency response functions from either pole-zero, pole-residue, or polynomial coefficient tables. In addition, the user can convert from one synthesis table format to another with a single keystroke. As an example, a designer can enter a model of a filter in terms of numerator and denominator polynomial coefficients and then convert to pole-zero format to find the roots of the polynomials. This s-plane calculator is a powerful network modeling tool and it exists within the same instrument that will be used to test and analyze the actual network—providing on-screen comparison of predicted and measured results. See the article on page 25 for more details about the HP 3562A's synthesis capabilities and some design examples.

Curve Fitter. One of the most powerful analysis features in the HP 3562A is a multidegree-of-freedom, frequency-domain curve fitter for extracting the poles and zeros of a network from its measured frequency response function. The curve fitter can fit up to 40 poles and 40 zeros simul-

taneously for the entire response function or any portion defined by the display markers. In addition, the table of poles and zeros generated by the curve fit can be transferred to the synthesis table—a direct link between the instrument's modeling and analysis features. A curve-fit algorithm that can only fit clean data is of little practical use since most real-world measurements are contaminated by some amount of noise. The HP 3562A curve fitter removes biases caused by measurement noise, resulting in a greatly reduced noise sensitivity and, therefore, more accurate estimates of the pole and zero locations. See the article on page 33 for more details.

Hardware Design

A block diagram of the HP 3562A's hardware system is shown in Fig. 2.

Input Amplifiers. Spectrum analysis for mechanical and servo applications requires that the analyzer's input amplifiers provide ground isolation (usually to reject dc and ac power-line frequency signals). Ground isolation at low frequencies can be achieved by running the input amplifiers on floating power supplies with the supply grounds driven by the shields of the input cables. This floating ground can then be used to drive a guard shield that encloses the floating amplifier's circuitry. This design is effective at rejecting dc common-mode signals and provides a large common-mode range, but is limited in its ability to reject higher-frequency common-mode signals.

Fig. 3 illustrates the problem. Common-mode signal V_{cm} is dropped across the voltage divider formed by the source impedance Z_s (which could be the resistance of the input cable) and the capacitance C between the floating ground and chassis ground. The voltage drop across Z_s is then measured by the analyzer just as if it were any other differential-mode signal V_{dm} . It can be seen that things get worse with increasing frequency. The fundamental problem here is that truly floating the amplifier is not practical since C cannot be made arbitrarily small in practice and the inputs are not balanced with respect to the chassis. In addition, since the source impedances to the two input terminals may not be quite the same, it is desirable to minimize the common-mode input currents by providing a high-impedance path for both signal inputs.

An effective way of providing high impedance for both inputs and good balance is to use a true differential amplifier (Fig. 4). For a perfect amplifier with a gain of -1 and all resistors shown of equal value, the common-mode rejection (CMR) will be infinite (neglecting parasitic components). An added benefit is that common-mode rejection is achieved without requiring floating power supplies, which would be prohibitively expensive in a multichannel instrument. The input amplifiers in the HP 3562A are based on this true differential amplifier topology with circuitry to provide calibration of CMR and bootstrap circuits to increase common-mode range and reduce distortion.

The two input paths in each of the HP 3562A's amplifiers contain programmable (0 dB, 20 dB, or 40 dB) attenuators, buffer amplifiers, and other circuits that contribute to imbalances in the two signal paths. To remove these imbalances (and improve CMR), a programmable circuit is used to adjust the gain of the low signal path. As shown in Fig.

Applications

The HP 3562A covers a broad range of applications in control systems, mechanical systems, low-frequency electronics, and acoustics.

Design of Closed-Loop Control Systems

The general development process is shown in Fig. 1. A particular application of this development process is the design of closed-loop control systems. The design activity involves selecting components that, when connected in a specified topology, produce system performance consistent with the design specification. Closely related to this activity is the modeling activity—developing equations to predict system performance. These equations are often transfer function relationships expressed as ratios of polynomials in the complex frequency domain.

The design and modeling activities are simplified using the HP 3562A because of its built-in s-plane calculator and transfer function synthesis capability (see article on page 25). A designer can enter system information directly into the analyzer in pole-zero, pole-residue, or polynomial format. Conversion between these formats is possible. Once the system information is entered, the analyzer will compute and display the frequency response of the system. The result can be displayed in gain/phase form, as a Nyquist diagram, or as a Nichols chart. The designer can then easily perform other calculations on the synthesized waveform. For example, the inverse transform of the frequency response can be displayed with one keystroke to identify the impulse response of the system. By pressing another key, this waveform can then be integrated to simulate the step response.

The test activity involves making measurements to determine actual system performance. Designers requiring both FFT and swept sine measurements can use the HP 3562A to perform both functions. The HP 3562A provides not only frequency domain test capabilities, but also time domain test functions. Its waveform recording features allow the control system designer to measure time domain functions such as impulse, step, and ramp responses.

The analysis activity studies measured performance. In cases where desired performance, such as gain-phase margin, overshoot, or undershoot has been estimated, analysis consists of comparing measured results against the expectation. This comparison can be as simple as reading marker values or using the

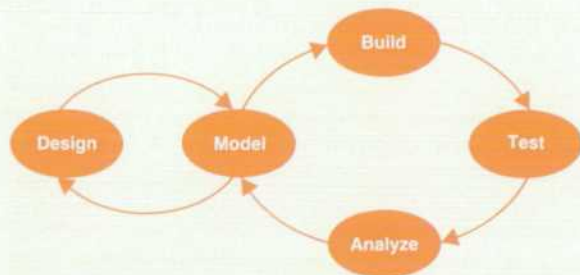


Fig. 1. Development process flowchart.

5, this is accomplished by increasing the gain through the low path by raising the value of R_d and then driving the grounded end with a signal that is $-K \times B$, where K is the gain of a multiplying digital-to-analog converter (DAC). Increasing the DAC gain reduces low-path gain to the point where CMR is maximized.

An algorithm for optimum adjustment of the DAC was developed that requires only two data points to set the gain

HP 3562A's front and back display format.

Often designers are constrained to measure control systems under operating conditions with the loop closed. Although the closed-loop response is measured, the open-loop response is desired. The HP 3562A provides an analysis function that can compute the open-loop response directly from closed-loop data with just one keystroke.

An extremely powerful analysis tool for the control system designer is the HP 3562A's curve fitter (see article on page 33), which extracts system poles and zeros from measured frequency response data for comparison against expected values. Furthermore, the curve fitter provides the link between the analysis and model activities in the development process, because of its ability to pass curve-fit data to the transfer function synthesis function.

Vibration Analysis

Vibration measurements on rotating machinery are extremely important. In many cases, the vibration signals of interest are modulation signals on a carrier frequency which corresponds to the shaft rotation speed. As an example, a broken tooth on a rotating gear can result in amplitude-modulated vibration signals. In a belt-driven pulley system, the dynamic stretching of the belt can produce frequency-modulated or phase-modulated signals.

The HP 3562A can perform AM, FM, or PM demodulation on these vibration signals even if the carrier frequency (shaft speed) is unknown.

Electronic Filter Design

All of the major modeling, test, and analysis features can come into play in solving filter design problems (see article on page 25 for examples). The initial filter model (i.e., transfer function) can be entered into the synthesis table in the form most familiar to the designer and then the model's frequency response function can be synthesized. Log resolution measurements or log-sine sweeps are appropriate for measuring broadband filter networks. High-Q notch filters can be accurately characterized using zoomed linear-resolution measurements or narrow sine sweeps. By applying the HP 3562A's curve fitter to these linear or log frequency measurements, the poles and zeros of the actual filter can be extracted for comparison with the model.

Audio and Acoustics

Noise identification and control are becoming more important in many environments. With its two input channels, the HP 3562A can make acoustic intensity measurements to determine sound intensity and direction. In the audio field, the instrument's digital demodulation capability has proven to be very effective in analyzing modulation distortion in phonograph pickup cartridges and loudspeaker systems.

Bibliography

1. *Control System Development Using Dynamic Signal Analyzers*, Hewlett-Packard Application Note 243-2.

correctly. A common-mode signal is internally connected to both differential inputs and the DAC is set to each of its extreme values. The relationship between the two resulting measured signal levels is then used to interpolate to obtain the optimum DAC gain setting. The resulting CMR is better than -80 dB up to 66 Hz and -65 dB up to 500 Hz.

One-megohm-input buffer amplifiers are inserted between the inputs and the differential amplifier stage to

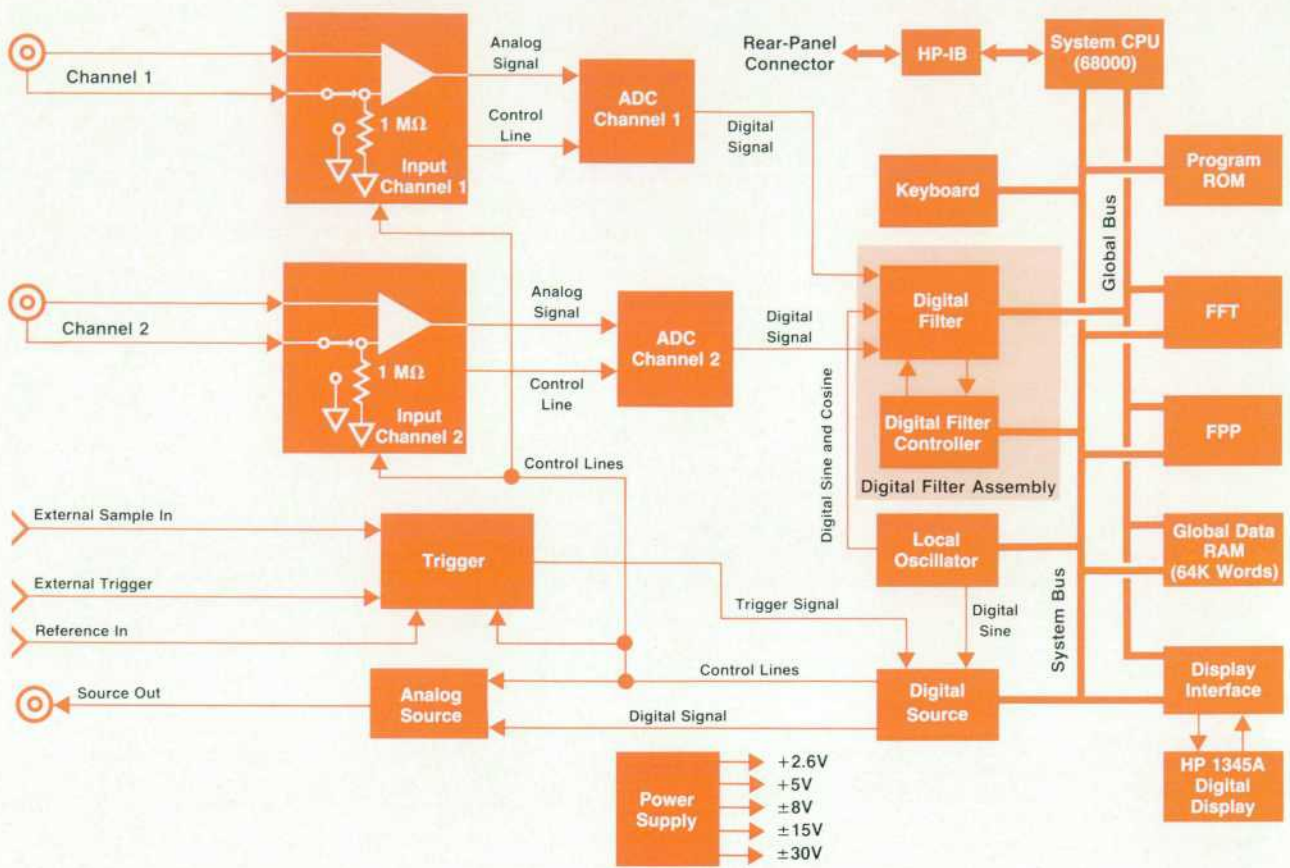


Fig. 2. Hardware block diagram of the HP 3562A.

provide impedance conversion. The common-mode range of these amplifiers is increased and distortion is reduced by adding bootstrap amplifiers that drive the buffer amplifier power supplies (Fig. 5). These bootstrap amplifiers are unity-gain stages with $\pm dc$ offsets added to the outputs so that the buffer amplifier supplies follow the ac input voltage. Thus, the input buffer amplifiers only need to provide correction to the signals present within their supply rails. In addition, the op amps used in these amplifiers are therefore able to work over a common-mode

range that would otherwise be beyond their maximum supply rail specification. The resulting common-mode range for the HP 3562A is $\pm 18V$ on the most-sensitive range (-51 dBV) and distortion is suppressed more than 80 dB below full scale.

Digital Circuitry. For certain analyzer frequency spans, the data collection time (time record length) will be shorter than the data processing time. As a result, part of the input data will be missed while the previous time record is being processed. In this case, the measurement is said to be not real-time. As the frequency span is decreased, the time record length is increased, and a span is reached where the corresponding data collection time equals the data processing time. This frequency span is called the real-time bandwidth. It is at this span that time data is contiguous

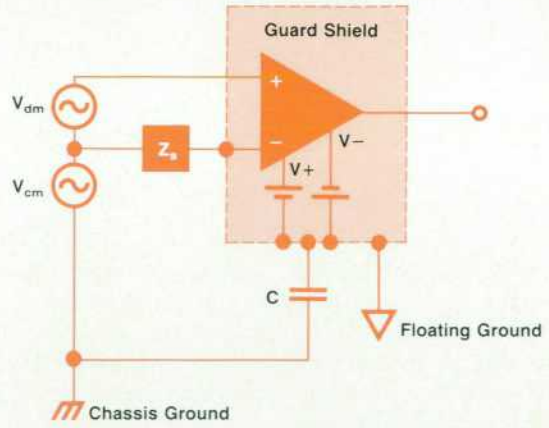


Fig. 3. A floating input amplifier.

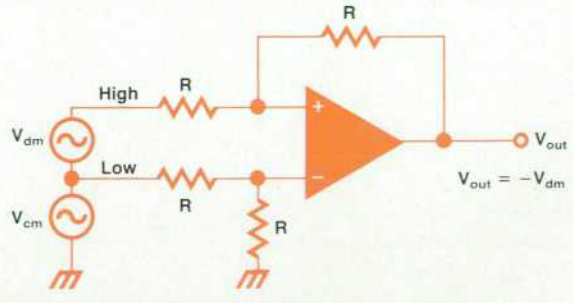


Fig. 4. True differential amplifier.

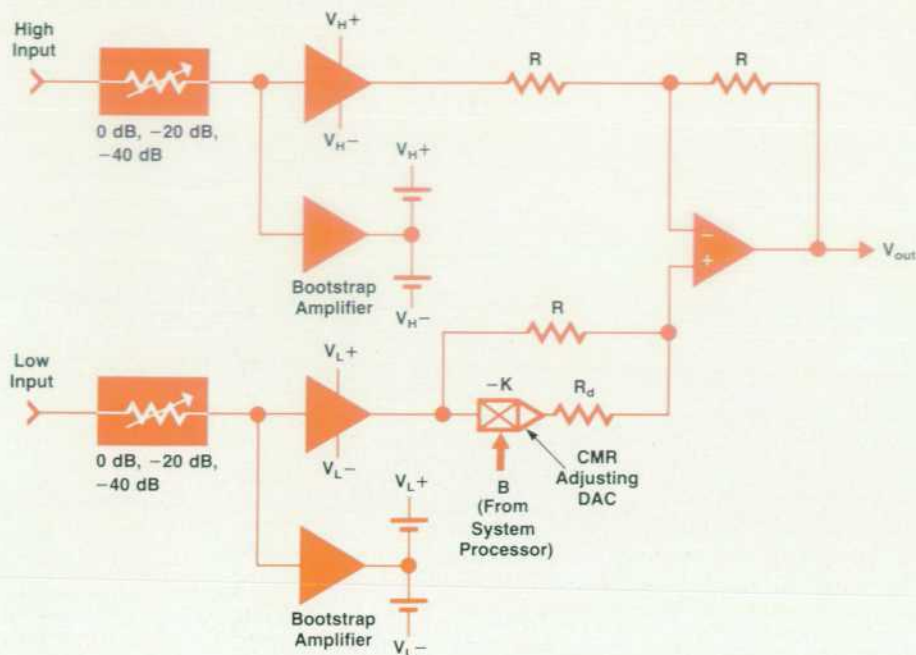


Fig. 5. Input amplifier configuration used in the HP 3562A.

and all input data is processed. Decreasing the span further results in data collection times longer than the data processing time. These measurements are said to be real-time.

Designing for a real-time bandwidth of 10 kHz (single channel) requires data processing times of less than 80 ms. It was apparent that a single processor could not perform the many computational tasks (fast Fourier transforms and floating-point calculations in particular) that are required within this amount of time. Consequently, the choice was made to develop separate hardware processors for these computational needs. The large real-time bandwidth of the HP 3562A is possible because of the use of multiple hardware processors and a dual-bus digital architecture (see right half of Fig. 2).

Before pipelined parallel data processing can begin, the input signals must be digitized and filtered to the current frequency span. Signals enter the analyzer and are conditioned by the programmable-gain, differential-input amplifiers described above. Low-pass anti-aliasing filters with a cutoff frequency of 100 kHz attenuate unwanted frequency components in the conditioned signal. The signal is then converted to digital data by analog-to-digital converters (ADCs) using the same design that was used in the earlier HP 3561A.¹ The data is then filtered to appropriate frequency spans by custom digital filters. Separate digital filters exist for each input channel.

The digital filters provide the gateways for data into the digital processing portion (Fig. 2). The architecture includes a system central processing unit (CPU), an FFT processor, a floating-point processor (FPP), a shared memory resource (global data RAM), an interface to the display, and two separate digital buses that allow simultaneous communication and data transfer.

The CPU controls a data processing pipeline by issuing commands, reading status, and servicing interrupts for the digital filter controller, FFT processor, FPP, and display

interface. The CPU also services the digital source and front-end interface, local oscillator, keyboard, and HP-IB. The CPU design consists of an 8-MHz MC68000 microprocessor, a 16K×16-bit ROM, a 32K×16-bit RAM of which 8K×16 bits is nonvolatile, a timer, HP-IB control, power-up/down circuitry, and bus interfaces. The CPU communicates with each of the hardware processors as memory mapped I/O devices over the system bus, which is implemented as a subset of the 68000 bus.

The other bus in the architecture, the global bus, provides a path to the global data RAM. This memory is used by all of the hardware processors for data storage. After the data has been digitized and filtered to the current frequency span, the digital filter stores it into the global data RAM. The CPU is signaled over the system bus when a block of data has been transferred, and as a consequence, the CPU instructs the FFT processor to perform a time-to-frequency transformation of the data. The FFT processor accesses the global data RAM, transforms the data, stores the result back into memory, and then signals the CPU. The CPU now commands the FPP to perform appropriate floating-point computations. The FPP fetches operands from the global data RAM, does the required calculations, and then stores the results back into memory. When finished, the FPP signals the CPU. The CPU then reads the data, and performs coordinate transforms on it as well as format conversions in preparation for display. The data is again stored into the global data RAM. The CPU finally instructs the display interface to transfer the data to the display. (The display used is the HP 1345A Digital Display, which requires digital commands and data to produce vector images.)

The above description describes one data block as it is transferred from digital filter to display. However, the HP 3562A Analyzer does not process one block of data at a time, it processes four. That is, while display conversions of block N-3 are being done by the CPU, block N-2 is

being operated on by the FPP, block N-1 is being transformed by the FFT processor, and block N is being filtered by the digital filter. Simultaneous operation of all processors provides the computational speed necessary to produce a large real-time bandwidth of 10 kHz.

As a consequence of the data processing architecture, the global memory system had two major requirements. The first was that multiple processors needed access to the memory. This required an arbiter, a decision maker, to monitor memory requests and allocate memory accesses (grants) based on a linear priority schedule. Devices are serviced according to their relative importance, independently of how long they have been requesting service. Each requesting device has associated with it a memory request signal and a memory grant signal. When a device needs access to the memory, it asserts its memory request. The global data RAM prioritizes all requests and issues a memory grant to the highest-priority device. That device is then allocated one bus cycle.

The second major requirement placed on the global memory system was minimal memory cycle time to satisfy instrument real-time bandwidth needs. Computer simulations were done to model the effects of memory cycle time on real-time bandwidth. It was concluded that cycle times less than 500 ns would satisfy the instrument's real-time bandwidth requirements.

To optimize memory cycle time, the timing for the global memory system is constructed from a digital delay-line oscillator. Three digital delay lines are connected in series, with the output of the third delay line inverted and connected to the input of the first delay line. Individual timing signals were customized for the 64K \times 16-bit dynamic RAMs and arbiter using combinational logic operating on signals from 10-ns delay taps on the delay line oscillator. As implemented, global memory cycles are available every 470 ns.

The FFT processor can perform both forward and inverse fast Fourier transforms and windowing. It is designed around a TMS320 microprocessor. Although an on-board ROM allows the TMS320 to execute independently, the FFT processor is slaved to the CPU. Commands are written to the FFT processor over the system bus, and the FFT processor accesses the global data RAM for the data to be transformed and any user-defined window information. Operations are executed, and the CPU is interrupted upon completion. The FFT processor computes a 1024-point, radix-4 complex FFT in 45 ms.

The FPP is constructed from six AM2903 bit-slice microprocessors using conventional bit-slice design techniques. It can operate on three different data formats: two's complement integer (16 bit), single-precision floating-point (32 bit), or double-precision floating-point (64 bit). Besides addition, subtraction, multiplication, and division, the FPP can perform 81 customized operations. A list of FPP instructions, called a command stack, is stored in the global data RAM by the CPU. The list consists of a 32-bit command word (add, subtract, et cetera), the number of entries in the data block to be operated on, constants to indicate if the data block is real or complex, the beginning address of the data block in the global data RAM, and the destination address for the results. The FPP is then addressed by the

CPU and the starting address of the command stack is given. The FPP executes the command stack and interrupts the CPU upon completion.

The digital filter assembly consists of two printed circuit cards. One card, the digital filter board, contains two sets (one per input channel) of custom integrated circuits designed for digital filtering. These integrated circuits were leveraged from the design of the HP 3561A.² The other card, the digital filter controller board, supplies local information to the digital filter board and contains the interface to the system bus. Digitized data is supplied to the digital filter from the ADCs at a 10.24-MHz rate. Upon command from the CPU, the digital filter can operate on the data in different ways. It can pass the data directly to the global data RAM (as is required if the user wants to view time data). It can frequency translate the data by mixing it with the digital equivalent of the user-specified center frequency, filter unwanted image frequencies (required in a zoom operation which enables the full resolution of the analyzer to be concentrated in a less than full span measurement), and store to the global data RAM. Or it can filter the data without any frequency translation and store to the global data RAM (required in the baseband mode where the start frequency is 0 Hz and the span is less than full span). The data can be operated on simultaneously in different modes, thus providing the analyzer the ability to view input data simultaneously with frequency data. The design is the most complex digital assembly in the HP 3562A, and is implemented using a number of programmable logic devices, direct memory access (DMA) controllers, and large-scale-integration counters.

Throughput. Throughput means acquiring input data, filtering it, processing trigger delays, and writing the data to the disc as fast as possible. To accomplish this, the HP 3562A has built-in disc drivers for HP Command Set 80 (CS/80) disc drives (e.g., HP 7945 and HP 7914), HP Subset 80 (SS/80) disc drives (e.g., HP 9122 and HP 9133D), and other HP disc drives such as the HP 9121, HP 9133XV, HP 82901, and HP 9895. These disc drives can be used for general save and recall functions in addition to throughput. The drives most suited for throughput are the CS/80 or SS/80 hard disc drives. The software disc drivers are tuned for these drives.

The throughput is accomplished with three internal processors to pipeline the transfer process (see Fig. 6). The digital filter assembly digitally filters the data for a channel into a rotating buffer with enough room for two time records. This allows one time record to be accumulated while the other record is moved out of the buffer. When the instrument is acquiring data in triggered mode, the digital filter assembly is putting the data in the rotating buffer even before the trigger is received. The data before the trigger is needed if a pretrigger delay is specified. Since the start of data (trigger point if no trigger delay) can occur anywhere in the rotating block, the full time record will not be contiguous in memory if the start of data is more than halfway into the buffer. The first part of the data will be at the end of the buffer and the last part of the data will be at the beginning of the buffer (i.e., the data is wrapped).

The next step is for the FFT processor to move and unwrap the data into another buffer. This buffer is one of a

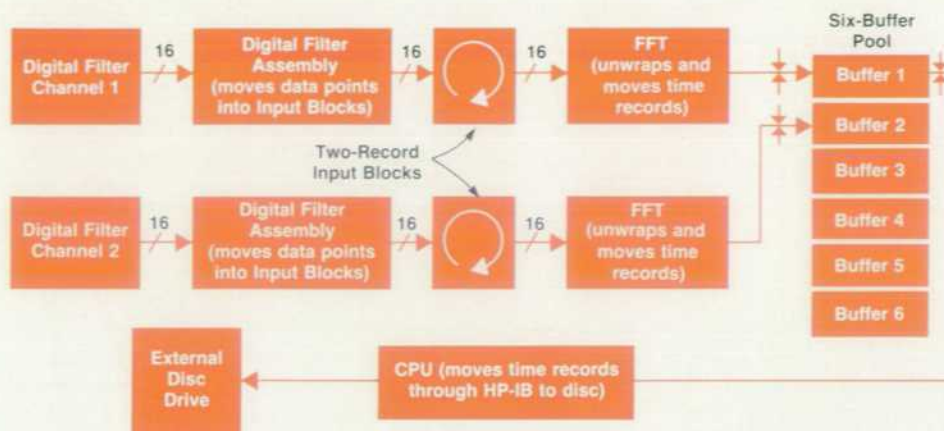


Fig. 6. Throughput data flow during a data acquisition to disc storage. (All processors are running simultaneously.)

set of six rotating buffers. The FFT processor is fast enough to move an entire time record out of the input block before the digital filter assembly overwrites it, except for a 100-kHz frequency span, which is faster than real-time anyway. The CPU moves the contiguous data in time record blocks through the HP-IB to the disc drive. To cut down on the disc overhead, the entire throughput operation is set up as a single disc write command. The six buffers help to smooth the nonlinear transfer rate to the disc.

The limit on the transfer rate is usually the CPU handshaking data out of the HP-IB chip. The CPU executes a tight assembly language loop to transfer data (about 60 kbytes/s). Most hard discs can receive data faster than this. This transfer rate translates into a 10-kHz real-time frequency span for a one-channel measurement.

Since throughput is essentially a real-time operation, the digital filter channels must never stop transferring data to memory, or gaps will appear in the data. If both channels are being throughput to the disc, then the data is interleaved on the disc in time record increments (Channel 1 followed by Channel 2). If both channels are being throughput in triggered mode, then both channels are triggered at the same time regardless of trigger delay. This allows both channels to be synchronized on time record boundaries. If the user specifies a cross-channel delay of greater than one time record, then an integral number of time records are thrown away from the delayed channel rather than wasting space on the disc with unused data. Hence, with the same number of records throughput to the disc in this mode, there will be a sequence of the undelayed channel records, followed by zero or more interleaved records (Channel 1 followed by Channel 2), followed by the remainder of the delayed channel records.

Since the throughput is normally done with one write command to the disc, the disc will fill an entire cylinder before moving the disc heads to the next cylinder (head step). Each cylinder consists of multiple heads, each on a separate surface (i.e., a track). The time to step the head one track is not long. If the track has been spared (i.e., because of a bad spot on the disc), the disc head automatically steps to the replacement logical track, which could be halfway (or more) across the disc. This could adversely affect the real-time data rate. To avoid this problem, the instrument does not use any logical track that has been

spared (replaced) for throughput. There will be a gap in the logical address space on the disc, but skipping the spared track only requires two head seek times. The spared track table can only be read from a CS/80 disc drive. If a spared track is in the throughput file used, then one disc write transfers data before the spared track, and another write is used for data after the spared track. The HP 3562A can skip up to nine spared tracks in each throughput file (identified in the throughput header) in this manner.

Any throughput data in the file can be viewed by the user as a time record or used as a source of data for a measurement. When performing a measurement of the throughput data, the data is input back into the digital filters for possible refiltering. This allows data throughput at the maximum real-time span (10 kHz, one channel) when test time is valuable, and then performing the measurement later at a desired lower span (e.g., 2 kHz). Since throughput data is processed off-line, more flexibility is allowed on this data than may be possible with a normal on-line measurement. Data from anywhere in the throughput file can be selected for postprocessing (i.e., a selected measurement) by specifying a delay offset when measuring the throughput. For a measurement on a real-time throughput file, the user can specify the exact amount of overlap processing to be used during averaging. Increased flexibility is realized by being able to continue a measurement from anywhere in the throughput file, which allows a measurement on the throughput data in a different order than it was acquired.

Operating System

The operating system used by the HP 3562A can be described as a multitasking system with cooperating processes. The operating system provides task switching, process creation, process management, resource protection (allocation), and process communication.

The operating system manages the use of processes. A process is a block of code that can be executed. In a multitasking system, several processes can be executed simultaneously. Since there is no time slicing (using a timer interrupt to split CPU time between processes), cooperation between processes is essential. To share CPU resources between processes, explicit suspends back to the operating system must be performed about every 100 ms.

Since our development language (Pascal) is stack oriented, there could be a problem with multiple processes. Does a process share a stack with another process? How do they share it? Because of demands on the CPU RAM (i.e., local data, states, et cetera), the stack RAM space was in short supply, so we chose to implement a hybrid stack management system.

We started implementing a system with only one stack instead of a stack for each process. This was accomplished by making one restriction upon the software designers. Breaks to the operating system (suspends and waits) could only be made in the main procedure in a process, which was not unreasonable for our original code expectations. Our Pascal development language allocates the procedure variables on top of the stack upon procedure entry and afterward references these variables with a base address register. The stack pointer is not used to reference the variables. The top of stack can be moved without affecting the access of the variables in the main procedure. This allows each process to have its own stack for its main procedure that is exactly the right size, and also a (possibly separate) stack for procedure calling. When a process ends, a call back to the operating system records the stack segment for the main procedure as being empty so that it can be reused.

This method of stack allocation presented problems when the code size grew significantly (to 1.1 Mbytes), but the RAM grew only slightly (to 64K bytes). Some processes became larger and the restriction on waits became unwieldy. Therefore, a second method was implemented of allocating entirely separate stacks (partitions) for some processes (e.g., calibration and HP-IB control). There was not enough RAM to do the same for all processes, so a hybrid system was maintained. A partition concept was defined to allow a single process to use a partition (allocated at power-up) or one of a group of processes to use a partition at any one time (e.g., one of the HP-IB processes). The addition of partitions allowed us to gain the maximum utility out of the limited RAM available.

The operating system is semaphore based. The classic Dijkstra $p()$ and $v()$ operations³ are used for resource protection and process communication. These operations have been implemented as counting semaphores with the only allowable operations of wait (corresponds to $p()$) and signal (i.e., $v()$). A counting semaphore can be defined as a positive integer (may never become negative).

A signal is defined as an increment operation. A wait is defined as a decrement operation where the result may not be negative. If the decrement would cause the result to be negative, then the process execution is blocked until the semaphore is incremented (i.e., a signal is sent by some other process), so that the decrement can be successful.

The actual implementation of a semaphore is different. The integer does become negative on a wait and the process is blocked in a queue of waiting processes associated with each semaphore (semaphore wait queue). A signal where the previous value before the increment was negative will unblock the first process in the wait queue (i.e., it can run again). In this sense, a semaphore count, if negative, represents the number of processes waiting on the semaphore.

There are generally two types of semaphore use: resource protection and process communication. A resource protec-

tion semaphore is initialized with a value that is the number of resources available (generally one). An example of this in the HP 3562A is the floating-point processor; only one process can use it at a time. A process uses a resource semaphore by executing a wait before the block of code that uses the resource, and a signal after that block of code. All processes that use this resource must do the same thing. The first process executing the wait decrements the semaphore from 1 to 0 before using the resource and afterward increments (signals) the semaphore back to 1. If a second process tries to wait on the resource after the first process has acquired the resource (wait), then the second process will be blocked (it cannot decrement 0 to -1) until the first process has signaled (released) the resource. A semaphore is also used to protect critical sections of code from being executed by two processes simultaneously.

A process communication semaphore (initialized to 0) is used by two processes that have a producer/consumer relationship. The producer signals the semaphore and the consumer waits on the semaphore. An example of a consumer in the HP 3562A is the marker process, which waits for a marker change before moving the marker on screen, while the producer is the keyboard interrupt routine, which detects that a marker change is requested and signals the marker process.

In many cases, a process communication semaphore is not enough. The consumer must know more about what is to be done. Therefore, a communication queue which has one entry for each signal by the producer is associated with the semaphore. This combination is known as a message queue. An example is the keyboard interrupt routine (producer) and the front-panel process (consumer). A queue containing key codes is associated with the semaphore.

Queues, a first-in-first-out construct, are also supplied by the operating system. As mentioned above, queues are mainly used to communicate between processes, but are also used by the operating system (i.e., the process ready queue) and other processes to keep a list of information. Another variant of the queue definition is the priority queue, where an addition-to-a-queue operation inserts new entries in order by a priority key. The operating system process ready queue is an example of a priority queue in the HP 3562A. The priority is set by the scheduling process to determine the relative execution order of several processes.

Command input to the HP 3562A is from three processes representing the keyboard, autosequence, and HP-IB. Each process accepts codes from the appropriate source, converts them to a common command function format, and invokes the command translator. The command translator controls the main user interface, the softkey menus, command echo field, numeric entry, and translation of the command function into a subroutine call to do the requested operation. The command translator is table driven. A single data base of all keys is maintained with all related softkey menus and actions to be performed when a key is pressed. A single command translator presents a consistent interface to the user, regardless of the source of the commands.

The following is an example of the power of a multitasking operating system. It shows how four processes can be run at the same time within the HP 3562A. There is a measurement running on one trace, a view input on another

trace, the display process itself, and a plot process. To see the results, press:

PRESET RESET (put HP 3562A in known state and start measurement)
UPPER LOWER (display both traces)
B VIEW INPUT INPUT TIME 1 (start view input in other trace)
PLOT START PLOT (start plot process)

The plotting is performed in parallel with the measurement and view input. This is called "plot on the fly." Resource semaphores are used to protect hardware from simultaneous use by several processes. Communication semaphores are used to communicate display status to data taking processes. Buffer lock flags are used to protect the display buffers from use by both the display and the plot.

In summary, the use and implementation of the operating system in the HP 3562A show the trade-offs that occurred during project development. The operating system has enough power and flexibility to implement otherwise very difficult operations (e.g., plotting during a measurement), but is simple enough not to impose much overhead. Less than 2% of the processor time is spent in the operating system.

Autosequence

An autosequence is a programmable sequence of key presses that can be invoked by a single key, which makes repetitive operations easier for the user. It is the simple programming language built into the HP 3562A. In the learning mode (edit autosequence), all completely entered command lines, from hard key to terminating key, are remembered in the autosequence. The line-oriented editor allows line deletion, addition, and replacement. A looping primitive and a goto command allow repetitive commands. Other autosequences can be called as subroutines. The autosequence can be labeled with two lines of text, which will replace the key label that invokes the autosequence.

Up to five autosequences in the instrument can be stored in battery-backed CMOS RAM. Since CMOS RAM is limited, an implementation was chosen that takes minimal memory to store an autosequence. The storage mechanism saves either the keycode (0 through 69) for the key pressed (only takes one byte) instead of the HP-IB mnemonic (4 bytes), or a unique key function token (over 900 functions defined by 2 bytes). This allows storage for an autosequence of 20 lines with an average of 10 keys/line. The command strings seen in the autosequence edit mode are only stored in a temporary edit buffer (not stored in CMOS RAM). When the edit mode is engaged again, the stored keys are processed through the command translator without executing the associated actions to rebuild the command strings. The commands are shown on the screen as they are being rebuilt.

The softkey menus of the HP 3562A may vary when the measurement mode changes. For example, in swept sine mode, there is a softkey STOP FREQ under the FREQ key. In linear resolution mode, the ZERO START softkey is in the same location (softkey 4). Key codes (e.g., softkey 4) are stored in the autosequence. If an autosequence that has this key sequence in it is run in both modes, a different

action could take place (zero start versus stop frequency). This is not what the user expects. To solve this problem, any time a variant softkey menu is displayed because of a key press, the variant menu number is stored in the autosequence following the key. When the autosequence is edited, the variant menu is forcibly displayed so that the user always sees the same command echo regardless of the measurement mode. When the autosequence is executed, the actual key function (e.g., stop frequency) is obtained by looking up the softkey number in the current softkey menu. If the key function is found, the new keycode is executed instead of the old keycode. This allows softkey functions to move in the menu in different modes. If the key function is not found, an error results and the autosequence stops. The additional overhead for this function is one byte for each variant menu displayed in the autosequence. Although the storage overhead is not very significant, the software to execute the autosequence became more complex.

While executing the autosequence, the front-panel keys are locked out (except **AUTOSEQ PAUSE ASEQ**). This is necessary because there is only one base translator with three command sources (keyboard, autosequence, and HP-IB), and only one current menu can be remembered at a time.

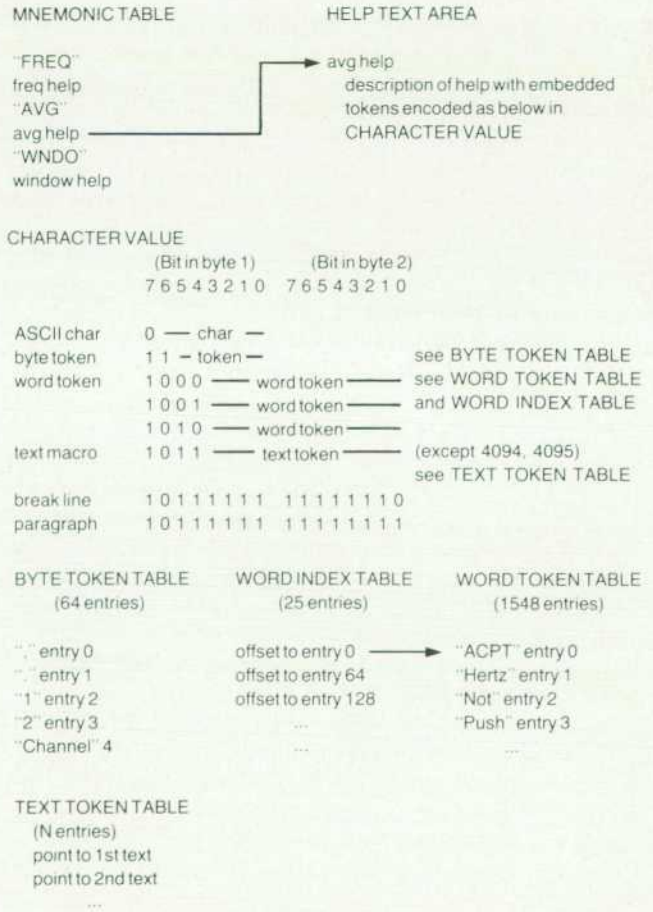


Fig. 7. Help table structure (top), definitions of two-byte character values (middle), and examples of token table entries (bottom).

For instance, from the keyboard, the user presses **WINDOW UNIFORM** (softkey 3), and from autosequence, **SELECT MEAS FREQ RESP** (softkey 1). If the key presses were interleaved between the two command sources when the functions are processed, then the commands would appear to be **WINDOW SELECT MEAS AUTO CORRELATION** (softkey 3). The wrong functionality would be invoked in this case. To prevent this situation, these two processes are synchronized and the autosequence lets the HP-IB commands execute only between autosequence lines. Each autosequence line always starts with a hard key (i.e., it has no memory of a previous softkey menu).

Autosequence commands may spawn child processes to do specific actions (e.g., a measurement). The front panel will continue to accept a new command before the child process is finished, but the autosequence will wait until the child process is finished before executing the next line. The required synchronization is provided by the operating system, which keeps track of the status of the children of the command front ends and communicates the information through a semaphore to the command source (e.g., autosequence).

Help Text Compression

The HP 3562A has a help feature for 652 soft and hard keys in the instrument. Help information is derived from the HP-IB mnemonic for the key pressed. The help display consists of information derived from the key table (full command name, type of key, number and type of parameters) and a text description of the key. The text description can take the entire display area (48 columns by 20 lines). If the average text description per key is five lines, then just the text portion of the help feature would consume 156,000 bytes. The actual help text for the HP 3562A takes 157,125 characters (bytes).

To save ROM space, the text is compressed before putting it in ROM (see Fig. 7). The method used is a token replacement of duplicate words used in the text. The two-pass compression program reads the input text on the first pass, breaks it into words, and updates an occurrence count for each unique word (29,509 unique words). At the end of the first pass, the words are sorted by their occurrence frequency. Token tables are created for multiple-use words to be replaced in the text during the second pass.

The ASCII character set (unexpanded) is a seven-bit code. To encode the tokens in the output text, the high-order bit (bit 7) is set in the byte. This allows 128 tokens to be encoded in a byte, which is not enough. A word token allows more tokens (32,767), but takes twice the memory to represent the output text. To compromise, both byte tokens and word tokens are used. The 64 most-used words are encoded into a byte token (bit 6 set) and 32,767 other text functions can be encoded into a word token.

The word text functions are split into two groups: word tokens and special functions. Special functions have bits 4 and 5 set, allowing 4095 functions to be encoded and leaving 12,288 possible word tokens (1548 actually used). The special functions are split into two groups: special formatting commands and text macros. The text expander in the HP 3562A also formats the displayed text. The text in the help table is in free format, just words separated by

spaces and punctuation. The expander knows how to left-justify text on the screen and break lines on word boundaries. In addition, it knows two special formatting commands: break line (go to next line) and start paragraph. The last of the special functions that are encoded are text macros. Groups of words can be defined and included in multiple places (useful to describe numbers). There is space for up to 4094 text macros. The 64 most-used byte tokens save almost as much space as the 1548 word tokens (45,189 bytes versus 55,363 bytes).

To save as much ROM space as possible in the help table, several concessions to size, space, and speed were made. To make the pointers to text shorter, 16-bit unsigned offsets are used instead of 32-bit pointers in the help mnemonic table (converts HP-IB mnemonics to text addresses), the text macro table, and the word token index table. Later, the help mnemonic table had to be changed to 32-bit pointers because the size of the tables and the help text had exceeded 64K bytes (the limit of a 16-bit offset). Since words vary in size, the byte and word token tables are organized as a linked list. Hence, they must be traversed to access any given token. This is acceptable for the 64 byte tokens, but takes too long for the 1548 word tokens. Therefore, a word token index table was created to index into every 64 word tokens, keeping the worst-case lookup sequence to 64 entries.

Compressing text to save room can be a good investment. In the HP 3562A, the text is compressed down to 36% of its original size through word replacement, saving 100,000 bytes of ROM.

Compression may be a good idea, but how long does it take to compress and decompress the text? The compression into assembler code takes approximately six hours on an HP 1000 Computer. However, the decompression of a worst-case full page takes less than a second, so it is acceptable for a help display.

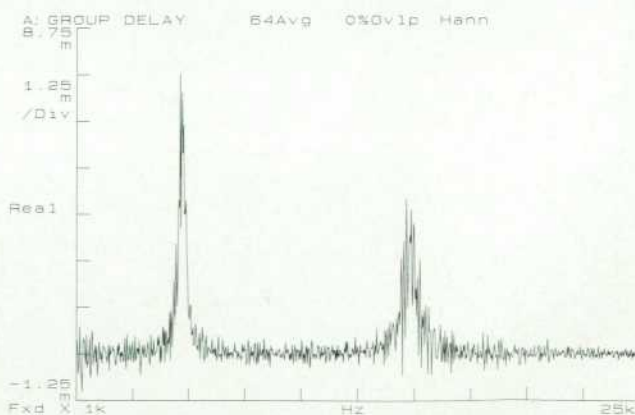


Fig. 8. Automath plot for group delay calculation.

Automath

The HP 3562A includes a rich set of math operations available to the user to postprocess the measurement data. In addition, automath adds the ability to perform calculations on the fly during a measurement for custom measurement displays (e.g., group delay). Automath also adds the ability to title a custom measurement display on the trace.

In many cases the automath calculation does not slow down the measurement appreciably. This is because most math operations (except log and FFT-type operations) are performed by the floating-point processor (FPP) in parallel with most operations in a measurement.

Automath is in reality an autosequence that is limited to math, measurement display, and active trace keys. When the **MEAS DISP AUTO MATH** keys are pressed, the automath autosequence is run to produce a sequence of math operations and the initial measurement displays are set. The measurement converts the sequence of math operations into a stack of FPP commands and appends it to its own FPP stack. This stack is retained for use on every measurement average; the stack is only rebuilt if something changes (e.g., a reference waveform is recalled). Considerable time is saved in this optimal case. For operations that the FPP does not execute directly, there is a pseudo FPP command that causes the FPP to stop and interrupt the CPU, which interprets and executes the command and then restarts the FPP upon completion. Since the math commands are generated once, a copy of the changed state of the data header must be saved during generation to be restored on every data average while the FPP computes the automath.

Fig. 8 shows the resultant traces generated by automath for group delay where group delay = $-\Delta\text{phase}/\Delta\text{frequency}$. Math is applied to the complex data block, not the coordinates that are displayed (e.g., magnitude and phase), so the phase of the frequency response must be expressed in complex form. The natural log of the complex data will put the phase information in the imaginary part of the data. Multiplying by $0 + j1$ results in negative phase being in the real part of the data. The real-part command clears the imaginary part of the data, resulting in negative-phase information only. The differentiate command completes the group delay by differentiating with respect to frequency. Fig. 9 shows the phase of the frequency response measurement used by automath for the plot in Fig. 8.

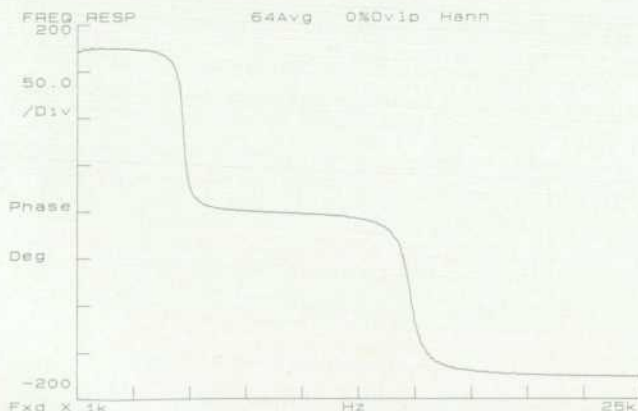


Fig. 9. Plot of phase for frequency response in Fig. 8.

Autocalibration

The HP 3562A calibration software has two parts. First, a calibration measurement routine calculates complex-valued correction data for the two input channels. These calibrations can run automatically to account for temperature-induced drift in the input circuits, including the power-up temperature transient. Second, correction curves (one for each display trace) are generated on demand for use by the measurement software. These correction curves combine the input channel correction data with analytic response curves for the digital filters to provide complete calibration of the displayed measurement results.

The HP 3562A input channels have four programmable attenuators.¹ The first, or primary, attenuator pads (0 dB, -20 dB, and -40 dB) each have significantly different frequency response characteristics. As a result, separate calibration data is required for each primary pad. Since the HP 3562A measurement software can perform both single-channel and cross-channel measurements, it may demand any of 15 distinct correction curve formats (see Fig. 10). Rather than collect 15 sets of data during calibration, six sets are carefully chosen so that the remaining nine sets can be derived.

The first choice for these basic six sets would be the single-channel calibrations A_{00} , A_{20} , A_{40} , B_{00} , B_{20} , and B_{40} (the subscripts refer to a particular pad of the primary attenuator). The HP 3562A single-channel calibration hardware and techniques are adapted from the HP 3561A.⁴ Two classes of errors can be identified in this technique. One class of errors has identical effects in both channels, and therefore has no effect on the accuracy of a cross-channel correction derived from single-channel corrections. Finite-resolution trigger phase correction is an example of such an error term. The other class of errors has different effects in each channel, and therefore will degrade the accuracy of derived cross-channel corrections. Slew-rate limiting of the pseudorandom noise calibration signal by the input channel amplifiers is an error of the second class. While the cumulative errors in this second class are small with respect to the single-channel phase specification (± 12 degrees at 100 kHz), they are too large to provide state-of-the-art channel-to-channel match performance.

Therefore, the basic six calibration data sets must allow cross-channel corrections to be derived with no contribution from single-channel terms. The highlighted elements of Fig. 10 identify the basic six sets. A typical relationship between derived and basic cross-channel corrections is:

$$B_{40}/A_{00} = [B_{40}/A_{20}][B_{00}/A_{00}]/[B_{00}/A_{20}]$$

where the square brackets enclose stored terms.

Single-channel correction data is derived from single-channel and cross-channel terms. For example:

$$B_{40} = [A_{00}](B_{40}/A_{00})$$

where B_{40}/A_{00} is derived as shown above.

The five basic cross-channel corrections are, like the single-channel correction, measured during the calibration routine. The periodic chirp source is connected internally to both channels for a bare-wire frequency response mea-

surement, a direct measure of input channel match. The periodic chirp provides even distribution of energy over the whole frequency span of interest. It is also synchronized with the sample block duration to eliminate leakage effects.

Now consider the corrections for the second, third, and fourth input attenuators (secondary pads). The primary pad correction data already contains corrections for the particular secondary pads engaged during primary pad calibration. Thus, secondary pad correction must account for changes in the input channel frequency response when different secondary pads are engaged. These relative secondary pad corrections are negligible when compared with the single-channel specification. However, their effects are significant for channel-to-channel match performance.

A pair of bare-wire channel-to-channel match measurements is used to generate relative calibration data for the secondary pads. One channel-to-channel match measurement is used as a reference—the absolute correction for its secondary pad configuration is contained within the primary pad data. In the other channel-to-channel match measurement, one secondary pad is switched in one channel. The ratio of these two channel-to-channel match measurements is the relative correction for the switched pad with respect to the reference pad. For example, consider a reference configuration of 0-dB primary pad (subscript 00) and 0-dB, 0-dB, 0-dB secondary pads (subscript 0,0,0) in both channels. Increment the last attenuator in Channel 2 to 2 dB for a configuration of 00-0,0,2. Then the relative correction for the -2-dB pad in the last attenuator of Channel 2 is calculated from:

$$(B_{00-0,0,2}/A_{00-0,0,0})/(B_{00-0,0,0}/A_{00-0,0,0}) = B_{00-0,0,2}/B_{00-0,0,0}$$

For an arbitrary secondary pad configuration, the relative corrections for each attenuator are combined. The relative secondary pad corrections can be modeled over the 0-to-100-kHz frequency span as a magnitude offset and a phase ramp. With this simplification, secondary pad calibration data can be measured at one frequency using the HP 3562A's built-in sine source. The combined secondary pad correction is then a magnitude-offset/phase-ramp adjustment to the primary pad correction. The resultant correction curve produces channel-to-channel match accuracy of ± 0.1 dB and ± 0.5 degree.

Acknowledgments

The HP 3562A is the end result of an extensive project that started at HP's Santa Clara Division and culminated at Lake Stevens Instrument Division. We would like to acknowledge those contributors at Santa Clara Division: Webb McKinney, Terry Donahue, Dick Fowles, and Don Langdon provided direction during the early phases of the project. Jim Dunlay, Al Gee, Al Helgelson, Chuck Lowe, and Doug White contributed to the initial hardware design. Barbara Grote, Lani Maragay, and Mary Ryan contributed to the initial software design. At Lake Stevens Instrument Division, many others made important contributions toward the project completion. Those designers who contributed during release to production were John Elm, switching power supply, Alan Henshaw, mechanical design, Jan Hofland, digital filter assembly, Larry Sanders, analog-to-dig-

	B ₀₀	B ₂₀	B ₄₀
A ₀₀	B ₀₀ /A ₀₀	B ₂₀ /A ₀₀	B ₄₀ /A ₀₀
A ₂₀	B ₀₀ /A ₂₀	B ₂₀ /A ₂₀	B ₄₀ /A ₂₀
A ₄₀	B ₀₀ /A ₄₀	B ₂₀ /A ₄₀	B ₄₀ /A ₄₀

Fig. 10. Single-channel and cross-channel primary attenuator pad correction curve variations. B_{40} represents the reciprocal of the frequency response of Channel 2 with the -40-dB pad engaged. B_{40}/A_{20} represents the correction curve for a cross-channel measurement with primary attenuator pads specified for each channel (40 dB in Channel 2, 20 dB in Channel 1). The six shaded values are stored in the HP 3562A and the other nine values are derived from the stored values.

ital converter, and Dave Shoup, local oscillator and trigger.

Bryan Murray implemented much of the display and synthesis software. Marty Bender implemented the diagnostics software, Praful Bhansali contributed to the demodulation software, and Peter Pias contributed to the waveform math software and FPP microcode. Frank Robey, Dave Rasmussen, Bill Brown, and Steve Kator contributed to automated testing.

Rick Kunin was the production engineer and Randy Eilert was the industrial designer. Larry Whatley was the program manager and Nick Pendergrass was one of the project managers. Also, Helen Chase and Sherry Clark provided key project support. Special thanks to all those involved in the final testing of the software and to the project teams' spouses for their understanding.

References

1. J.S. Epstein, et al, "Hardware Design for a Dynamic Signal Analyzer," *Hewlett-Packard Journal*, Vol. 35, no. 12, December 1984, pp. 12-17.
2. C.R. Panek and S.F. Kator, "Custom Digital Filters for Dynamic Signal Analysis," *ibid*, pp.28-36.
3. A.C. Shaw, *The Logical Design of Operating Systems*, Prentice-Hall, 1974.
4. G.R. Engel and D.R. Hiller, "Instrument Software for Dynamic Signal Analysis," *Hewlett-Packard Journal*, Vol. 35, no. 12, December 1984, pp. 17-19.

CORRECTION

In the December 1986 issue, the second language listed in Fig. 5 on page 7 should be Canadian French.

Measurement Modes and Digital Demodulation for a Low-Frequency Analyzer

by Raymond C. Blackham, James A. Vasil, Edward S. Atkinson, and Ronald W. Potter

THE HP 3562A DYNAMIC SIGNAL ANALYZER provides three different measurement modes for low-frequency spectrum and network analysis from 64 μ Hz to 100 kHz within one instrument with two input channels and a dynamic range of 80 dB:

- Swept sine
- Logarithmic resolution
- FFT-based linear resolution.

These measurement modes use advanced digital signal processing algorithms to provide more accurate and more repeatable measurements than previously available with conventional analog circuit approaches.

Swept Sine Mode

The swept sine measurement is based on the Fourier series representation of a periodic signal. Consider a stimulus signal:

$$s(t) = \sum_{n=-\infty}^{\infty} c_n e^{j\omega n t}$$

applied to the device under test (DUT) and a response signal:

$$r(t) = \sum_{n=-\infty}^{\infty} d_n e^{j\omega n t}$$

where c_i and d_i are complex numbers, and $c_i = c_{-i}^*$ and $d_i = d_{-i}^*$ so that $s(t)$ and $r(t)$ are real signals. Ideally, $s(t)$ is a perfect sine wave with perhaps a dc offset, or $c_n = 0$ for $n > 1$. The frequency response of the DUT at frequency $f = \omega/2\pi$ Hz is defined to be d_1/c_1 . The swept sine measurement is a series of measurements of d_1 and c_1 for different frequencies.

The HP 3562A assumes that $s(t)$ is connected to its Channel 1 input and $r(t)$ is connected to its Channel 2 input. c_1 is calculated using the standard Fourier series integral:

$$c_1 = \frac{1}{T} \int_0^T s(t) e^{-j\omega t} dt,$$

where $T = 2\pi/\omega = 1/f$. d_1 is calculated in the same way using $r(t)$ in place of $s(t)$.

The HP 3562A carries out this calculation on both input signals as follows (given for $s(t)$ only):

1. The signal is sampled at intervals of $T_s = 1/f_s$, with $f_s =$ the sample frequency. Let $s_n = s(nT_s)$ for $n = 0, 1, 2, \dots$

2. s_n is multiplied by $e^{-j\omega n T_s}$ in real time using a digital local oscillator.

3. Samples spanning the time $(N+1)T_s$, where N is a positive integer, are used in a numerical integration algorithm to calculate an approximation to:

$$\frac{1}{MT} \int_0^{MT} s(t) e^{-j\omega t} dt$$

where M is a positive integer, $MT \leq (\text{user-entered integration time}) < M(T+1)$, and $(N-1)T_s \leq MT < NT_s$.

4. $|c_1|^2$, $|d_1|^2$, and $d_1 c_1^*$ (the trispectrum) are calculated.
5. Steps 3 and 4 are repeated the number of times specified by the user-entered number of averages.
6. The values of $|c_1|^2$ thus calculated are averaged together and stored. The same is done for $|d_1|^2$ and $d_1 c_1^*$ (trispectrum average).

Integration Algorithm. Since:

$$\int_0^T e^{j\omega(n-1)t} dt = \begin{cases} T & \text{for } n = 1 \\ 0 & \text{for } n \neq 1 \end{cases}$$

we can simplify:

$$\begin{aligned} \frac{1}{T} \int_0^T s(t) e^{-j\omega t} dt &= \frac{1}{T} \int_0^T e^{-j\omega t} \left(\sum_{n=-\infty}^{\infty} c_n e^{j\omega n t} \right) dt \\ &= \frac{1}{T} \sum_{n=-\infty}^{\infty} c_n \int_0^T e^{j\omega(n-1)t} dt \end{aligned}$$

and we obtain:

$$\frac{1}{T} \int_0^T e^{-j\omega t} s(t) dt = c_1$$

as desired.

The integral can be evaluated separately for the real and imaginary parts. However, if the integral is not evaluated exactly, the terms for which $n \neq 1$ will not cancel completely and will contribute an error to the estimate of c_1 . This error could be particularly severe if there is a large dc offset or significant harmonic components in $s(t)$. The integration algorithm is designed to guarantee that any contribution from a component c_i of $s(t)$ for $i \neq 1$ to the integral estimate is a factor of 3×10^4 (90 dB) less than $|c_1|$. For example, if $c_1 = 1$ and $c_0 = 10$, the resulting estimate of c_1 would be

affected by the c_0 term by a factor of no more than $10/(3 \times 10^4) \approx 3 \times 10^{-4}$, or -70 dB.

The numerical integration is performed using a fifth-order composite quadrature integration formula as follows. Consider six adjacent samples of $s_n e^{-jn\omega T_s}$. Find the fifth-order polynomial passing through these six points. Integrate this polynomial between the middle two points. Now move everything forward, in time, one sample (overlapping five samples) and repeat the procedure (see Fig. 1).

The interval over which this polynomial is integrated is contiguous with the previous integration interval. The sum of these two integrals is an approximation of the integral of the continuous signal $s(t)e^{-j\omega t}$ over the two sample periods. Continuing, we can cover as large an interval as desired that is a multiple of the sample period T_s . This integration method is similar to Simpson's method of numerical integration (also a composite quadrature formula), which integrates a second-order polynomial passing through three adjacent samples over two sample periods as shown in Fig. 2. Note that the fifth-order formula used in the HP 3562A uses two points beyond each end of the integration interval.

Now, to allow integration over an arbitrary interval rather than a multiple of the sample period T_s , a separate quadrature integration formula is calculated to integrate over a portion of a sample period at the end of the integration interval, as shown in Fig. 3. This last quadrature formula depends on the value of MT , the interval of integration of the Fourier series integral, which, in turn, depends on the frequency of the signal component we are trying to estimate. Therefore, this formula must be recalculated for each point in the frequency sweep of the measurement. Fortunately, it is a relatively simple calculation. For more information, many numerical analysis texts such as reference 1 are available containing sections on general composite quadrature integration methods.

This numerical integration method, when applied to a signal of the form $e^{jn\omega T_s}$, loses accuracy as the frequency ω increases. For example, using this method to estimate the value of:

$$\frac{\omega}{2\pi} \int_0^{2\pi/\omega} e^{j\omega t} dt = 0$$

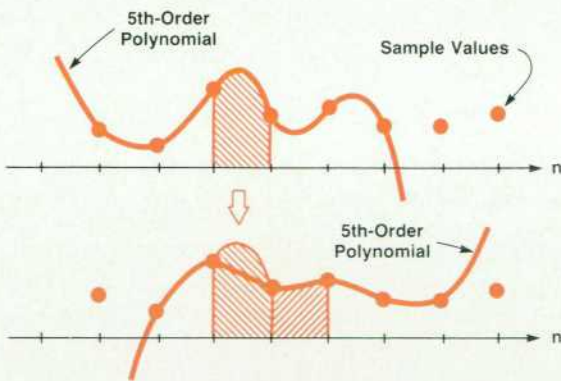


Fig. 1. Fifth-order polynomial integration.

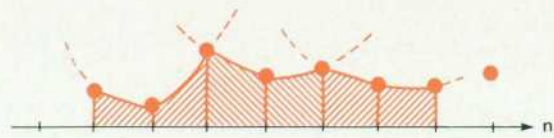


Fig. 2. Second-order polynomial integration using Simpson's method.

results in a value with magnitude less than 3×10^{-5} (90 dB below the signal level) for frequencies such that there are at least ten samples per cycle, or for $T_s < 0.1(2\pi/\omega)$. Hence, limiting the frequency components of the incoming signal to $0.1f_s$, where f_s = the sample frequency, would guarantee a sufficient amount of rejection of the terms in the Fourier expansion of $s(t)$ for which $n \neq 1$. If we used Simpson's rule instead, this limit would be considerably lower.

To attenuate the errors caused by terms in $s(t)$ above this frequency limit, the signal $s_n e^{-jn\omega T_s}$ is passed through a short low-pass FIR (finite impulse response) filter with a gain of 1.0 at dc. This FIR filter is of the same form as the quadrature integration formulas, and can be combined with them as if they were two consecutive FIR filters. The result is a modified composite quadrature integration formula which incorporates the attenuation characteristics of both the fifth-order composite quadrature formula and the FIR filter, and therefore, will attenuate contributions of terms in the expansion of $s(t)$ by at least 90 dB for all such terms between dc and $f_s/2$. The form this formula takes is a dot product between samples of the input signal and a vector of coefficients as illustrated in Fig. 4.

The number of coefficients in the leading and ending portions of the coefficient vector is the same for all integration intervals MT . The number of coefficients of value 1 in the center of the coefficient vector increases with MT . In the HP 3562A, there are 18 leading and 19 ending coefficients. Because of the separate quadrature formula for the last (partial) interval, the coefficients are not symmetric (e.g., the ending coefficients are not a reversed version of the leading coefficients).

Signal components not harmonically related to the measurement frequency ($f = 1/T$) will not, in general, be attenuated by as much as 70 dB. Fig. 5 represents the relationship between attenuation of a frequency component of a signal and frequency. To attenuate such nonharmonically related signal components, the integration time MT can be increased by increasing M .

Errors in the estimation of c_1 can also be produced by a

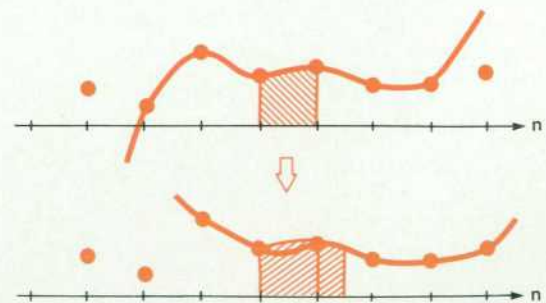


Fig. 3. Integrating over a portion of a sample period.

less than perfect sequence for $e^{-jn\omega T_s}$, say, $e^{-jn\omega T_s} + E_n$, where E_n is an error sequence. If $s(t)$ has a significant dc component s_0 , so that $s(t) = s_0 + s_1 \cos(\omega t + \phi)$, and E_n has a significant dc component E_0 , then the term $E_0 s_0$ will appear in the Fourier series integral and not cancel out. In the HP 3562A, the local oscillator signal $e^{-jn\omega T_s} + E_n$ has $E_n \leq 3 \times 10^{-4}$, hence this is not a significant problem.

Similarly, a stimulus signal with significant harmonic components (and subharmonic components in the case of a signal generated digitally and passed through a digital-to-analog converter) can cause harmonic and intermodulation distortion products from a nonlinear DUT to add spurious signal components at the measurement frequency, which will contribute directly to the error in estimating c_1 for a pure stimulus signal. This problem is addressed by the signal source design of the HP 3562A, which has no harmonic or subharmonic signal terms greater than 60 dB below the fundamental frequency component.

Sweep. The HP 3562A uses a stepped sine sweep rather than a continuous sweep. The instrument determines the frequency to be measured and sweeps the stimulus source frequency to this determined value. The HP 3562A then waits for transients to settle (usually 20% of the chosen integration time) and then begins the measurement at that frequency point.

The frequency resolution of a sweep (selected by the user) is the difference between adjacent frequencies in the measurement. Since the frequencies are at discrete points, important frequency response characteristics between these points might be missed. For this reason, the HP 3562A has an autorotation mode. In this mode, the instrument automatically takes finer frequency steps where necessary so that the user can specify a coarse resolution to speed up the measurement sweep without missing important information about the frequency response. The HP 3562A determines the frequency spacing based on the magnitude of the (complex) difference between the frequency response estimates of the last two measurements relative to the geometric average of the magnitudes of the two frequency responses. A large relative difference indicates a significant change in magnitude or phase of the frequency response over this frequency interval, and so the instrument decreases the frequency step to the next measurement point.

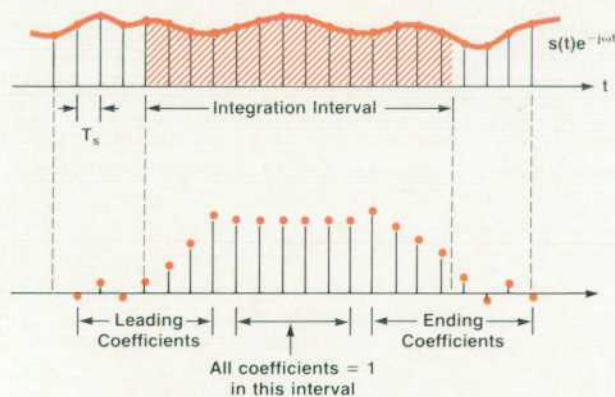


Fig. 4. Operation of modified composite quadrature integration formula using an FIR filter. (Top) Input signal. (Bottom) Coefficient vector.

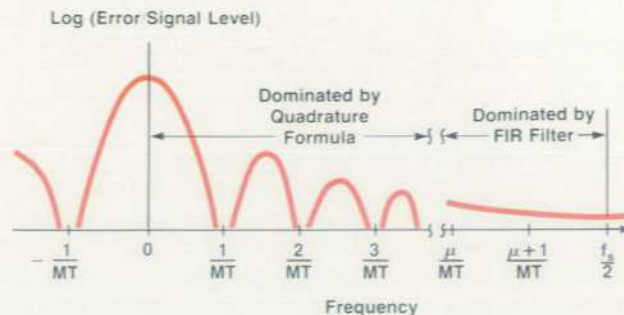


Fig. 5. Attenuation of signal components versus frequency.

After taking a measurement at a particular frequency point, if the instrument determines it has stepped too far, it backs up and takes the measurement again at a frequency closer to the previous measurement frequency.

Autointegrate. The nonharmonic noise rejection of the Fourier series integration algorithm depends on the integration time selected by the user. For measurements in which the signal-to-noise ratio is not constant over the frequency range, the user must select an integration time that takes the lowest signal-to-noise ratio into account. Because integration over a period this long is not required over the entire frequency span, time will be wasted if this integration time is used at each frequency point. The HP 3562A has an automatic integration feature that decides how long to integrate (up to the user-entered integration time) at each measurement frequency based on the effect the noise has on the measured transfer function magnitude at that frequency.

This is accomplished as follows. A small amount of data is taken and the transfer function estimate is computed. The same amount of data is taken again and the transfer function is again computed. The two data segments are contiguous; no data is missed. This is done three times, and the normalized variance of the magnitude of the transfer function is estimated. From this, the variance is estimated for the error on the transfer function magnitude that would result if it were calculated using a single Fourier series integral over all the data. If this estimate is larger than a user-entered level, another contiguous block of data is taken and the procedure is repeated. Otherwise, the result of the single Fourier series integral over all the data is used to calculate the transfer function estimate.

Autogain. For nonlinear devices, the transfer function will be a function of the level of the input to the device. It may be important to control the level of the signal at either the input or the output of the DUT. The HP 3562A's automatic gain feature allows this by adjusting the signal source level such that the desired reference signal level is achieved. The HP 3562A first estimates what the correct source level would be based on the result at the previous measurement frequency. The source is set at this level and a measurement is taken. If the reference signal is estimated to be within 1 dB of the desired level, the HP 3562A goes on to the next measurement frequency. If not, the HP 3562A adjusts the source level, waits for transients to die out, and repeats the measurement. The possibility of an infinite loop is eliminated by the algorithm, which quits trying to adjust

the source level and goes on to the next frequency point if the algorithm tries to move the source signal back toward a previous level.

Log Resolution Mode

The logarithmic resolution mode of the HP 3562A makes frequency-domain measurements at logarithmically spaced frequency points (lines) instead of the linearly spaced points measured by a standard linear resolution FFT measurement. This results in more measurements at the lower frequencies than at the higher frequencies (as shown in Fig. 6a), which is exactly what we want when trying to understand the response of a system over a large span. When the measured points are plotted on a log frequency axis, they are spaced linearly over the entire measurement span (see Fig. 6b). Hence, a single measurement provides an excellent understanding of the response of a system over a wide frequency range (up to five decades in the case of the HP 3562A). This Bode-plot type of measurement is especially useful when measuring electronic filters and dynamic control systems.

Measurement Points. Another way to say that the log resolution lines are logarithmically spaced is to say that they are proportionally spaced. That is, the ratio between the locations of any two adjacent lines is a constant. If we call this constant k , we can relate the locations of adjacent lines by:

$$f_c(m+1) = kf_c(m)$$

Applying this equation recursively yields:

$$f_c(m+n) = k^n f_c(m)$$

Since the log resolution mode of the HP 3562A provides a fixed resolution of 80 points per decade, we know that $f_c(m+80) = k^{80} f_c(m)$ and that $f_c(m+80) = 10f_c(m)$. Taken together, these equations state:

$$f_c(m+n) = 10^{n/80} f_c(m)$$

If we define the center frequency of the first measured point to be f_{st} (start frequency) and let $m = 0$ at this frequency, then we see that the n th line is defined by:

$$f_c(n) = 10^{n/80} f_{st} \quad (1)$$

If we now define the upper band edge of the n th line to

be halfway between $f_c(n)$ and $f_c(n+1)$ and the lower band edge of the n th line to be halfway between $f_c(n)$ and $f_c(n-1)$ when plotted on a log frequency axis, we can define the bandwidth of line n to be:

$$f_{bw}(n) = f_{upper}(n) - f_{lower}(n)$$

Since the halfway point on a log axis is the geometric mean on a linear axis, we have:

$$f_{bw}(n) = \sqrt{f_c(n) \times f_c(n+1)} - \sqrt{f_c(n-1) \times f_c(n)}$$

Substituting the center frequency definition derived above (Equation 1) yields:

$$f_{bw}(n) = f_c(n) \times (10^{1/160} - 10^{-1/160}) = 0.0288 f_c(n)$$

which shows that bandwidth is a percentage of the center frequency.

Single-Decade Measurements. The difference between a log swept sine measurement and a log resolution measurement is that while the former is computed a point at a time, the latter uses the FFT to compute a decade at a time.

The first step in making a single-decade log resolution measurement is to produce a linear spectrum for each channel. To do this, a 2048-point time block is taken with the low-pass filters set to a span equal to the stop frequency of the desired measurement. A Hanning window is then applied to the time domain data and two FFTs are performed, yielding the linear spectra: S_x and S_y .

Now three results are computed: Channel 1 power spectrum G_{xx} , Channel 2 power spectrum G_{yy} , and the cross-power spectrum G_{yx} . Averaging the cross-power spectrum rather than computing the frequency response directly is called trispectral averaging and is described in appendix A of reference 2.

For each log resolution line within the decade, the linear resolution measurements between $f_{lower}(n)$ and $f_{upper}(n)$ are added together. The result of these summations is the log resolution measurement for this line. Fig. 7 shows the shape of a representative log resolution line and it can be seen that there is no perceptible ripple in the bandpass area.

The summing process is repeated for each of the 80 log resolution lines. Since the linear resolution points being summed have a constant spacing, the number of points used to form each line will increase as the line center frequency increases. Since the roll-off of the line shape is related to the Hanning window that was applied to the

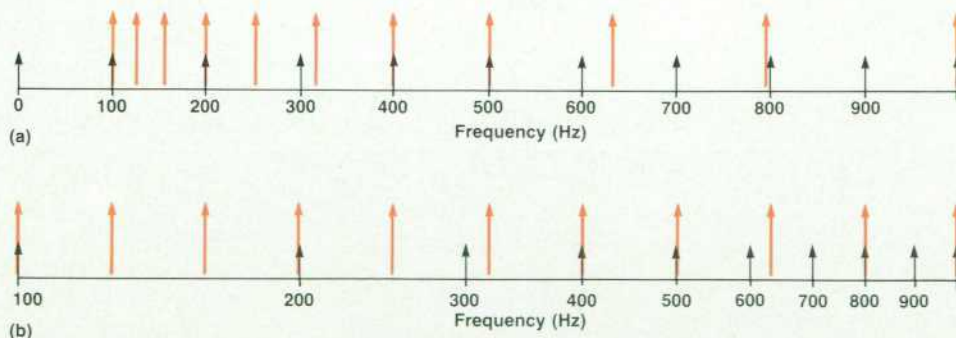


Fig. 6. Log and linear resolution modes. Black arrows indicate linear resolution measurement points, colored arrows indicate log resolution measurement points. (Top) Linear axis. (Bottom) Logarithmic axis.

linear resolution points, the roll-off becomes steeper (with respect to log f) for lines at higher frequencies.

Multidecade Measurements. When a span of more than one decade is selected, the above procedure is applied once per decade. In the multidecade case there is one important difference. Each channel of the HP 3562A contains two digital filters (required for filtering the complex time data produced by zoomed linear resolution measurements), and the digital filter controller hardware is intelligent enough to allow each filter to be set independently. Therefore, measurements on two decades can be made in parallel. The collection of data in log resolution mode is always done on a baseband span (i.e., it has a 0-Hz start frequency). In the multidecade case, the data for the first decade (lowest frequency) is processed by one filter and the data for the remaining decades is processed sequentially by the other filter. In addition to the parallel data collection and processing, overlap processing can also be done on the first decade (which has the longest time record) to speed up the measurement even more.

Multidecade log resolution measurements should be used only for measuring stationary signals, because the data records for the various decades are not collected simultaneously. This is also the reason for disallowing triggered measurements in log resolution mode.

Power Spectrum Accuracy. It was pointed out above that each log resolution line is formed by combining an integer number of linear resolution points. It should be apparent that this causes the actual bandwidth of each line to vary from the ideal bandwidth for that line. Frequency response measurements are not affected by this deviation since the amount of deviation is the same in each channel. However, it is noticeable when a power spectrum measurement of a broadband signal is made. The amplitude of the resulting spectrum (compared to the ideal result) would be too high at lines where the bandwidth is greater than ideal (because the line measures power over a wider band than ideal) and too low at lines where the bandwidth is less than ideal.

To resolve this problem, the power spectrum is "corrected." This is done by first dividing the measured power of each line by the actual bandwidth of that line (thus producing the power spectral density of the line) and then multiplying by the ideal bandwidth of that line. Although this technique works well when measuring broadband signals, it does mean that log resolution measurements will

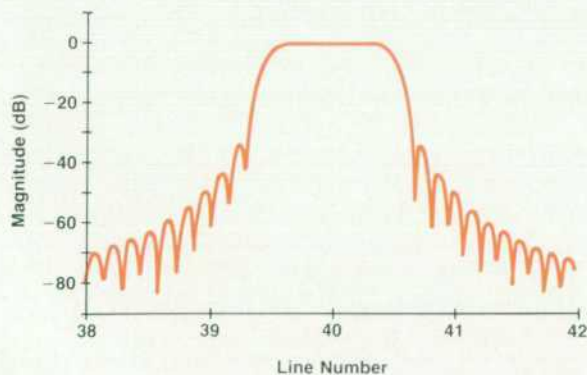


Fig. 7. Representative log resolution line shape.

not yield accurate results when measuring a narrowband (e.g., fixed sine) signal. The error in this case can be as much as +1.7 dB, -2.3 dB above the basic measurement accuracy and Hanning window uncertainty.

All of this implies that when using log resolution mode, the user should keep its purpose in mind: to provide accurate measurements on stationary broadband signals over a wide span.

Linear Resolution Mode

In its FFT-based linear resolution mode, the HP 3562A can perform a broad range of frequency, time, and amplitude domain measurements:

- Frequency: linear spectrum, power spectrum, cross-spectrum, frequency response, and coherence measurements
- Time: autocorrelation, crosscorrelation, and impulse response measurements
- Amplitude: histogram, probability density function (PDF), and cumulative distribution function (CDF).

The frequency domain measurements are made with 800 spectral lines of resolution over the selected frequency span in either single-channel or dual-channel operation. The custom digital filtering hardware allows the selected frequency span to be as narrow as 10.24 mHz for baseband measurements, or 20.48 mHz for zoom measurements. The center frequency for zoom measurements can be set with a resolution of 64 μ Hz anywhere within the instrument's overall frequency range.

Averaging can best be described in terms of the quantity being averaged and the type of averaging being performed. If time averaging is selected, then the quantity being averaged is the linear spectrum of the input time record. The averaging is done in the frequency domain so that the calibration and trigger correction can be done as a simple block-multiply operation. However, this is essentially the same as averaging the time records. Time averaging is useful for improving the signal-to-noise ratio when an appropriate trigger signal is available. A time-averaged frequency response function is computed by simply dividing the averaged output linear spectrum by the averaged input linear spectrum.

If time averaging is not selected, then the quantity being averaged is the selected measurement function itself (e.g., power spectrum, autocorrelation, etc). If a frequency response measurement is selected, then a trispectrum average is performed, where the averaged quantities are the input power spectrum G_{xx} , the output power spectrum G_{yy} , and the cross spectrum G_{yx} . Power averaging is useful for reducing the variance of a spectral estimate.

The type of averaging depends on the algorithm applied to the average quantity. Stable averaging is simply the sum of the averaged quantities divided by the number of averages (all data is weighted equally). Exponential averaging results in a moving average in which new data is weighted more than previous data (useful for dynamically changing signals). For frequency domain measurements, an additional type of averaging, peak hold, is provided. Peak hold results in a composite spectrum made up of the maximum value that has occurred in each of the 800 spectral lines. This type is useful for detecting noise peaks, signal drift,

Demodulation Example

Fig. 1 shows the instrumentation for measuring the distortion of an ordinary cone loudspeaker, in which both amplitude and phase distortion occur. The technique is to apply the sum of a low-frequency voltage and a midrange-frequency voltage to the speaker, and then to demodulate the components that appear around the midrange tone. Here, the midrange signal becomes the carrier, and the low-frequency tone generates the modulation sidebands.

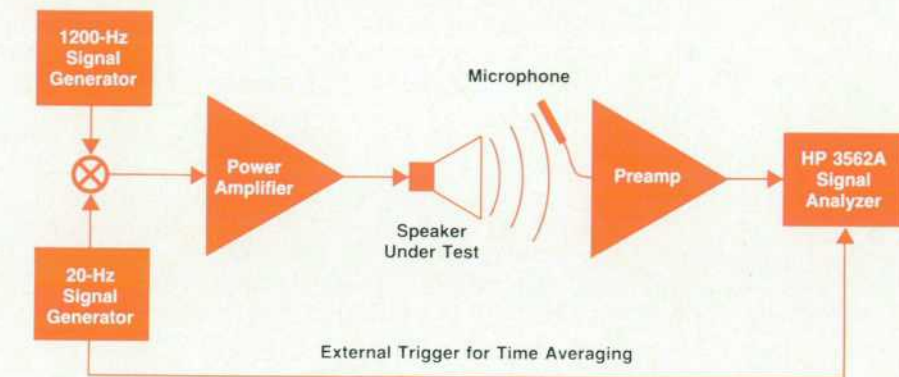


Fig. 1. The measurement of loudspeaker intermodulation distortion. When the sum of a low-frequency sinusoidal signal and a higher-frequency sinusoidal signal is applied to a speaker, various nonlinearities cause modulation of the higher-frequency signal by the low-frequency signal. Both phase and amplitude modulation will generally occur simultaneously.

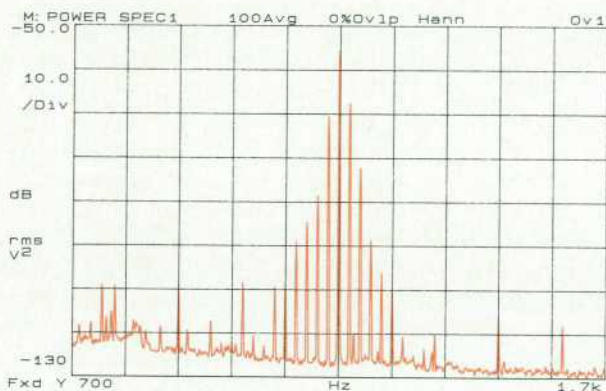


Fig. 2. Power spectrum of microphone signal showing 20-Hz modulation sidebands around the 1200-Hz carrier.

these two types of intermodulation distortion are caused by different mechanisms, their waveforms are still very much related in time.

Fig. 2 shows the power spectrum of the speaker output signal centered around the midrange frequency, as measured by the microphone. Fig. 3 shows the demodulated time waveforms for amplitude and phase using the midrange frequency as the carrier. Finally, Fig. 4 shows the HP 3562A's demod-polar plot, in which both types of distortion are shown simultaneously. The origin for the carrier vector is at the left of the plot, and only the tip locus is shown. The carrier vector amplitude varies coherently with the phase modulation, even though these variations tend to be caused by totally different types of nonlinearity.

Ronald W. Potter
Consultant

or maximum vibration at each frequency.

The HP 3562A's built-in source provides a variety of output signals to satisfy many different network measurement requirements. The random noise output is a broadband, truly random signal. Since it is nonperiodic, it has continuous energy content across the measurement span and an appropriate window (e.g., a Hanning window) must be applied to reduce leakage. Averaging the random noise input and response signals reduces the effects of nonlinearities in the network under test. This, in effect, linearizes the network's frequency response function. By using random noise as a stimulus, the response function calculated from a trispectrum average (G_{yx}/G_{xx}) is a linear least-squares estimate of the network frequency response.

The periodic chirp output is essentially a fast sine sweep wholly contained within the measurement time record. This output is also a broadband signal, but since it is

periodic, leakage-free measurements can be made without windowing. Thus, a single, nonaveraged measurement using the periodic chirp signal gives accurate results in many linear applications. The periodic chirp also has a higher rms-to-peak ratio than stimuli using random noise.

The HP 3562A also provides a burst random noise output that combines many of the benefits of the above source outputs. The random noise output is gated on for a user-selectable percentage of the measurement time record. If the network response signal decays to a negligible level within the time record, then a leakage-free measurement can be made without windowing. Since the signal is truncated random noise, it has continuous rather than discrete energy across the frequency span.

The periodic chirp output can also be operated in a burst mode. All of these broadband signals are band-limited to the measurement frequency span, even for zoomed opera-

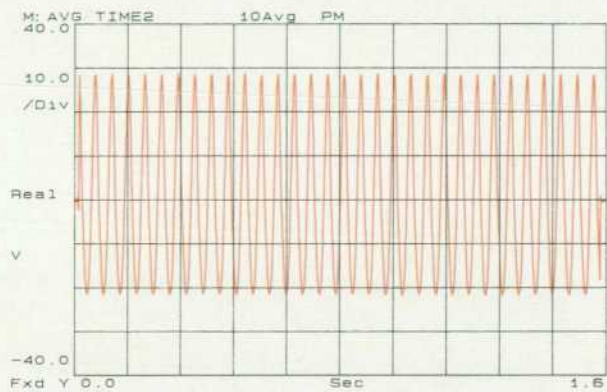
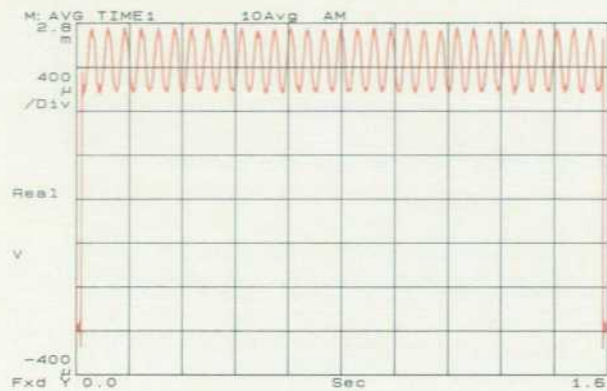


Fig. 3. (Top) Average of ten demodulated AM time waveforms. The modulation index is approximately 12%. The endpoints are set to zero to show carrier amplitude. (Bottom) Average of ten demodulated PM time waveforms showing approximately 50 degrees of peak-to-peak phase modulation. The endpoints are set to zero phase.



Fig. 4. Demod-polar plot. Approximately three cycles of the locus of the tip of the modulated carrier vector are shown after ten demodulated time record averages. The dashed line shows the locus of constant carrier amplitude. The marker on this line shows the zero phase point, corresponding to an unmodulated carrier.

tion. In addition to these broadband signals, a narrowband signal, fixed sine, is available for stimulating a network at a single frequency.

A special source protect feature is provided for those applications where an abrupt change in excitation power level can cause damage to the device under test. When this feature is invoked, it provides two types of protection. First, when the source level is changed, the output ramps to the new level instead of changing immediately. Second, if the source is on and the user changes any parameters that affect the source output (e.g., center frequency, span, or source type), then the source output ramps down to zero volts.

Within the HP 3562A, the demodulation process is performed digitally rather than by more conventional analog methods. This process (see following section) can be thought of as a measurement preprocessor in which the input is a modulated carrier and the output is a time record of only the modulating signals. The demodulated AM, FM, or PM time records can then be used as inputs to any of the measurements available in the linear resolution mode.

For applications where the carrier frequency may not be known, there is an autocarrier mode in which the demodulation process automatically determines the carrier fre-

quency required to demodulate the input signal correctly.

Digital Demodulation

In many applications, a signal spectrum will have a narrow bandwidth, and the information of interest is carried by the modulation of that signal. A carrier can either be amplitude or phase modulated, and both types of modulation can coexist. Frequency modulation is essentially the same as phase modulation, with the frequency-versus-time modulation waveform being the time derivative of the equivalent phase-versus-time waveform.

In general, the user would like to demodulate any such narrowband signals, and would like to separate amplitude modulation from phase or frequency modulation. In addition, the amplitude and/or phase of the carrier may also be of interest. In the HP 3562A, either the modulation time waveforms or their equivalent frequency spectra can be obtained, and the two types of modulation can be displayed together in a polar plot to show any relationships that might exist between the two.

Demodulation Equations. Any narrowband signal can be represented as a modulated carrier, and expressed in the following form:

$$x(t) = m(t)e^{j\omega_0 t} + m^*(t)e^{-j\omega_0 t}$$

where the carrier angular frequency is ω in radians/second and $m(t)$ is the complex modulating waveform given by:

$$m(t) = A[1 + a(t)]e^{j\phi(t)}$$

The carrier amplitude and phase are represented by the complex number A , while $a(t)$ and $\phi(t)$ represent the amplitude and phase modulating waveforms, respectively. The quantity $m^*(t)$ represents the complex conjugate of $m(t)$. Thus, $x(t)$ is a real modulated waveform formed by the sum of a complex waveform and its complex conjugate.

As long as the spectra for the two parts of $x(t)$ do not overlap in the frequency domain, it is possible to separate the amplitude and phase modulation components unambiguously. There are several ways to accomplish this, but one of the simplest ways is to construct a single-sided filter that only passes the positive frequency image. This is done in the HP 3562A by digitally shifting the positive image of the signal spectrum down to a frequency band near the origin, and then passing the resulting complex time waveform through a digital low-pass filter.

Assuming that the negative frequency image has been completely rejected by the low-pass filter, the output time waveform will contain only the positive spectral image, and can be written as:

$$\begin{aligned} y(t) &= m(t)e^{j\omega_0 t} \\ &= A[1 + a(t)]e^{j[\omega_0 t + \phi(t)]} \\ &= u(t) + jv(t) \end{aligned}$$

Here, $u(t)$ and $v(t)$ are real time waveforms, and are the real and imaginary parts of $y(t)$, respectively.

The magnitude of $y(t)$ gives the amplitude modulation waveform, while the phase of $y(t)$ gives the phase modulation, including the carrier ramp $\omega_0 t$. Hence:

$$\sqrt{u(t)^2 + v(t)^2} = A[1 + a(t)] \quad (2)$$

$$\tan^{-1}[v(t)/u(t)] = \angle A + \omega_0 t + \phi(t) \quad (3)$$

If Equation 3 is differentiated with respect to time, the carrier phase disappears, giving:

$$v(t) = \omega_0 + d\phi/dt \quad (4)$$

The carrier frequency ω_0 is estimated by calculating the average value of Equation 4 weighted by a Hanning window to reduce errors caused by leakage effects. When this carrier term is removed, the remainder is the frequency modulating waveform. This quantity is then integrated with respect to time, obtaining the phase modulating waveform $\phi(t)$ with all carrier components removed. A similar technique of weighted averaging is used in Equation 2 to define the carrier amplitude A so that the amplitude modulating waveform $a(t)$ can be obtained.

Digital Demodulation in Practice. Although the basic demodulation equations described above are straightforward,

there are a number of practical details that must be considered in the implementation of these equations.

The data flow block diagram for this process is shown in Fig. 8. The demodulation is introduced immediately after the digital antialiasing filter, and just before the various measurement options are selected. Thus, most of the normal types of measurements that can be made on input time records can also be made on demodulated time records. This includes ensemble averaging, transfer and coherence function calculations, correlation functions, histograms, and power spectra. The type of demodulation can be selected separately for each of the HP 3562A's two input channels, so it is possible to demodulate amplitude and phase (or frequency) simultaneously.

If any parts of the original positive and negative frequency spectral images overlap, then it is not possible to separate the two modulating waveforms unambiguously. If demodulation is attempted in this case, the resulting waveforms will not be entirely correct. This overlap can occur about the frequency origin if the original modulation bandwidth is excessive, or it can occur around half the sampling frequency because of aliasing.

Any spectral components in the band near the carrier are assumed to be modulation sidebands around that carrier. If this is not the case, errors will be introduced into the demodulated waveforms. For example, if there is an additive spectral component at the ac line frequency, this component must be removed before demodulation is attempted, or there will be line frequency components in both the amplitude and phase waveforms. If dc or any carrier harmonics are present in the original spectrum, these must also be removed, or else they will appear as

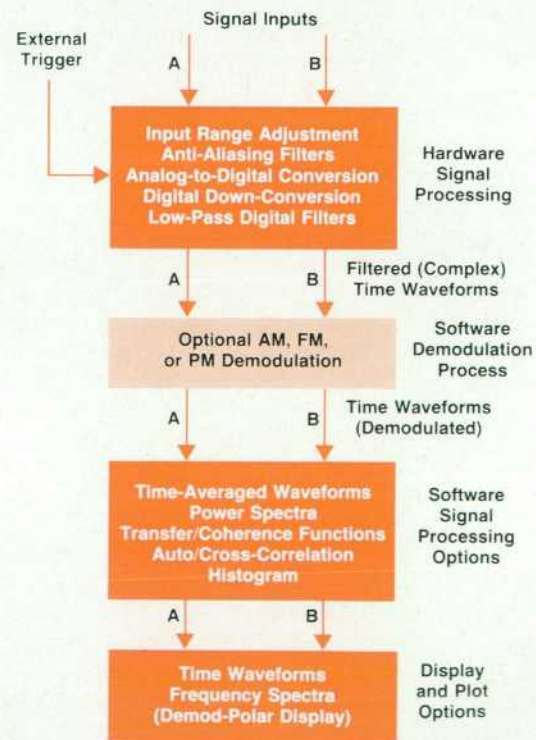


Fig. 8. Data process flow for digital demodulation.

modulation components at the carrier frequency.

The HP 3562A has provisions for removing selected regions of the spectrum before the demodulation process. There is a preview mode, in which the original spectrum can be displayed before the demodulation step. A number of frequency regions can be selected, within which the spectrum is replaced by a straight line segment connecting the end points of each region. In this manner, narrow bands of contaminating interference can be effectively removed from the data before demodulation is initiated.

If the original signal passes through any filter whose passband is not constant with frequency, there will be some conversion of one type of modulation to the other. Thus, it is very important that the filter transfer characteristics be completely removed from the signal before demodulation is attempted. This is particularly important for any phase slope (or group delay variation) introduced by the filter, since most antialiasing filters have large group delay changes, especially near the cutoff frequency, even though the passband amplitude characteristic may be very flat. There is an automatic system calibration cycle in the HP 3562A, during which the filter transfer function is measured. It is then automatically removed from each record

of the data before demodulation is initiated. This keeps intermodulation conversion errors below a nominal level of about -50 dB.

Since it is possible to measure both amplitude and phase modulation waveforms simultaneously, there is a special display mode in which these two quantities can be shown together. A modulation process can be envisioned as a variation in the amplitude and/or phase of a carrier vector (or phasor). Thus, the locus of the tip of this vector can be used to describe the modulation. The HP 3562A's demod-polar display mode shows this locus using rectangular coordinates. This allows any mutual relationships that might exist between amplitude and phase modulation waveforms to be shown. This can be very useful, for example, when the causes of various types of intermodulation distortion are under investigation, as illustrated by the example given in the box on page 22.

References

1. L.W. Johnson and R.D. Riess, *Numerical Analysis*, Second Edition, Addison-Wesley, 1982.
2. *The Fundamentals of Signal Analysis*, Hewlett-Packard Application Note 243.

Analyzer Synthesizes Frequency Response of Linear Systems

by James L. Adcock

A COMPLETE CAPABILITY for synthesizing the frequency response of linear systems based on their pole-zero, pole-residue, or polynomial model is included in the HP 3562A Signal Analyzer. This synthesis capability includes table conversion, the ability to convert automatically between the three models. The frequency responses can be frequency scaled and system time delays can be added. The designed system frequency responses are synthesized with exactly the same frequency points as those used by the corresponding HP 3562A measurement mode. Hence, the synthesized version of the desired frequency response can be directly compared to the measured response of the actual system.

Pole-zero formulation corresponds to the series form of filter design most often used by electrical engineers. Pole-residue form is commonly used by mechanical engineers and in modal analysis, and corresponds to parallel design filters in electrical engineering. All three forms, pole-zero, pole-residue, and polynomial, find direct application in analysis and design of servo systems. Having all three synthesis forms available in the HP 3562A, users can select the formulation best-suited for their application. The HP 3562A also facilitates the solution of mixed-domain problems, such as electromechanical servo design, which re-

quires a variety of electrical and mechanical design techniques.

Synthesis Table Conversion

Let us try table conversions on the following simple example with two poles at $-1 \pm j10$ and a zero at -2 . The response of this filter is shown in Fig. 1a. The pole-zero equation is:

$$\frac{s+2}{(s+1-j10)(s+1+j10)}$$

which is represented by the HP 3562A's display as shown in Fig. 1b. The pole-zero diagram is shown in Fig. 1c.

If the HP 3562A converts the equation to polynomial form, the numerator and denominator terms are simply multiplied out:

$$\frac{s+2}{s^2+2s+101}$$

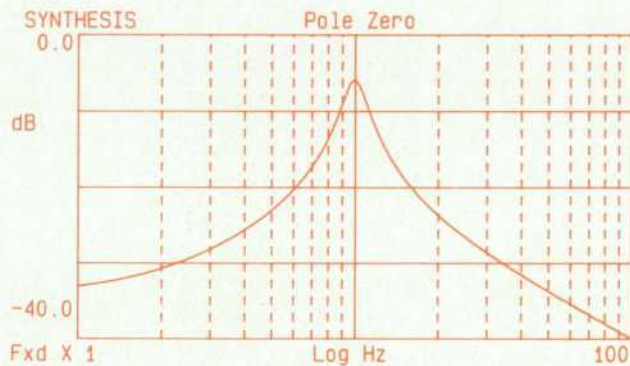
Alternatively, the pole-zero or polynomial representation can be converted to pole-residue form:

$$\frac{a}{s+1-j10} + \frac{a^*}{s+1+j10}$$

where the HP 3562A solves for $a = 0.5 - j0.05$ as expected. Trying a slightly more difficult example with repeated poles:

$$\frac{s+2}{(s+1+j10)^2(s+1-j10)^2}$$

Converting to polynomial form leads to the expected result (Fig. 2a):



(a)

Synthesis				
Poles And Zeros				
	POLES		ZEROS	
1	-1.0 ±j	10.0	-2.0	
Time delay= 0.0 S Gain= 1.0 Scale= 1.0				

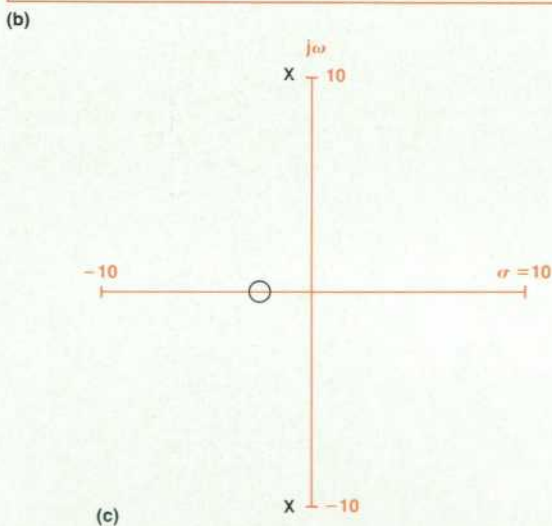


Fig. 1. (a) Response of a filter with two poles at $-1 \pm j10$ and a zero at -2 . (b) Poles and zeros displayed by HP 3562A for (a). (c) Pole-zero diagram for (a).

$$\frac{s+2}{s^4+4s^3+206s^2+404s+10201} \quad (1)$$

However, how is this function represented in pole-residue form? Again, the HP 3562A has the answer (Fig. 2b):

$$\frac{-j250 \times 10^{-6}}{s+1-j10} + \frac{j250 \times 10^{-6}}{s+1+j10} + \frac{-2.5 \times 10^{-3} - j25 \times 10^{-3}}{(s+1-j10)^2} + \frac{-2.5 \times 10^{-3} + j25 \times 10^{-3}}{(s+1+j10)^2} \quad (2)$$

A more interesting pole-residue case occurs when there are more zeros than poles:

$$\frac{(s+1)(s+2)(s+1-j5)(s+1+j5)}{(s+1-j10)(s+1+j10)}$$

This results in extra Laurent expansion terms—isolated powers of s that did not appear in the original pole-zero form (see Fig. 2c):

$$\frac{-37.5 - j375}{s+1-j10} + \frac{-37.5 + j375}{s+1+j10} - 73 + 3s + s^2 \quad (3)$$

(a)

Synthesis		
Polynomials		
	NUMERATOR	DENOMINATOR
1	2.0	10.201k
2	1.0	404.0
3		206.0
4		4.0
5		1.0
Time delay= 0.0 S Gain= 1.0 Scale= 1.0		

(b)

Synthesis				
Poles And Residues				
	POLES		RESIDUES	
1	-1.0 ±j	10.0	0.0 ±j	-250.0μ
2	(-1.0 ±j)	(10.0) ²	-2.5m±j	-25.0m

(c)

Synthesis				
Poles And Residues				
	POLES		RESIDUES	
1	-1.0 ±j	10.0	-37.5 ±j	-375.0
2	s^0		-73.0	
3	s^1		3.0	
4	s^2		1.0	

Fig. 2. HP 3562A display of (a) polynomial table for equation 1, (b) pole-residue table for Equation 2, and (c) table for equation 3 with Laurent expansion terms.

The implementation of the synthesis table conversions in the HP 3562A was straightforward, except for keeping track of a multitude of details. The pole-residue form requires many different cases. A consistent representation of the many forms had to be designed within the constraints of the HP 3562A's displayable character set. The necessary zero-finder routine is shared with the curve-fitting algorithm (see article on page 33).

Many table conversions result in small residual error terms after conversion. To keep from displaying the error terms, the table conversion routines estimate the errors in their arithmetic as they proceed. If a term is as small as the expected error for that table conversion, then the table conversion routines assume that the term is indeed caused by arithmetic errors and sets the term to zero. This allows most table conversions to work exactly as expected. For example, converting a polynomial table to a pole-zero table will give a zero at 0 Hz, if that is where it belongs, not at 2.31×10^{-23} . Also, converting a pole-zero table with Hermitian symmetry to rational polynomial form will result in purely real polynomials.

Another problem is caused by the large numbers often encountered in polynomials. For example, $(s-10,000)^{10}$ converted to polynomial form results in numbers as large as 10^{40} . This will lead to numerical problems. We solve this problem in several ways. First, a scale frequency parameter is introduced to allow designers to design in units that are appropriate for their particular design problem. For many designs, choosing one hertz or one radian as the design unit is no more applicable than designing electrical filters using ohms, farads, and henries (most practical de-

signs use k Ω , μ F, and mH). Likewise, many filter designs are more naturally expressed in kHz or kiloradians, or normalized to the corner frequency or the center frequency of the filter bandwidth. Second, the formulas used for table conversions and frequency response synthesis were carefully designed to try to minimize numerical problems. Finally, if numerical problems cannot be avoided, the HP 3562A table conversion routines attempt to diagnose the error, and warn the user of the numerical problems.

Designing a Chebyshev Low-Pass Filter

For a simple example of using the synthesis and curve-fitting capabilities of the HP 3562A, let's construct an equiripple low-pass filter. The magnitude of the response of this type of filter is equal to $1/|T_k(\omega) + j\epsilon|$, where $T_k(\omega)$ is the kth-order Chebyshev polynomial. Using the Chebyshev recursion relationship, we can quickly arrive at, for example, a fifth-order Chebyshev polynomial:

$$T_5(\omega) = 16\omega^5 - 20\omega^3 + 5\omega$$

This polynomial oscillates between ± 1 over the frequency range of $\omega = \pm 1$. We choose $\epsilon = 1.0$, resulting in a passband ripple of 3 dB. This function can be synthesized using the HP 3562A's polynomial synthesis capability (Fig. 3).

Since the HP 3562A polynomials are expressed in terms of $j\omega$, the filter function synthesized is actually:

Synthesis Polynomials		
NUMERATOR	0	DENOMINATOR 5
1	1.0	1.0
2		5.0
3		0.0
4		20.0
5		0.0
6		16.0

Time delay= 0.0 S Gain= 1.0 Scale= 10.0K

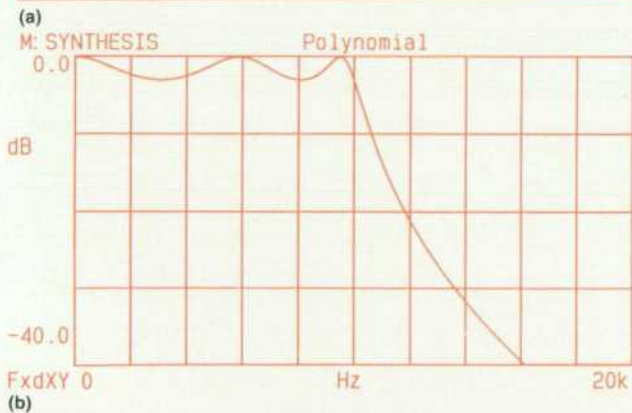


Fig. 3. Table (a) and synthesized response (b) for fifth-order Chebyshev polynomial (equation 4).

Curve Fit Poles And Zeros			
	POLES	10	ZEROS 0
1	-1.77184k		
2	1.77184k		
3	1.43346k+j	5.96936k	
4	-1.43346k+j	5.96936k	
5	-547.536 +j	9.6586k	
6	547.536 +j	9.6586k	

Time delay= 0.0 S Gain=-39.E+36 Scale= 1.0



Fig. 4. (a) Table of ten poles obtained by curve fitting equation 5. (b) Response obtained by discarding right-hand poles listed in (a) and resynthesizing the resulting pole-zero function.

$$H(\omega) = [16(j\omega)^5 + 20(j\omega)^3 + 5(j\omega) + 1]^{-1}$$

$$= [16j\omega^5 - 20j\omega^3 + 5j\omega + 1]^{-1}$$

Synthesis			
Poles And Zeros			
	POLES		ZEROS
	5		2
1	-14.0k		0.0 ±j 32.4k
2	-8.75k±j	16.24k	
3	-2.44k±j	22.0k	
Time delay= 0.0 S Gain=2.2E+12 Scale= 1.0			

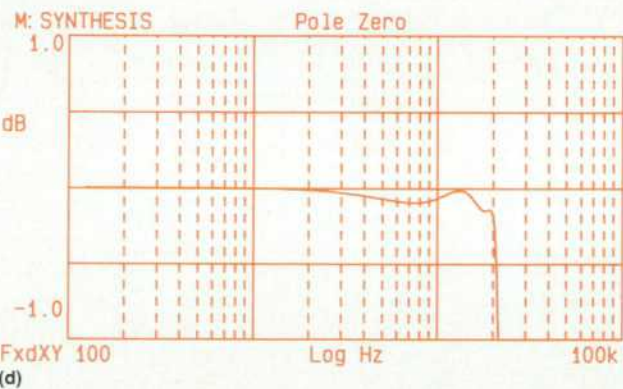
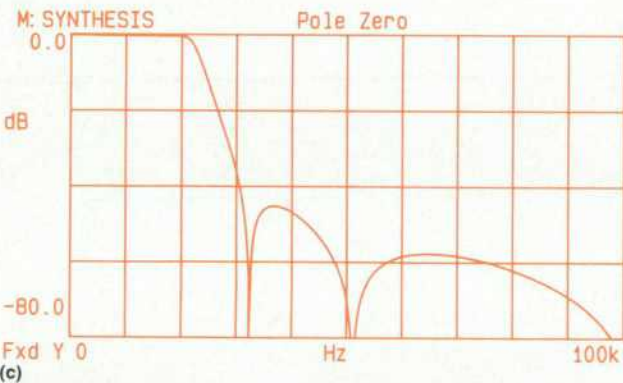
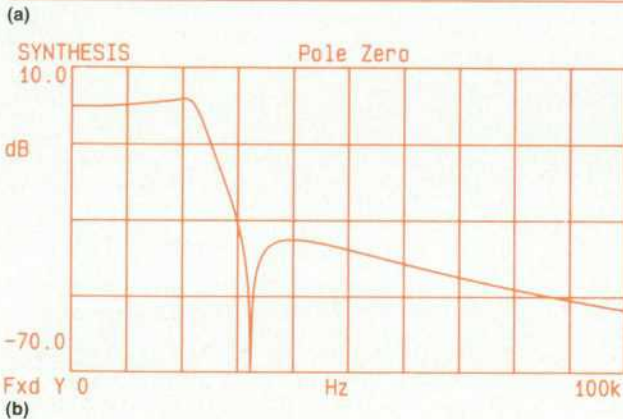
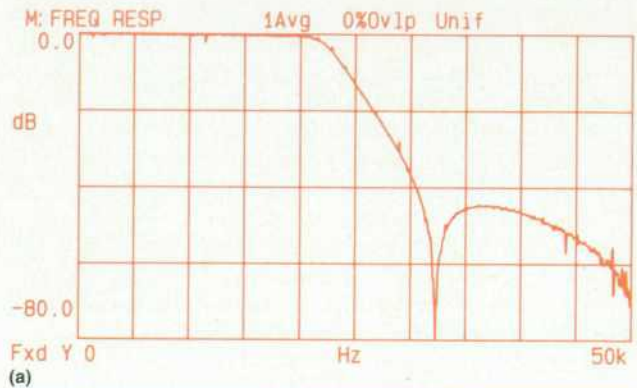


Fig. 5. (a) Table of poles and zeros for synthesis of reconstruction filter. (b) Synthesized response of filter by itself. (c) Response of combined system (DAC plus filter). (d) Detailed passband performance.



Curve Fit			
Poles And Zeros			
	POLES		ZEROS
	5		10
1	-13.7761k		-71.9401 ±j 32.4116k
2	-8.69592k±j	16.3601k	0.0 ±j 51.2k+
3	-2.51997k±j	22.0867k	0.0 ±j 102.4k+
4			0.0 ±j 153.6k+
5			0.0 ±j 204.8k+
Time delay= 0.0 S Gain= 1.0 Scale= 1.0			

Fig. 6. Measured response (a) and table of poles and zeros (b) for actual filter constructed using the parameters synthesized in Fig. 5.

In addition, a frequency scaling factor has been entered to specify a corner frequency at $(\omega = 1) \equiv 10$ kHz.

While the synthesized function has the correct magnitude for the filter we want, the phase of the synthesized function may not correspond to the phase of the filter we are trying to synthesize. Let's ignore phase for a moment by taking the squared magnitude:

$$|H(\omega)|^2 = H(\omega)H^*(\omega)$$

Curve fitting this function gives us ten poles, half of which are in the right-hand plane (Fig. 4a). Discarding the right-

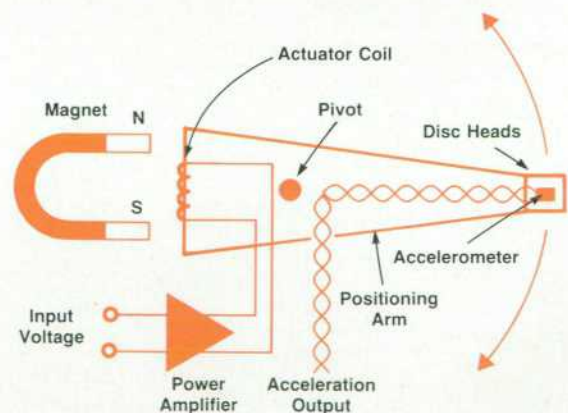


Fig. 7. Simple head positioning system for a disc drive.

hand plane poles, and resynthesizing the resulting pole-zero function, gives us the filter response desired (Fig. 4b). An alternate approach would be to curve fit the original synthesized function $H(\omega)$ with results very similar to those used for Fig. 4b. The right-hand pole is then reflected into the left-hand plane to arrive at a stable function with identical magnitude response.

Designing a Reconstruction Low-Pass Filter

As a more complicated filter design example, we wish to design a simple low-pass filter that will aid in the reconstruction of an analog signal from digital data samples using a digital-to-analog converter (DAC). The filter has three requirements:

1. The filter must block alias components at 1.56 times the passband corner frequency (the sample rate of the dig-

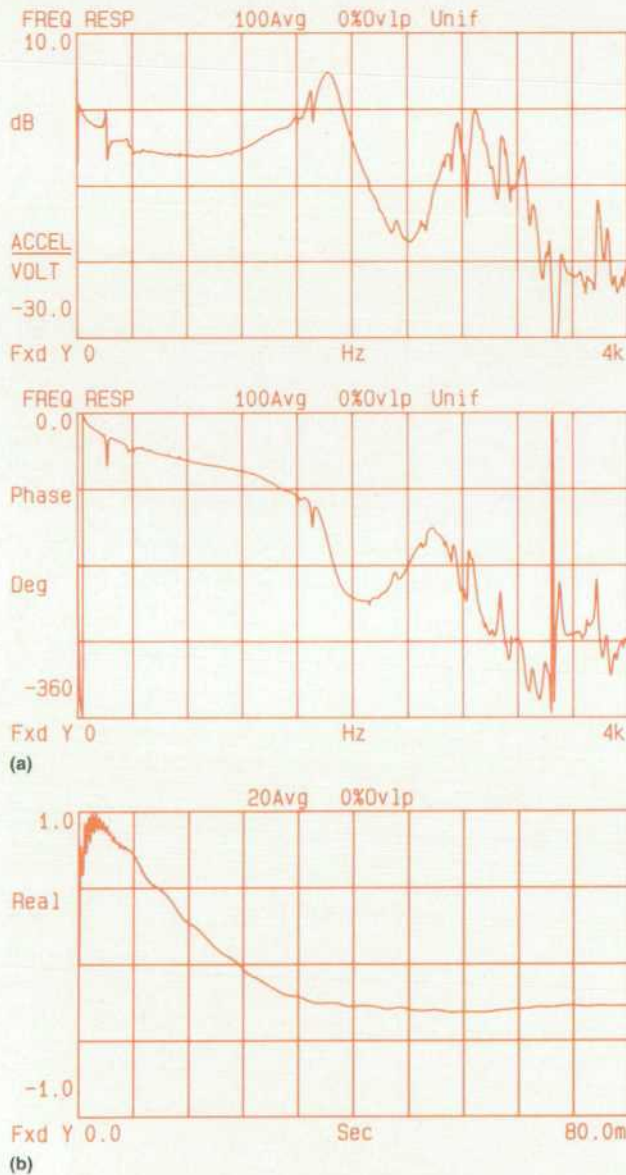


Fig. 8. (a) Plot of the acceleration response of the system in Fig. 7 as a function of the excitation voltage frequency. (b) System step response.

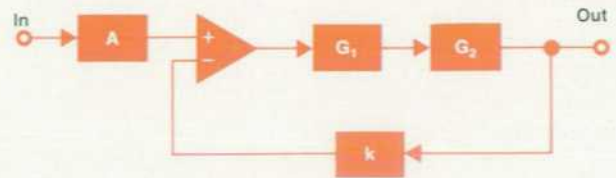


Fig. 9. Typical feedback control system.

ital system is 2.56 times the filter corner frequency). For this particular application, 45 dB of alias protection is considered adequate.

2. The filter must compensate for a $(\sin \omega)/\omega$ roll-off caused by the DAC outputting rectangular pulses rather than theoretically infinitely narrow impulses. After compensation, the DAC and reconstruction filter together should have a passband flatness better than ± 0.5 dB.

3. The filter must be inexpensive, using a minimum of second-order stages.

To implement this design, the characteristic $(\sin \omega)/\omega$ roll-off is first examined on a sample frequency of 51.2 kHz. This is approximated by the real part of:

$$[-16,300 \exp(-j9.766 \times 10^{-6} \omega)] / (\omega - 10^{-6})$$

which the HP 3562A can synthesize as a pole at 10^{-6} Hz with a gain factor of $-16,300$ and a time delay of $9.766 \mu\text{s}$.

A small offset is added to ω in the denominator to avoid dividing by zero. Curve fitting the $(\sin \omega)/\omega$ roll-off over a 0-to-20-kHz range indicates that the roll-off can be well represented as a single heavily damped pole within this

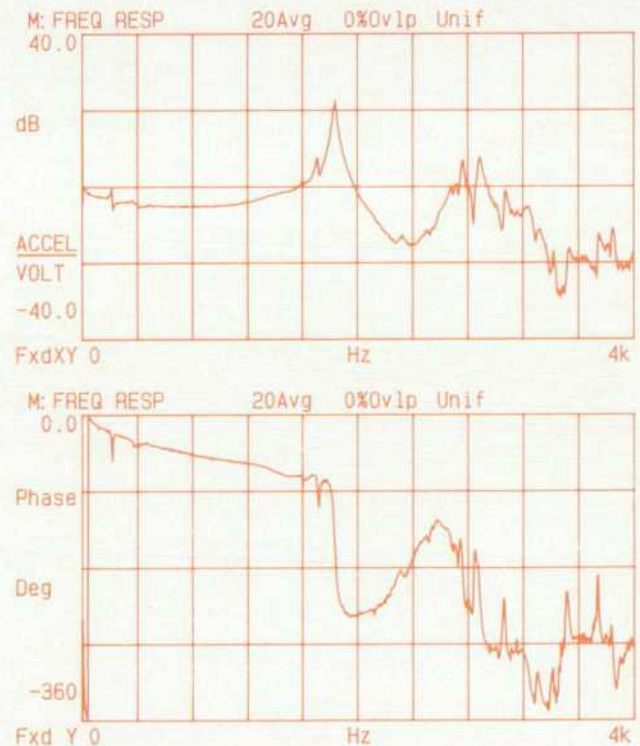


Fig. 10. Improvement in system response (Fig. 8a) as k is increased slightly until the system is nearly unstable.

frequency range. This convinces us that the roll-off correction can be easily handled by slightly modifying the pole locations of a standard low-pass filter design.

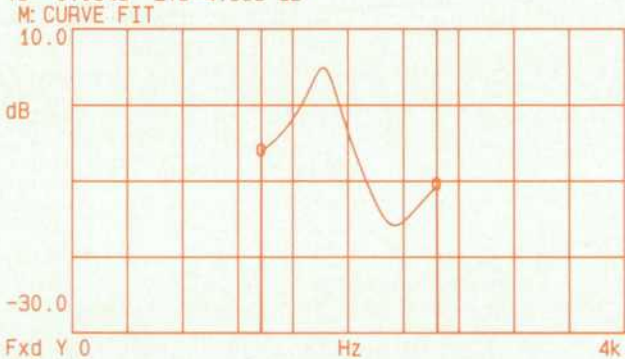
Accordingly, the fifth-order Chebyshev polynomial discussed in the earlier example is tried, leading to the conclusion that a zero is needed near 1.6 times the corner frequency of the filter to make the transition to 45 dB of attenuation quickly enough. Trial solutions for the pole locations are determined by using the above Chebyshev design technique with a passband ripple of ± 0.25 dB. The zero location chosen is based on the 1.56-times-the-corner-frequency criterion (requirement 1 on page 29). Then, including the $(\sin \omega)/\omega$ roll-off factor, the pole and zero locations are modified manually by trial and error until the desired performance is reached. The fast synthesis capabilities of the HP 3562A make this manual adjustment ap-

X=1.37kHz $\Delta X=1.275$ kHz
Ya=-4.9411 $\Delta Ya=4.076$ dB



(a)

Yb=-6.0948 $\Delta Yb=4.555$ dB



(b)

Curve Fit				
Poles And Zeros				
	POLES		ZEROS	
1	-93.2738 +j	1.83088k	2.20902k+j	1.10028k
2	-716.078 +j	3.13107k	-168.942 +j	2.31066k
Time delay= 0.0 S Gain=-6.725m Scale= 1.0				

(c)

Fig. 11. By curve fitting the resonance portion of the response in (a) as shown in (b), the HP 3562A obtains the poles and zeros listed in (c).

proach practical. The table in Fig. 5a lists the pole-zero locations of the designed reconstruction filter. Fig. 5b shows the performance of the filter by itself and Fig. 5c shows the combined system performance. Fig. 5d shows the detailed passband performance of the combined filter and $(\sin \omega)/\omega$ system. While not quite optimal, the design is flatter than can be achieved with the 1% resistor and capacitors to be used in the circuit implementation.

Once the circuit is constructed, the actual performance can be measured using the HP 3562A's measurement facilities (see Fig. 6a). Then the curve fitter is asked to find the pole-zero locations actually obtained. Since the zeros of the $(\sin \omega)/\omega$ part of the system are known exactly, based on the system's sample frequency, the first four of these values are entered explicitly in the curve fitter table (Fig. 6b). The curve fitter then considers the entered values to be known constraints on the curve-fit pole-zero locations to be found and it only solves for the unknown pole and zero locations.

Because of component tolerances, stray capacitance, finite op amp bandwidths, and other imperfections, the prototype circuit performance is seldom exactly as designed. The pole-zero locations can be adjusted again, based on the performance achieved in the first-pass design, so that the production filter will be as desired.

A Servo Design Example

As a more advanced example of the use of the HP 3562A's synthesis and curve fit abilities, consider the task of designing a servo control system. Fig. 7 is a sketch of the prototype of a disc head positioning system. A voltage is applied to an electromagnet attached to the disc head positioning arm,



Fig. 12. Compensated open loop system response.

and the electromagnetic field generated opposes the static magnetic field of the permanent magnet, causing the positioning arm to move to an equilibrium position. That motion is detected by an accelerometer attached to the arm. We would like to use the information provided by the accelerometer to improve the response of the positioning arm to the excitation voltage. The accelerometer provides no static dc information on the position of the arm, so an additional position feedback system will ultimately be required for movement at very low frequencies.

Fig. 8a is a plot of the acceleration response of the original system as a function of the frequency of the applied voltage. The two initial primary problems with this response are the strong resonance at 1.8 kHz and a sharp roll-off in the response near dc. Fig. 8b shows the corresponding step response. This is clearly a poor response for a positioning system to have. We need to improve the response using a feedback control system.

Fig. 9 shows a diagram of a typical feedback control system, where G_2 is the existing electromechanical system whose performance we would like to improve. G_1 is a filter to control loop performance, k is a loop gain parameter, and A is a precompensation filter to improve the output response without changing the control loop performance.

As an initial try, close the loop without compensating networks. Set $A = 1$, $G_1 = 1$, and increase k from zero until G_2 begins to go unstable. Fig. 10 shows the resulting improvement in the system as k is increased until the system is almost unstable. However, even with very small values of k the system becomes unstable, the response is dominated by a very sharp resonance at 1.8 kHz, and the system

response is not significantly improved at frequencies below 1.8 kHz. This problem is typical of electromechanical control systems. The pole is caused by the first significant structural resonance of the positioning arm.

As we close the loop by increasing k , the system instability occurs at the frequency where the open-loop phase crosses 180 degrees. With the 180-degree phase shift, the negative feedback becomes positive feedback, and when k times the magnitude of G_2 is greater than one, we have an oscillator (or a broken system).

The fundamental problem to solve is to keep the system phase away from 180 degrees for as long as possible, and then to bring the system loop gain below unity before the phase does go to 180 degrees. A major portion of the phase problem is caused by the resonance at 1.8 kHz and the accompanying antiresonance at 2.2 kHz. We curve fit these two features to find their actual frequencies and dampings (Fig. 11). The poles at ± 1.83 kHz and the zeros at ± 2.31 kHz found by the curve fitter (Fig. 11c) are the actual poles and zeros we are looking for. The others are computational poles and zeros added by the curve fitter to compensate for resonances outside the frequency range the curve fitter examined.

As a first attempt at loop compensation, we place a zero ($-95 \pm j1830$) on top of the pole location just found, and place a pole ($-170 \pm j2300$) on top of the zero found. Fig. 12 shows the resulting compensated open-loop system performance. While smoother, the phase still rolls toward 180 degrees faster than we would like, and there are a number of difficult-to-control resonances around 2.4 kHz. Let's try rolling off the loop gain below 2.4 kHz, but keep the loop gain high at 200 Hz where there is a sharp structural resonance.

Fig. 13 shows the response of an additional G_1 element we design to meet these loop roll-off goals. This element is a pole pair at $-250 \pm j250$ Hz. Unfortunately, this pole pair creates added phase delay, greatly lowering the frequency at which the G_1 - G_2 system response crosses 180 degrees (Fig. 14). We ask the HP 3562A's curve fitter to find an all-pass phase compensation network to solve this phase problem. Setting the magnitude to unity and curve fitting to the phase response as shown in Fig. 14 gives two poles (-1721.15 and -300.517) and two zeros (287.609 and 1558.58). To be strictly all-pass, the pole locations



Fig. 13. Response of additional G_1 element designed to meet loop roll-off goals.

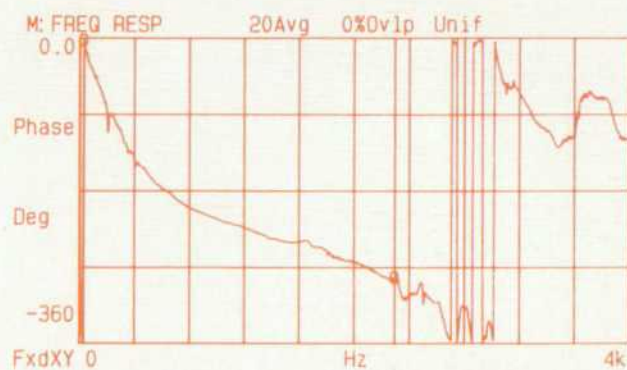


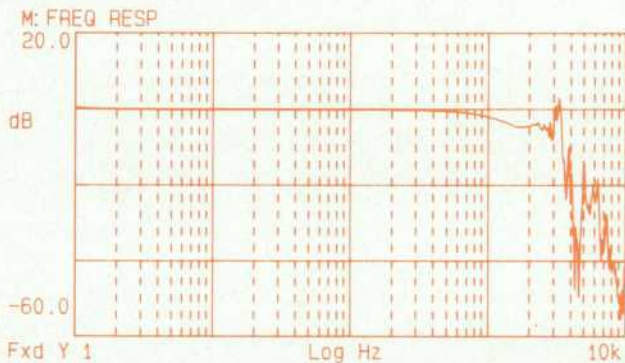
Fig. 14. Phase response with added G_1 element of Fig. 13. This response is curve fitted to help find parameters for an all-pass phase compensation network.

must match the zero locations exactly. We choose $\sigma_1 = -300$ Hz and $\sigma_2 = 1600$ Hz for the pole-zero locations of our all-pass phase compensation filter. This results in a conditionally stable control loop, but the right choice of k will give a stable response. Fig. 15a shows the table of the total pole-zero locations for the combined G_1 loop compensation network. With $k = 80$, the closed-loop system response is as shown in Fig. 15b and Fig. 15c.

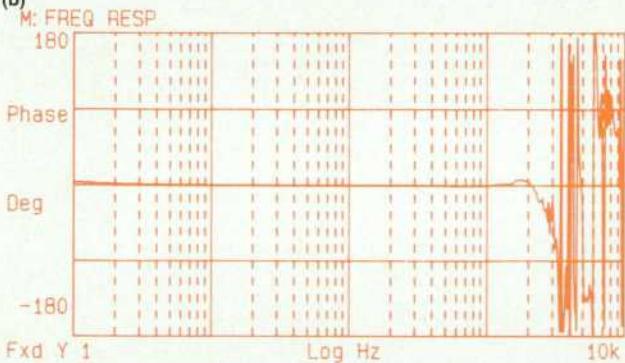
The system has greatly improved flatness, but there are still troublesome system resonances above 3 kHz. Designing a simple precompensation network A with poles at $-400 \pm j500$ for the overall closed-loop feedback system gives the acceptable system response and corresponding step response shown in Fig. 16.

Synthesis				
Poles And Zeros				
	POLES		ZEROS	
	6		4	
1	-170.0 ±j	2.3k	-95.0 ±j	1.83k
2	-250.0 ±j	250.0	-300.0	
3	300.0		-1.6k	
4	1.6k			
Time delay= 0.0 S Gain= -20.0M Scale= 1.0				

(a)

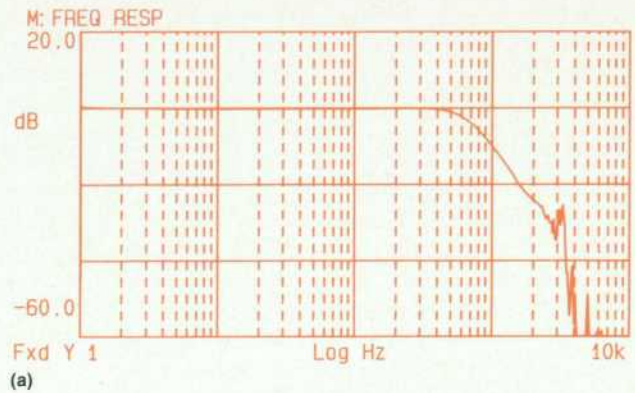


(b)

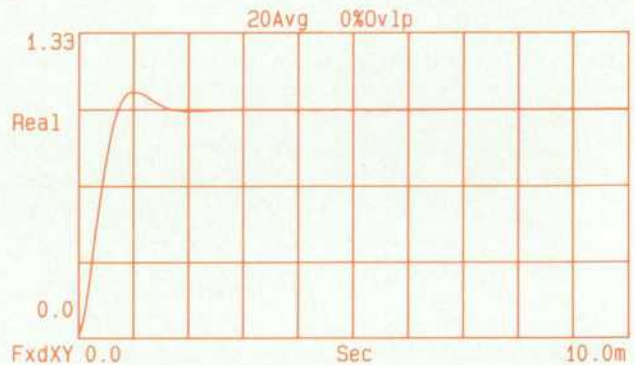


(c)

Fig. 15. (a) Table of poles and zeros for the combined G_1 loop compensation network. (b) Frequency response of closed-loop system. (c) Phase response of closed-loop system. Troublesome resonances still exist above 3 kHz.



(a)



(b)

Fig. 16. (a) Acceptable frequency response gained by adding a precompensation network A. (b) Corresponding step response.

Acknowledgments

Many of the HP 3562A's frequency response synthesis, table editing, and display routines were implemented by Bryan Murray. Conversations with Ron Potter clarified many issues in the synthesis table conversion theory.

Curve Fitter for Pole-Zero Analysis

by James L. Adcock

THE CURVE-FITTING ALGORITHM in the HP 3562A Analyzer finds a pole-zero model of a linear system based on the system's measured frequency response. The curve fitter does this by calculating a weighted least-squares fit of a rational polynomial to the measured frequency response:

$$H'(s) \approx \frac{a_0 + a_1s + a_2s^2 + \dots}{b_0 + b_1s + b_2s^2 + \dots}$$

Then the polynomials in the denominator and numerator can be factored to find the poles and zeros of the measured system (or alternatively the pole-residue or polynomial form). Fig. 1 demonstrates actual curve fits by the HP 3562A for very clean and very noisy measurements.

The curve fitter implements the inverse of the traditional engineering design process, which attempts to predict the measured response of a system based on an analytical model. Instead, the curve fitter attempts to predict the analytical model of the linear system based on the measured response. If we are to close the loop on the engineering design process, both analytical modeling tools and parameter extraction (curve fitting) are necessary as follows:

1. Design a prototype system (using the HP 3562A's synthesis capabilities, see article on page 25).

2. Build the prototype.

3. Measure the prototype's frequency response (using any of the HP 3562A's three measurement modes).

4. Extract the prototype's pole-zero parameters (using the HP 3562A's curve fitter on the measured response).

5. Compare the results to the original design goals.

6. Modify the results (using the HP 3562A's math and synthesis capabilities) to arrive at an improved design.

7. Go to step 2 and repeat the process until the desired design goals are achieved.

Theory

In discussing the theory behind the HP 3562A's curve fitter, the following variables and definitions are used:

ω	sampled, normalized frequency, such that $\omega_{\max} = 1$
ω_{\min}	lowest frequency to be curve fitted
ω_{\max}	highest frequency to be curve fitted
j	$\sqrt{-1}$
$H[\omega]$	modeled frequency response function to be determined
$H'[\omega]$	measured frequency response function
P	numerator polynomial in ω
Q	denominator polynomial in ω
W'	weighting function
$\epsilon[\omega]$	weighted error function
E	sum of the squared error
T_n	n th-order Chebyshev polynomial
Φ	numerator orthogonal polynomials
Ψ	denominator orthogonal polynomials
A	numerator polynomial coefficients to be determined
B	denominator polynomial coefficients to be determined
M^T	conjugate transpose of a matrix M
O	Null matrix, all elements = 0

The theoretical frequency response function of a linear system can be expressed as the quotient of two polynomials:

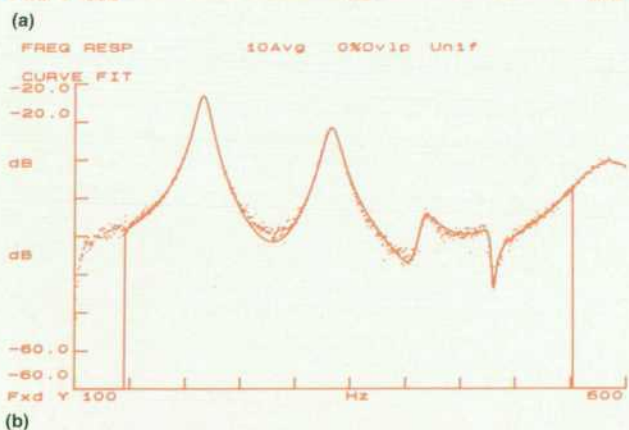
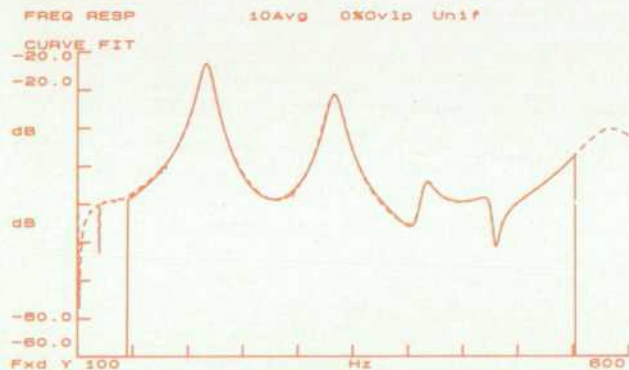


Fig. 1. HP 3562A curve fit to a very clean measurement (a) and a very noisy measurement (b).

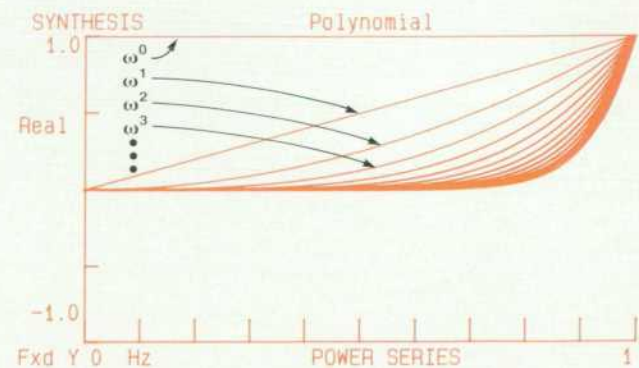


Fig. 2. The shapes of power series become very similar for high orders over most of the normalized frequency range.

$$H[\omega] = P[\omega]/Q[\omega] \quad (1)$$

where ω is the sampled, normalized frequency. This is in rational fraction form. By using any of the linear resolution, log resolution, or swept sine measurement modes of the HP 3562A, we obtain the measured frequency response function:

$$H'[\omega] \quad (2)$$

in real and imaginary form. We want to find the coefficients of $P[\omega]$ and $Q[\omega]$, given that the error, or the weighted difference of $H[\omega]$ and $H'[\omega]$, is minimized. This can be written (leaving out the functionals):

$$\epsilon' = W'(H' - H) \quad (3)$$

where W' is a weighting function to be used in the weighted least-squares derivation to improve the overall quality of the curve fit.

P and Q could be defined to be just the ordinary power series in $j\omega$. Then P/Q would be the ordinary rational fraction representation of a frequency response function. However, ordinary power series have several very bad characteristics when used for curve fitting. First, they have a very large dynamic range. Second, for the higher orders, the shapes of the power series become very similar over most of the normalized frequency range (see Fig. 2). These properties lead to numerically very ill-conditioned matrices when power series are used to derive least-squares equations.

In contrast, Chebyshev polynomials have a small dynamic range, being bounded in amplitude between -1 and 1 , and each Chebyshev polynomial has a unique shape in the normalized frequency range where they will be used (see Fig. 3). This makes these polynomials particularly well-suited for least-squares derivations. Thus, the HP 3562A curve-fit algorithm uses Chebyshev polynomials instead of ordinary power series.

From the following defining equations for Chebyshev polynomials it can be seen that for every series of Chebyshev polynomials of order n , there is an equivalent n th-order power series:

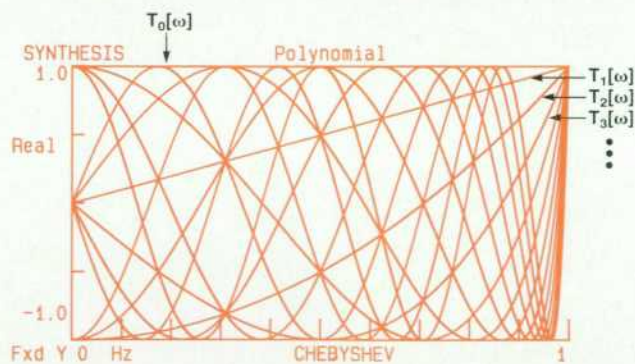


Fig. 3. Chebyshev polynomials have different shapes over the normalized frequency range.

$$\begin{aligned} T_0[\omega] &= 1 \\ T_1[\omega] &= \omega \\ &\dots \\ T_n[\omega] &= 2\omega T_{n-1} - T_{n-2} \end{aligned}$$

also (4)

$$\begin{aligned} T_{-n}[\omega] &= T_n[\omega] \\ 2T_{nm}[\omega] &= T_{n+m}[\omega] + T_{n-m}[\omega] \\ T_n[\omega] &= \cos(n \cos^{-1} \omega) \end{aligned}$$

Thus, after solving for the coefficients of P and Q expressed in Chebyshev polynomials, the equivalent coefficients of the same order power series can be found. For reference, here are the first few Chebyshev polynomials expressed in power series:

$$\begin{aligned} T_0[\omega] &= 1 \\ T_1[\omega] &= \omega \\ T_2[\omega] &= 2\omega^2 - 1 \\ T_3[\omega] &= 4\omega^3 - 3\omega \\ T_4[\omega] &= 8\omega^4 - 8\omega^2 + 1 \\ T_5[\omega] &= 16\omega^5 - 20\omega^3 + 5\omega \\ &\dots \end{aligned} \quad (5)$$

To solve for up to 40 poles and 40 zeros, the HP 3562A curve fitter requires Chebyshev polynomials up to $T_{80}[\omega]$.

The second-to-last equation in equation set 4 is equivalent to the ordinary trigonometric product-equals-sum-plus-difference formula. In addition, consider that the Chebyshev polynomials can be thought of as a frequency warped set of cosines. As a result, many of the concepts of fast Fourier transforms (FFTs) can be used to hasten the solutions of these equations. In fact, we will show later how these trigonometric identities are used to speed up, by an order of magnitude, the calculation of a major portion of the curve-fit algorithm.

Define the two sets of orthogonal polynomials $\phi_i = T_i$ and $\psi_k = T_k$, where T_i and T_k are the i th-order and k th-order Chebyshev polynomials in ω , such that:

$$P = \sum_{i=1}^m a_i \phi_i \quad (6)$$

$$Q = \sum_{k=1}^n b_k \psi_k \quad (7)$$

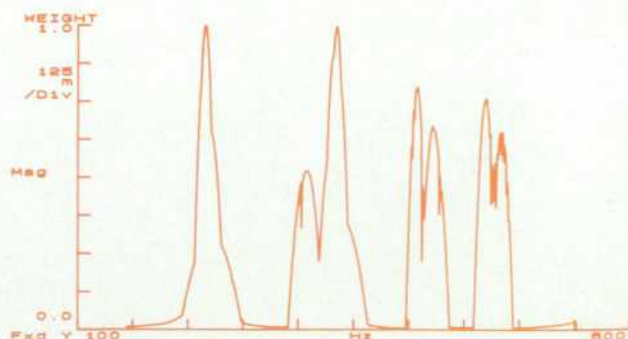


Fig. 4. Weighting function for the curve fit shown in Fig. 1b.

or, in matrix form,

$$P = \Phi A \quad (8)$$

$$Q = \Psi B \quad (9)$$

Substituting equations 1 and 2 into equation 3, and multiplying through by Q (a necessary trick to make the resulting least-squares equations linear), we arrive at:

$$\epsilon = (WQH' - WP) \quad (10)$$

where W is equal to W' with the pole locations deemphasized. Since W must be determined before P and Q are to be solved, W is preweighted by an experimentally derived pole location estimation technique. Then the multiplied-through-by-Q formulation gives very similar results to the original nonlinear P/Q formulation: $\epsilon' \approx \epsilon$. We have found that increasing the weighting of the zero locations, as well as the pole locations, gives improved results. This maximizes the quality of fit when viewed on a decibel scale rather than a magnitude-squared scale. In addition, the weighting function is improved by deemphasizing regions of particularly noisy data. Fig. 4 shows the weighting function corresponding to the curve fit shown in Fig. 1b and Fig. 5 shows the equivalent curve fit if the weighting function is set to unity.

Using equations 8 and 9, the formulation becomes:

$$\epsilon = (WH'\Psi B - W\Phi A) \quad (11)$$

or,

$$\epsilon = (W_b\Psi B - W_a\Phi A) \quad (12)$$

where $W_a = W$ and $W_b = WH'$.

The sum of the squared errors is given by:

$$E = \sum_{\omega=\omega_{\min}}^{\omega_{\max}} (B^T\Psi^TW_b^* - A^T\Phi^TW_a^*)(W_b\Psi B - W_a\Phi A)$$

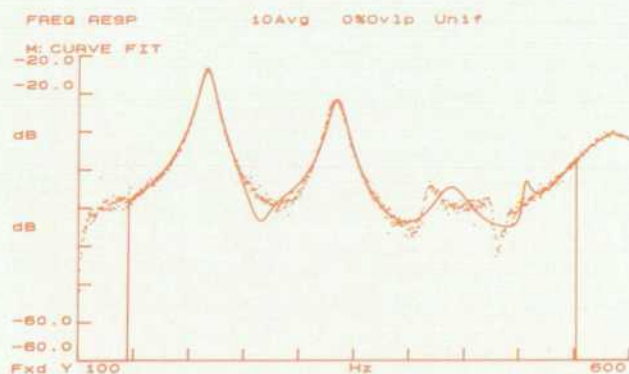


Fig. 5. Curve fit for Fig. 1b when the weighting function is set to unity.

To minimize the squared error, set $\partial E/\partial A$ and $\partial E/\partial B = 0$. This gives two equations:

$$\frac{\partial E}{\partial A} = - \sum_{\omega=\omega_{\min}}^{\omega_{\max}} \Phi^TW_a^*(W_b\Psi B - W_a\Phi A) = 0$$

$$\frac{\partial E}{\partial B} = \sum_{\omega=\omega_{\min}}^{\omega_{\max}} \Psi^TW_b^*(W_b\Psi B - W_a\Phi A) = 0$$

Written in partitioned matrix form:

$$\begin{pmatrix} G & -D \\ -D^T & F \end{pmatrix} \begin{pmatrix} B \\ A \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad (13)$$

where

$$G = \sum_{\omega=\omega_{\min}}^{\omega_{\max}} |W_b|^2\Psi^T\Psi$$

$$D = \sum_{\omega=\omega_{\min}}^{\omega_{\max}} W_a W_b^*\Psi^T\Phi \quad (14)$$

$$F = \sum_{\omega=\omega_{\min}}^{\omega_{\max}} |W_a|^2\Phi^T\Phi$$

Two problems with equation set 14 must be addressed before we have a practical solution to the curve fitting problem. First, $|W_b|^2$ consists of $|W|^2|H'|^2$ where H' typically has some noise on its measurement. Let

$$H' = H'' + e_h$$

where H'' is the ideal frequency response and e_h is the random error caused by measurement noise (assumed to be not correlated with H''). Then the expected value of $|H'|^2$ is:

$$|H''|^2 + \bar{e}_h^2$$

$|H'|^2$ is typically a biased estimate of $|H''|^2$, and was found to lead to biased estimates of the curve fit, and results in

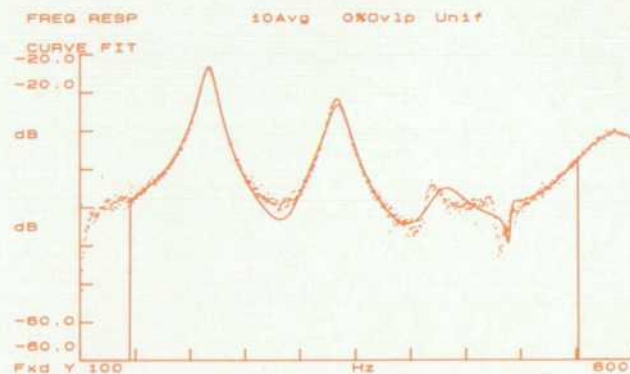


Fig. 6. Curve fit for Fig. 1b when the bias caused by the random error introduced by measurement noise is not removed.

biased pole-zero locations. Once we discovered the source of this bias, the solution was simple: subtract an estimate of \bar{e}_h^2 from $|H'|^2$. Fig. 6 shows the results when this bias effect is not corrected for.

The second problem that must be solved to have a usable algorithm concerns calculation speed and memory storage requirements. A curve fit with 40 poles and 40 zeros requires the solution of an 80-by-80 matrix—6400 double-precision floating-point numbers. This exceeds the memory available in the HP 3562A for storing the matrix. In addition, the 6400 numbers are each constructed from the product and summation of up to 800 measured complex frequency response data points. Thus, several million double-precision floating-point calculations can be required to construct the matrix. We needed a way to reduce these memory storage requirements and to hasten the calculations of the matrix elements if the curve fitter was to be usable for more than a dozen poles and zeros.

Fortunately, the Chebyshev product = sum + difference relationship:

$$2T_i T_k = T_{i+k} + T_{i-k}$$

provides the solution. This relationship can be applied simultaneously to each of the four submatrices **G**, **D**, **D^T**, and **F**, resulting in great savings in calculation time and storage space. The following discussion uses submatrix **G** to illustrate how this relationship is used to obtain the necessary calculation speed and memory savings:

$$\begin{aligned} G_{ik} &= \sum_{\omega=\omega_{\min}}^{\omega_{\max}} |W_b|^2 \Psi_i \Psi_k \\ &= \sum_{\omega=\omega_{\min}}^{\omega_{\max}} |W_b|^2 T_k T_i \\ &= \frac{1}{2} \sum_{\omega=\omega_{\min}}^{\omega_{\max}} |W_b|^2 T_{k+i} + \frac{1}{2} \sum_{\omega=\omega_{\min}}^{\omega_{\max}} |W_b|^2 T_{k-i} \\ &= G'_{k+i} + G''_{k-i} \end{aligned}$$

At first glance, this appears to have doubled the number of summations that need to be performed. However, now many of the summations are identical, so that we only need to perform each different summation the first time we encounter it in the calculation of the submatrix. After that,

when we encounter a particular summation again, we can use its previously calculated value. For example, $T_{k+i} = T_{2+3} = T_{3+2} = T_{1+4} = T_{4+1} \dots$, so that all of the summations along the diagonals of **G'** are identical. Likewise, all summations along the opposite diagonals of **G''** are identical. Thus, for an $i \times k$ submatrix there are only $2(i+k)$ different summations that need to be performed. This reduces the number of summations that must actually be calculated by an order of magnitude. Similarly, **G** no longer needs to be actually stored in memory, since each individual element G_{ik} can be quickly recalculated whenever needed from G'_{k+i} and G''_{k-i} . That is, $G_{ik} = G'_{k+i} + G''_{k-i}$. This solves our memory storage requirements.

Finally, the solution to the homogeneous set of equations (equation set 13) gives the coefficients of polynomials **A** and **B**. Special matrix techniques are required to maintain accuracy for the high-order equations desired. Of the techniques evaluated, a Gram-Schmidt orthogonalization technique¹ gives the most accurate results. Equations 6 and 7 yield the Chebyshev coefficients of the rational fraction polynomials **P** and **Q**. The coefficients are converted to coefficients of the ordinary power series in $j\omega$, the zeros are found using common zero finder techniques,^{2,3} and then the pole and zero locations are renormalized to their original values by multiplying through by $1/\omega_{\max}$. For design examples using the HP 3562A's curve fitter, see the article on page 25.

Acknowledgments

Ron Potter made many contributions to the success of the HP 3562A's curve fitter. Peter Pias added double-precision routines to the floating-point processor so that the curve fitter would have the necessary speed and accuracy. Nick Pendegras and Sandy Atkinson offered early enthusiasm and support at a time when no one on the project really knew if the curve fitter was practical.

References

1. A. Björck, "Solving Linear Least Squares Problems by Gram-Schmidt Orthogonalization," *Nordisk Tidskr. Informationsbehandling*, Vol. 7, 1967, pp. 1-21.
2. W.M. Kahan, "Personal Calculator Has Key to Solve Any Equation $f(x) = 0$," *Hewlett-Packard Journal*, Vol. 30, no. 12, December 1979.
3. L.R. Grodd and C.M. Patton, "ROM Extends Numerical Function Set of Handheld Computer," *Hewlett-Packard Journal*, Vol. 35, no. 7, July 1984.

Authors

January 1987

4 Low-Frequency FFT Analyzer

Gaylord L. Wahl, Jr.



Born in St. Paul, Minnesota, Gaylord Wahl is an alumnus of the University of Minnesota (BEE 1979). He has been with HP since 1979 and has been a hardware designer, hardware project manager, and software quality assurance manager for the HP 3562A Signal Analyzer. Gaylord and his wife live in Everett, Washington. He plays racquetball and golf and enjoys all kinds of music.

Michael L. Hall



Born in Malden, Missouri, Mike Hall attended the University of California at Santa Barbara and received his BSEE degree in 1979. He joined HP the same year and has contributed to the software for the HP 3562A Signal Analyzer. He's now a resident of Marysville, Washington, is married, and has two children. He has a collection of over 8,000 comic books and is a member of the Everett Comic Art Club. He also enjoys programming his home computer.

Edward S. Atkinson



Sandy Atkinson was born in Atlanta, Georgia and attended the Georgia Institute of Technology. His BEE degree was awarded in 1974. He also completed work for an MSCS degree from the University of Santa Clara in 1980. With HP since 1974, he was a soft-

ware project manager for the HP 3562A Signal Analyzer and continues to be responsible for dynamic signal analysis projects. His past projects include software design for the HP 5451C Fourier Analyzer. Sandy and his wife and two sons live in Marysville, Washington and he enjoys fishing for salmon and trout.

Eric J. Wicklund



A Washington native, Eric Wicklund was born in Seattle and attended the University of Washington (BSEE 1977). After coming to HP the same year, he worked on the HP 3497A Data Acquisition/Control Unit, the HP 3326A Synthesizer, and the HP 3561A

Signal Analyzer. He was a project manager for the HP 3562A Signal Analyzer and continues to be responsible for signal analyzer projects. He co-authored a 1984 *HP Journal* article on the hardware design of the HP 3561A. A resident of Woodinville, Eric is married and has three children. He's interested in amateur radio (KR7A) and woodworking and is restoring an old car. He also designed and built his own home.

Steven K. Peterson



Born in Henderson, Nevada, Steve Peterson attended Harvey Mudd College, completing work for a BS degree in engineering in 1980. He came to the Loveland Instrument Division the same year and has contributed to software development for the HP

3562A Signal Analyzer, the HP 3577A Network Analyzer, HP META-MODAL Analysis Software, and HP VISTA Signal Processing Software. He also did mechanical design for the HP 3314A Function Generator. He's currently studying for his MSME degree in applied mechanics at Stanford University through HP's Resident Fellowship program. Steve and his wife and daughter are residents of Marysville, Washington, and his wife is an HP project administrator. His comment on his favorite sport is, "Snow is good. But powder, aah powder!"

17 Measurement/Demodulation

Edward S. Atkinson

Author's biography appears elsewhere in this section.

Raymond C. Blackham



With HP since 1979, Ray Blackham contributed to the signal source, LO, input ADC, and swept sine software for the HP 3562A Signal Analyzer. His work on a swept sine integration algorithm is the subject of a patent application and his professional interests include digital signal processing, numerical analysis, and optimization. He was born in Huntington Park, California and attended Brigham Young University, completing work for his BSEE degree in 1978 and his MSEE degree in 1979. He continued his education at the University of Washington and received an MA degree in mathematics in 1986. Ray and his wife and three daughters are residents of Marysville, Washington. He's active in his church and lists mathematics, languages, hiking, and cross-country skiing as outside interests.

He was born in Huntington Park, California and attended Brigham Young University, completing work for his BSEE degree in 1978 and his MSEE degree in 1979. He continued his education at the University of Washington and received an MA degree in mathematics in 1986. Ray and his wife and three daughters are residents of Marysville, Washington. He's active in his church and lists mathematics, languages, hiking, and cross-country skiing as outside interests.

Ronald W. Potter



Ron Potter started with HP in 1958 and has been an R&D project leader for Fourier analyzers and a section leader for electronic counter development. He designed the digital demodulation section and worked on quality assurance for the HP 3562A

Signal Analyzer. Three patents have resulted from his work on digital filters and DACs and he has written several articles on modal analysis, digital signal processing, and digital demodulation. In addition to his HP experience, he has worked on computer design for automatic machine tool control for Hughes Aircraft Company. He recently left HP and is now a consultant. A native of Carthage, Missouri, Ron earned his BSEE degree from the University of Illinois in 1954 and his MSEE degree from the University of Southern California in 1958. He was also a first lieutenant in the U.S. Army. He's a resident of Cupertino, California, is married, and has two sons. For recreation, he likes photography, hi-fi sound reproduction, and astronomy (he's an astrophotographer).

James A. Vasil



A graduate of the University of South Alabama, James Vasil received his BSEE degree in 1977. He came to HP's Lake Stevens Instrument Division in 1981 and contributed to the FFT processor board and log resolution software for the HP

3562A Signal Analyzer. He has also worked on the throughput feature for HP VISTA Signal Processing Software. James lives in Everett, Washington and enjoys backpacking, skiing, volleyball, and reading in his spare time.

25 Frequency Response Synthesis

James L. Adcock

Author's biography appears elsewhere in this section.

33 Pole-Zero Curve Fitter

James L. Adcock



Born in Seattle, Washington, Jim Adcock attended the Massachusetts Institute of Technology and received his BSEE degree in 1978. After working as a test engineer at the Boeing Company, he came to HP in 1980 and worked on the curve fitter for the HP

3562A Signal Analyzer. He has also contributed to the development of the HP 3565S Signal Processing System. He's coauthor of a paper on a curve-fitting algorithm and is named inventor on two patent applications. Jim is a resident of Marysville, Washington and is interested in skiing, kayaking, volleyball, and electronic music.

38 Series 70 Cache Performance

James R. Callister



With HP since 1981, Jim Callister holds a 1978 BS degree in computer science and mathematics from Brigham Young University and a 1981 MSEE degree from the University of Utah. His work on the HP 3000 Series 70 Computer centered on performance

engineering and he has designed computer hardware and microcode for a number of other products. Before coming to HP he was involved in a broad range of R&D development activities for computer hardware and software. A California native, Jim was born in San Mateo and now lives in Santa Clara. He's married and is active in his church and in public services for amateur radio operators. (His call letters are N7CBT.) His other interests include volleyball and French novels.

Craig W. Pampeyan



A product engineer for the HP 3000 Series 70 Computer, Craig Pampeyan has been with HP since 1981. He was the principal hardware designer for the Series 70 Computer and has also worked on the HP 3000 Series 37 and Series 68 Computers. His other

HP experience includes a stint at the Guadalajara, Mexico Computer Operation. He was born in San Francisco and graduated from California Polytechnic State University at San Luis Obispo with a BS degree in electronic engineering in 1981. He also has a 1984 MSEE degree from Stanford University and is currently studying for an MBA degree from the University of Santa Clara. Craig lives in Santa Clara, California and holds a private pilot's license with an instrument rating. His other interests include skiing, photography, and travel.

Performance Analysis of the HP 3000 Series 70 Hardware Cache

Measurements and modeling pointed the way to improved performance over the Series 68.

by James R. Callister and Craig W. Pampeyan

THE HP 3000 SERIES 70 is a recent addition to the HP 3000 Business Computer product line. Its design objectives were to provide a significant upgrade to the Series 68 in a short design cycle. The performance of the Series 70 was engineered through adherence to a strict methodology of measurement, modeling, and verification. This paper gives a description of the methodology with examples drawn from the major component of the Series 70—the cache memory subsystem.

Cache memories are notorious for being poorly characterized.¹ At the system level, caches appear to be nondeterministic, which makes them very difficult to model. In addition, they are very sensitive to the system workload. The design of a cache provides a severe test for any estimation methodology.

An outline of the generic performance engineering cycle is shown in Fig. 1. The steps of characterization (modeling, design analysis and tracking, benchmarking and product testing) are straightforward. Yet rigorous adherence to precision in each step returns enormous dividends in overall accuracy, thus ensuring a product's success.² A precise methodology includes the use of sampling theory, statistical analysis, and measurement validation. The overall methodology includes the additional benefit that the field tests in the final stage of a product design are also used to provide customer characterization data in the first stage of the next product.

Current System Characterization

The major goals of the Series 70 were to maximize performance improvement and minimize change to the Series 68. These conflicting goals were resolved by extensive measurements of current Series 68 systems. Measurements of systems in the field indicated how the systems were being used and which system components were being stressed. Measurements were also used to construct a characterization of the customer base. Any proposed performance enhancement could then be evaluated in terms of its effect across the entire customer base.

Measurement tools play a crucial role in the performance design methodology. It is important for tools to be chosen that both accurately measure the metrics of interest and are easy to use (see "Measurement Tools," next page). Sometimes the right tool does not exist. In a development lab, late data is the same as no data. If a tool must be developed, the development time must be short enough to deliver timely data. The Series 70 performance design team made use of both existing tools and new tools. The new

tools were largely leveraged from existing products.

HPSnapshot

The premier measurement tool for HP 3000 systems is HPSnapshot. HPSnapshot is a suite of software collection tools, reduction programs, and report generators. The tools measure such things as peripheral activity, system resource utilization, process statistics, and file use. The collected data is stored on tape and sent to a dedicated system for reduction. HPSnapshot is easy to use and allows many systems to be measured quickly.

HPSnapshot runs were made on fifty Series 68s in the field. Individual runs were analyzed and selected metrics were stored in a data base. This data was used to characterize customer workloads and to analyze bottlenecks restricting additional performance. The measurements and resulting analysis showed that the best performance opportunities lay in improving the CPU. Several teams were created to investigate the areas suggested by the data (see "Series 70: Not Just a Cache," page 40).

Hardware Monitor

The HPSnapshot data suggested that more extensive measurements be made on the CPU itself. Several tools were created to make these measurements. One of these tools,

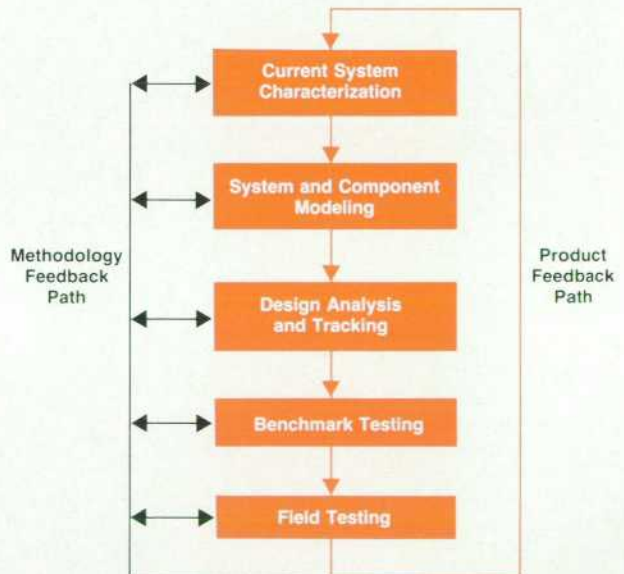


Fig. 1. Outline of the generic performance engineering cycle.

Measurement Tools

Computer systems are extremely complex, requiring comprehension at many levels. Measurements, too, must be made at different levels to provide a complete system characterization. For a particular metric, it is important to choose the best tool. The tools can be compared in terms of what they can measure, their ease of use, the overhead they incur, and their sampling frequencies. Table I compares the tools for each category.

Hardware measurement tools analyze signals from the machine under test. Most hardware tools are passive and do not disturb the system. They are fast enough to capture individual machine cycles and therefore are very good at collecting low-level traces and sequences. The short-term sampling frequency can be very high. Hardware tools cannot measure higher levels of the system such as processes and programs. They require equipment attached to the system under test and extensive setup time.

Machines such as the HP 3000 Series 68 have writable control store, allowing the use of microcoded tools. The sampling rate is slower than hardware tools and the extra microcode takes some minimal amount of system overhead. Microcode tools are best suited for obtaining statistics at the procedure and process level. They are also very good at sampling at fixed time intervals. The installation of the tools is straightforward but does require a cool start of the system.

The most abstract level of measurement requires software running on the system under test. There can be significant overhead associated with software tools, which may also perturb the system. But software tools can track paths in system level algorithms, monitor interactions between processes, and log a large number of software event counters. Software tools are generally very easy to install and use.

Table I
Comparison of Measurement Techniques

Monitor	Overhead	Ease of Use	Sampling Frequency	Type of Measurements
Hardware	0%	Hardware required	10^3 - 10^6 /s	Signals, short traces
Microcode	0.1-1%	Cool start required	10^3 /s	Fixed-time sampling, procedures, processes
Software	5-10%	Simple Installation	10^2 /s	Software event counters, system interactions

The Series 70 performance analysis made use of each type of tool. Besides the HPSnapshot and hardware monitor tools mentioned in the accompanying article, several microcode-based tools were used.

Microcode Tools

The Instruction Gatherer records the current executing instruction at one-millisecond intervals. It also gives some information about the most frequent subcases of the instructions. The data from ten sites gave a time-weighted profile of which instructions were executed most often. This information led to the remicrocoding of two instructions for increased performance.

The Microsampler is a simple microcoded version of the Sampler software tool. It records program counter values in the operating system code. From this information, six high-frequency procedures were identified and rewritten in microcode.

The cache post microcode is special-purpose microcode used to investigate alternative solutions to cache posting. The microcode validated the use of the cache simulator for the posting circuitry investigation.

the hardware monitor, provided invaluable measurement capability.

The monitor consists of an HP 1630 Logic Analyzer coupled to an HP Touchscreen Personal Computer via the HP-IB (IEEE 488), as shown in Fig. 2. The probes of the HP 1630 are attached to pins on the backplane of the system under test (HP 3000 Series 64, 68, or 70). The Touchscreen computer serves as controller, reduction tool, and data storage for the HP 1630.³ The monitor can automatically run a series of independent tests with a simple command file.

For each test, the Touchscreen computer begins by downloading configuration information to the HP 1630. The measurement is then started on the HP 1630. The Touchscreen waits for a specified time, then halts the measurement. The collected data is uploaded to the Touchscreen computer where it is reduced and stored to disc. A side benefit of the automated process is that the uploaded data from the HP 1630 is more detailed than that available through manual operation. Careful analysis of the internal operation of the HP 1630 gave us confidence that it would satisfy statistical sampling demands.

A collection of ten tests was run on four HP 3000 Series 68 systems. These systems were chosen based on how well they represented the customer base, as determined from the HPSnapshot data. The tests measured many variables

including instruction paths, I/O use, and cache statistics. The tests required a total of 59 probes. It was calculated that half-hour samples would best meet the conflicting de-

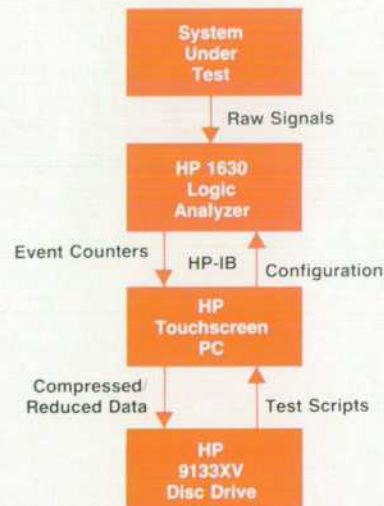


Fig. 2. The hardware monitor performance measurement tool.

The Series 70: Not Just a Cache

The HP 3000 Series 70 Business Computer is a collection of performance enhancements. To be included in the Series 70 product, each enhancement had to qualify in terms of measurable performance, applicability across the customer base, and independence of the other enhancements. Each enhancement was the result of the methodology of measurements and analysis used in the Series 70 cache design and described in the accompanying article.

Microcode was written to find the most common instruction executions. Two instructions that were surprisingly prominent in the mix were rewritten to execute faster.

Microcode was also written to find the most often used MPE procedures. This information led to the selection of six procedures to be rewritten in microcode. The microcoded versions of these procedures execute up to ten times faster.

The expanded main memory offered on both the Series 68 and 70 was also subjected to measurement and analysis. The effect of additional memory on the multiprogramming level, the number of physical I/Os, and the CPU utilization were studied. Methods to identify when main memory bottlenecks occur were developed, which resulted in projections of performance improvements resulting from additional memory.

There were also several improvements in the MPE operating system software. The HPSnapshot tool revealed areas where changes could have a significant effect on system performance.

Analytic models helped estimate the relative merits of several proposed algorithmic changes in the memory manager, dispatcher, and disc driver. The most beneficial changes were implemented by the MPE software engineers. They were then run through many benchmarks to verify the performance improvements.

Although not part of the Series 70, the HP 7933/35XP cached disc drive was also a result of the performance engineering cycle described here. Important workload parameters were identified. A comparative study of the I/O subsystem performance with MPE disc caching, no caching, and the HP 7933/35XP was conducted. Engineers at HP's Disc Memory Division worked closely with the commercial systems performance engineers to model various design alternatives. Benchmarks were run on prototypes of the enhanced disc drives and the performance gains were verified. The system workload characteristics under which the cached discs performed best were explicitly identified to ensure customer satisfaction.

All of these components were evaluated with respect to each other and to the system as a whole. The performance increases of the components complement each other and keep the system balanced. The result of the Series 70 project is a product offering a 20% to 35% increase in system throughput over a Series 68. Expanded main memory and the HP 7933/35XP can provide additional performance improvements.

mands of aggregate and variance analysis. Each sample captured about half a million cycles, and a total of 144 samples were collected.

The data gave insight into the low-level activities of the CPU. As a result, a number of performance opportunities were identified. The biggest opportunity lay in improving the cache memory subsystem.

The Series 68 cache is a 4K-word, 2-set cache (see "How a Cache Works," page 42). The hit rate measured on the four systems was 92.5%. However, although a cache miss occurs only 7.5% of the time, almost 30% of the CPU time is spent waiting for cache misses to be resolved. During this time, the CPU is frozen and cannot proceed. A simple model of the cache was created and validated through the hardware monitor. The model suggested that if the hit rate could be improved by 5 or 6 percent, the CPU would freeze only about 10% of the time. This would result in a savings of almost 20% of the CPU cycles, which translates into an effective speedup of about 25%. The availability of denser RAMs and programmable array logic parts (PALs) indicated a strong possibility for just such an improvement.

Modeling

Modeling extracts the essential characteristics of a system and converts them into a form that is easily evaluated and modified. There are two major types of modeling. Analytic modeling computes steady-state values of a system according to laws of queuing theory.⁴ Models are convenient and can guarantee correct results if the input data is correct and complete. The disadvantages lie in restrictions placed on the type of environments that can be modeled. Typically, simplifications must be made to the environ-

ment to make the analysis tractable.

Simulation is also modeling, but at a lower level of abstraction. In a Monte Carlo simulation model, the key system variables are represented by their measured probability distributions. Random numbers are applied to the model to generate specific values for the key system variables. These specific values are then used to compute the output variables of interest.

Trace-driven simulations are at a still lower level of abstraction. Traces of values for each of the key system variables are collected and applied together to a deterministic model of the system. The output variables of interest are computed for the specific set of trace data.

Simulation models can be constructed and solved for virtually any system. However, although simulation models provide valuable information, their limitations must also be recognized and understood. For example, simulation models cannot guarantee a steady-state solution, but only a particular solution corresponding to the input data.

Any model runs the risk of ignoring some unknown, yet essential feature of the system. Additional dangers exist in the measurement of system variables and construction of the model. Careful modeling subjects the recorded variables to independent tests of correctness. The model construction can then be validated by modeling the existing system and comparing the results to measurements of the existing system.

After validation, the model is used in design analysis. Various changes can be introduced and the performance changes observed. The set of changes that maximizes performance (and satisfies constraints of cost, design time, etc.) is then chosen as the final design. The final design is

then modeled and the performance for the product predicted.

There are currently no good system variables that will accurately predict how a cache will perform. This makes it impossible to construct analytic or Monte Carlo simulation models that accurately predict cache performance. Currently, the best way to model caches is through trace-driven simulators. The Series 70 cache design used both analytic and trace-driven simulation. A simple analytic model at the system level was constructed using the results of the trace-driven cache simulator.

The cache modeling process involved collecting traces of memory reference addresses. Software was written to simulate various cache organizations. The traces were then run through the simulator under various cache organizations.

Ideally, the traces would have been collected from a Series 68. However, the speed of the Series 68 prevents data collection without special high-speed hardware. Instead, a Series 37 was chosen. A special memory controller that reads the memory reference address and writes it to its own local memory was constructed quickly. The special controller is passive and traces all system activity, including the operating system. One million consecutive memory references can be collected. This number is sufficient to guarantee many calls to the operating system, and also includes many task switches. Three different customer environments were run to collect the data. Approximately nine measurements were collected for each environment, for a total of 30 million memory references (see "Realistic Cache Simulation," page 45). The traces were then subjected to a series of tests to confirm that the collected data was correct.

Cache Simulator

The cache simulator was developed in parallel with the collection process. The simulator takes the trace data and applies it to various models of the cache. The simulator development consisted of two phases. The first phase concentrated on completeness and correctness of the model implementation, the ease of use, and the choice of statistics to be kept. These goals led to a modular structure, very detailed statistics, and gave considerable freedom in adding and altering features. This flexibility has allowed the simulator to be leveraged to model caches for several other machines besides the Series 70.

It is extremely important that the simulator model the cache designs accurately. Artificial data was generated and simulated and then compared with the expected results to verify the accuracy of the cache simulator. Next, the Series 68 cache was modeled with the trace data. The same environment was then run on the Series 68 and the actual cache statistics were collected with the hardware monitor. The close correlation between the simulated and actual results gave a high level of confidence that the simulator would provide reasonable information on which to base the design of the Series 70 cache.

Phase two of the simulator development concentrated on the speed of the simulator by porting it to a mainframe computer. During the port, unnecessary code and structure were eliminated. The verification tests were then run again

and compared to the original simulator output. The streamlining and port to the mainframe resulted in a functionally equivalent simulator that runs 80 times faster. A simulation of a single trace of one million memory references now takes about 45 seconds.

The simulator has the ability to vary seven different parameters: total size of the cache, associativity, block size, algorithms for handling writes, block replacement strategies, the handling of I/O requests, and tag indexing. Simulations were run varying each of the parameters. The effect of each parameter on cache performance and the sensitivity of performance to each parameter were determined.

Fig. 3 shows an example of the simulation results. The graph shows that the biggest contributor to cache performance is the size of the cache. The effect of diminishing returns with increasing cache size is clearly seen. Fig. 3 also shows the effect of associativity on different sizes of caches. The increased complexity of a multiple-set cache can be weighed against the performance gain it provides. This information was computed for all parameters, and the best combination of performance, complexity, and cost was determined. The final design was then chosen and simulated. The Series 70 cache size is 64K words, which is 16 times larger than the Series 68 cache. Like the Series 68, the Series 70 cache has 2 sets.

The simulator provides cache performance information such as hit rates. It does not, however, provide a higher-level view of system performance, such as throughput. In particular, the memory reference traces do not include any measurement of time between memory references. The hardware monitor provides such information for the Series 68. An analysis of the system showed that the timing information would be valid for the new cache. The simple analytical model uses both the simulator data and the hardware monitor data to produce estimates of the effect of the new cache on the system. Besides the expected values for cache statistics, the model also shows a saturation effect. The lower the hit rate a system has, the more the new cache will benefit it.

While the cache simulation traces were fairly stable, the data from real Series 68s had more variance. Statistical analysis of the data led to a range of values for the percentage of CPU cycles that could be recovered by the Series 70 cache. A 90% confidence interval was chosen for the range, which means that, given normal distributions and random sampling, the mean of the cycles recovered should lie within the range nine times out of ten. The range given by the analysis was 19.4% to 28.3%, with an estimated mean of 23.8% recovery of "lost" CPU cycles. Because it was known that the traces from the Series 37 would give optimistic results (since main memory for the Series 37 is several times smaller than the Series 68), the target for the Series 70 cache was set at 20%.

The other components of the Series 70 underwent similar analysis. The analyses were then combined to give a prediction of the overall system performance gain. Care was taken to estimate the overlap effects exhibited when two or more components try to free the same resource. Predictions of the bottlenecks within the new system were also made.

At this point, the performance prediction was used by the marketing department to start working on the pricing

Series 70 Cache Simulation

Hit Rate versus Total Cache Size Versus Associativity

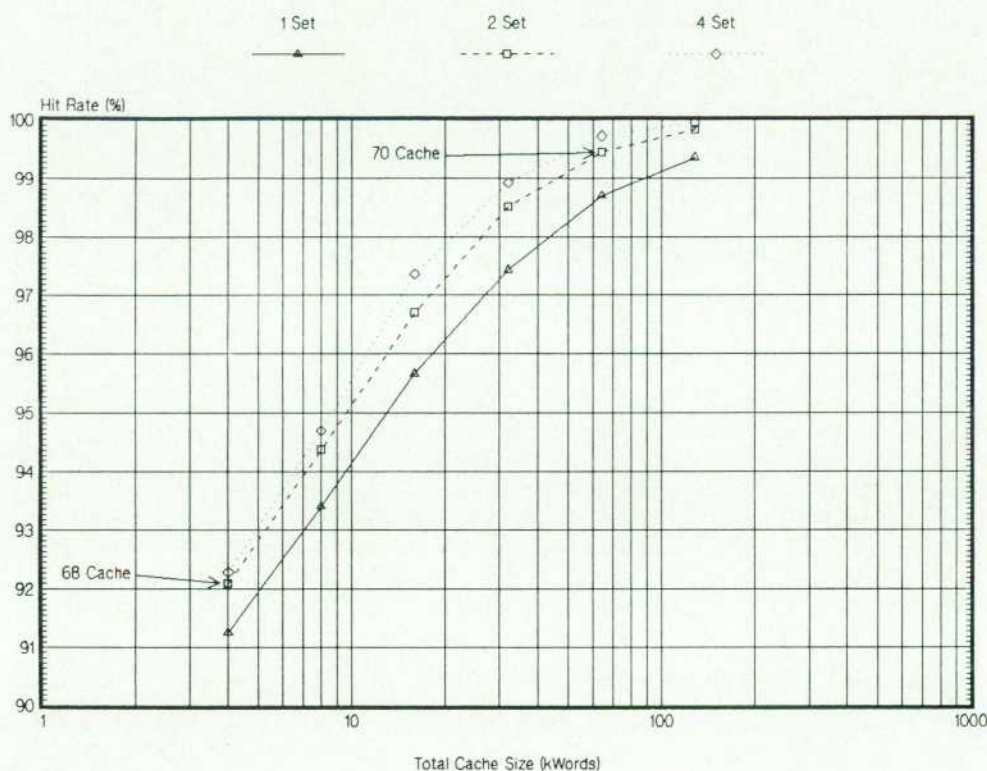


Fig. 3. An example of simulation results, showing that cache size has the greatest effect on cache performance.

and positioning strategy for the product. Since the normal approach is to wait for the actual hardware to become available to measure the performance gain, valuable time was saved in the introduction process through the use of the highly accurate simulation results.

Design Tracking

Measurements and modeling require a lot of work that precedes product development. Not only do they help establish the correct design, but they are also often valuable during the development phase. Unexpected design changes

How a Cache Works

The purpose of a cache memory is to reduce the average effective access time for a memory reference. The cache contains a small high-speed buffer and control logic situated between the CPU and main memory.¹ The cache matches the high speed of the CPU to the relatively low speed of main memory.

Caches derive their performance from the principle of locality. This principle states that, over short periods of time, CPU memory references from a process tend to be clustered in both time and space. This means that data that will be in use in the near future is likely to be in use already. It also means that data that will be in use in the near future is located near data in use already. The degree to which systems exhibit locality determine the benefits of a cache. A cache can be as small as 1/2000 the size of main memory, yet still have hit rates in excess of 90% under normal system loads.

The cache takes advantage of locality by keeping the most often used data in a high-speed buffer. When the CPU requests data, the cache first searches its buffer. If the data is there, it is returned in one cycle; this is termed a cache hit. If the data is not there, a cache miss occurs and the cache must retrieve the data from main memory. As it does so, it also retrieves several more words of data in anticipation that they will soon be requested

by the CPU. Servicing the miss can take many cycles.

To achieve one-cycle access, it is not possible to search the entire cache for the desired data. Instead, every word of data is mapped to (or associated with) only a limited number of places in the cache. When the data is requested, these places are searched in parallel. Each part of the cache that can be searched in parallel is called a set. One-set and two-set caches are most common because of their reduced complexity.

Cache designs also include decisions on how much data to fetch for each cache miss and how to decide which data to replace on a miss (for two or more sets). The Series 70 simulations confirmed previous academic work that the most important factor in cache performance is the size of the cache. The simulations also showed that the design decisions for the other factors of the Series 68 cache were still appropriate for the Series 70 cache. Thus the performance could be achieved with a minimum of change to the operating characteristics of the Series 68 cache.^{1,2}

References

1. K.M. Hodor and M.E. Woodward, "A High-Performance Memory System with Growth Capability," *Hewlett-Packard Journal*, Vol. 33, no. 3, March 1982.
2. A.J. Smith, "Bibliography and Readings on CPU Cache Memories and Related Topics," *Computer Architecture News*, Vol. 14, no. 1, January 1986.

can be quickly modeled and the impact on system performance assessed. The measurement tools can be valuable aids in debugging the prototype. The Series 70 cache development benefitted from these uses of the modeling and measurement tools.

One of the requirements for the cache was to be able to post all modifications of its buffer to main memory in the event of a powerfail. Because the Series 70 had to be backwards compatible with the existing Series 64 and 68 machines, it had to work with the two types of power supplies on those machines. It was determined that the older power supplies had insufficient carryover time to complete the worst-case post.

Several alternatives were proposed to solve this problem. The first alternative was a microcode routine that would, for the older systems, routinely post all modified data from the cache to main memory. The second alternative was circuitry to monitor the cache and post the data whenever the amount of modified data exceeded a certain threshold.

To choose between the alternatives, it was necessary to simulate them. It was very easy to modify the cache simulator to evaluate the posting alternatives. However, there was concern over whether the trace data would be accurate for this type of simulation. Microcode was written for the Series 68 to examine the percentage of the cache that contained modified data. The microcode was executed every half second across several machines for several days of prime-time work. A total of 500,000 samples were collected. The simulator also computed the same information for a simulated Series 68. Fig. 4 shows the two distributions,

which agree closely. This provided the confidence needed to use the results of the simulator for estimating the performance impacts.

The simulator results quickly ruled out the use of the microcode solution. It remained to calculate the performance impact of the posting circuitry. The Series 70 was simulated with different values of the posting threshold. The results showed minimal impact and the posting circuitry was included.

The measurement tools also helped in the effort to debug the initial prototype. The programs used to validate the measurement tools were now used to validate the operation of the new cache. In one case, a failure that occurred only after hours of operating system testing took just seconds to reproduce with the validation software. This helped track down the offending bug quickly and make the prototype ready for further testing. In another case, when the simulator results were being correlated with the actual hardware results, there continued to be a significant, unexplainable discrepancy between the actual results and the simulator results. It turned out that a defective component in the hardware was causing a performance degradation, but not a fatal system error. Once the component was replaced, the actual measurements correlated with the simulator results very closely.

Benchmark Testing

Benchmarks are repeatable workloads designed to exercise a system or system component. The benchmarks are first run on a standard system to establish a baseline. Usu-

DISTRIBUTION OF MODIFIED CACHE DATA

Series 68 Simulated and Measured Distribution

SIMULATED
DISTRIBUTION

MEASURED
DISTRIBUTION

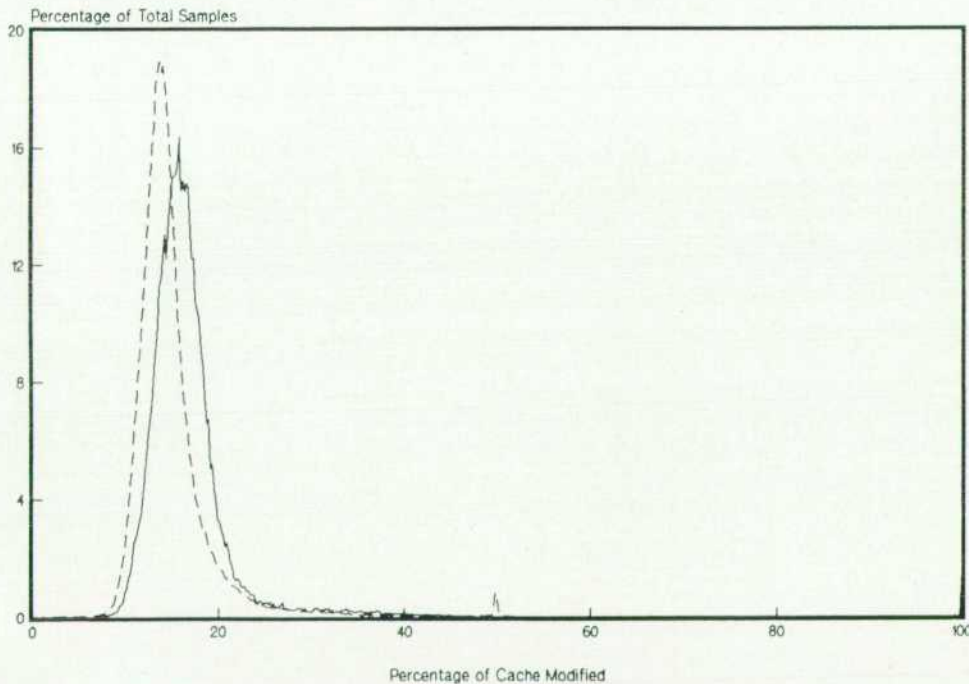


Fig. 4. Comparison of simulated and measured results for the percentage of the cache that contained modified data.

ally several runs are made, varying the number of users, the amount of main memory, or other primary variables of interest. The benchmarks are then run again with the new system and results are compared.

While a model predicts the effect of performance enhancements, a benchmark can be used to measure the actual effect. The repeatability of the benchmark is a key advantage in measuring performance changes. Customer systems, while more realistic, have constantly changing workloads and environments. Hence it is much more difficult to extract the effects of the new product from customer systems.

Benchmark selection and creation constitute a major effort.⁵ The value of a representative benchmark cannot be overstated. It is also valuable to include benchmarks that have extreme values of important system variables. Sensitivity to these variables can then be calculated and used to provide performance estimates for classes of customers, not for just the typical case. The benchmark results are analyzed in light of the model predictions. Together, a refinement of estimates can be made for customer systems.

The benchmarks used for the Series 70 are designed to be representative of heavy interactive data base use. Data base activity represents the majority of work done on the Series 68. The benchmarks are created by taking a complete snapshot of a quiescent customer system, including a track-by-track dump of each disc. Monitors installed on each terminal record every keystroke. The system is then run normally for at least an hour. The traces of terminal activity are then turned into scripts for a terminal emulator.

To run a benchmark, the system and discs are restored to look exactly like the original system. The terminal emulators are run on an HP 3000 Series 48 connected to the terminal ports of the target system. The number of users can be varied by enabling or disabling emulators. Each benchmark is run for one hour. The repeatability of the benchmarks is typically well under a 1% change in transaction throughput.

The baseline for the benchmarks was an HP 3000 Series 68 running the MPE V/E operating system. The number of users and the amount of main memory were varied over a total of seven runs. The system then had the Series 70 hardware cache added, keeping everything else constant (except for small changes in the microcode required to support the cache). The benchmarks were run again under the same parameters. All benchmark runs used both the hardware monitor and the HPSnapshot collection system.

Several metrics are required to understand the effect of the Series 70 cache completely. The results of the benchmark runs are shown in Table I.

Table I

Hardware Metric	Before	After	Difference
Hit Rate	91.0%	98.2%	7.2%
Frozen Cycles	34.8%	11.6%	23.2%
Paused	8.0%	14.0%	6.0%
Software Metric	Improvement		
Transactions/hour	18.9%		
Transactions/CPU second	32.3%		

The hardware metrics show the average measured values before and after the addition of the Series 70 cache. The hit rate of the Series 68 baseline benchmarks is lower than that of the simulation runs, and the Series 70 cache benchmarks show a lower hit rate than the simulations. However, the difference between the hit rates for the benchmarks is similar to that for the simulations. Since the performance increase is determined by the hit rate difference, the results are still significant. "Frozen cycles" indicate the percentage of CPU time that is spent waiting for a cache miss. Essentially no useful work can be accomplished during this time. The time that the CPU is paused waiting for I/O has been factored out of this metric. The difference in the percentages of frozen cycles is the primary metric of raw CPU performance improvement. Again, the simulation and benchmarks are in close agreement in the difference in the percentages of frozen cycles between the Series 68 and 70 caches. The percentage of time that the system was paused also increases with the Series 70 cache. This is an indication that the system can tolerate additional load with nearly the same response time.

The software metrics presented here use transactions as the basic unit of measurement. Transactions per hour is a customer oriented metric. It is an indication of how much interactive work the system is processing. A transaction is defined as the work the system does between the time the **Return** or **Enter** key is pressed until the system replies and posts another read at the terminal. Transactions per hour cannot be averaged over the benchmarks, since each workload has its own mix of simple transactions (an MPE command, for example) and complex transactions (a data base query). The improvement, however, can be averaged. The benchmarks show an improvement of 18.9% more transactions per hour with the Series 70 cache. The metric transactions per CPU second is based on transactions per hour but factors out the percentage of time paused. It is a measure of the improvement the system is capable of delivering when loaded with potential bottlenecks discounted. The 32.3% measured improvement gave a high level of confidence that the Series 70 cache would be successful.

The initial cache performance estimates were then reevaluated with this new data. Since these benchmarks are data base oriented and include up to 78 active users, the benchmarks tend to push the system fairly hard. Since busy systems will benefit most from the improved cache, the benchmarks were estimated to be slightly optimistic. The benchmark runs, therefore, led to changing the upper bound for performance improvement from 28.3% to 23.2%.

More extensive benchmark runs were made later with all of the Series 70 components. A careful study had been made beforehand to ensure that the components didn't fight over the same resource and that the performance increases would mesh well. The benchmark runs showed that the integrated Series 70 met or exceeded expectations under all the conditions tested.

Field Testing

Field testing is an integral part of the product life cycle. It tests all aspects of product functionality, including reliability and documentation. Field testing should also be an integral part of the performance engineering cycle. Mea-

Realistic Cache Simulation

Trace-based simulators are heavily dependent on the traces used. It is generally very difficult to collect memory reference traces for cache simulators. It is even more difficult to obtain representative traces of the system being studied. Alan Jay Smith of the University of California at Berkeley surveyed a large number of cache studies and found that very few contained realistic traces.¹ His article points out six major pitfalls in memory reference tracing:

1. A trace represents only a small sample of the workload. To combat this deficiency, many long traces are needed. The traces must be long enough to overcome start-up effects of the simulator and also long enough to capture major perturbations to the cache. Smith's study used 49 traces of varying lengths for a total of 13.5 million memory references. Each trace in the Series 70 effort is one million memory references long. Very few previously published studies have used traces of this magnitude. One million memory references virtually guarantees the capture of all essential influences to the cache. In addition, less than 1% of the trace is used in the start-up of the simulator. A total of 30 million references were used in the Series 70 cache study. This amount of data is significant. No study known to the authors has used as much data from a single machine. The statistical analysis of the simulations also indicates that the 30 million references are sufficient to make good predictions.

2. Traces are usually taken only from the initial part of small programs. This is because the large majority of studies use program interpreters as the means of collecting the traces. The interpreters have a large overhead and can usually handle only small programs, and then only the initial part of the program's execution. This is not representative of either the program or the system. The Series 70 study used passive hardware to monitor the addresses. The sampling was random and covered all parts of the program execution for both large and small programs. The random sampling ensured that the traces were representative.

3. Most studies do not trace operating system code. Another result of using interpreters is the inability to trace the operating system. However, it has been shown that operating systems have rather poor locality compared to user programs. Current operating systems consume a major portion of CPU time, so their effects must be included for accurate representation. Again, the passive hardware tracing, as used in the Series 70 analysis, automatically includes operating system effects. This is a major contribution to the state of the art in cache simulation.

4. Task switches, which can greatly impact cache performance, are not adequately accounted for. Most simulators

that use interpretive tracing flush the cache every so often to simulate the effects of task switching. If the distribution of task switches is not well understood, serious errors may develop. Hardware tracing eliminates the guesswork, since task switches are included if the trace is long enough. Each trace taken for the Series 70 study was analyzed and found to have dozens of task switches.

5. The sequence of memory addresses is dependent on any buffering implemented on the machine, and on the architecture of the machine. The traces taken for the Series 70 came from the Series 37, which has the same architecture as the Series 70 and no internal buffering. While all architectures exhibit the same general cache behavior with respect to design parameters, absolute numbers for cache performance must depend on having similar architecture and workload.

6. I/O activity is seldom included in the traces. For this reason, not much has been known on the effect of I/O activity on cache behavior. However, the bus structure of the Series 37 permits the collection of I/O references as they occur. Therefore, the simulations accurately model the effects of I/O on cache performance, which leads to improved predictions.

The traces taken for the Series 70 advance the state of the art in cache simulation. The predictions of cache performance agree closely with measured results. However, there is still room for improvement. The Series 37 used in the tracing only had 2M bytes of main memory. The Series 70 can have up to 16M bytes. The Series 37 also did not have MPE disc caching enabled. It is anticipated that many Series 70s run with disc caching. These deficiencies were recognized early and their effects were compensated by conservatism in the analysis.

The Series 70 analysis spawned a more accurate hardware tracing system. It can record a million cycles of machine execution in real time for many high-speed systems. Up to 144 bits can be collected during each cycle. The system consists of 18M bytes of very high-speed RAM, a Series 37 acting as a controller, an HP 9144 cartridge tape drive for data storage, and interface circuitry to the system under test, all on a portable cart. It is currently being used with the HP 3000 Series 930 and is providing state-of-the-art measurement capabilities for this RISC-like HP Precision Architecture machine.

Reference

1. A. J. Smith, "Cache Evaluation and the Impact of Workload Choice," Report UCB/CSD85/229, March 1985, *Proceedings of the 12th International Symposium on Computer Architectures*, June 1985, pp. 64-75.

measurements under real conditions are the final validation of the product.

Measurements also help validate the whole performance methodology. Results of field testing are compared with the models to gauge the accuracy of the prediction techniques. Sometimes the measurements suggest how the models can be improved. Field measurements also validate the benchmarks. The performance increase shown by the benchmarks can be compared to those seen in the field to show how representative the benchmarks really are.

Field testing does not always produce the most accurate data on the effects of a new product. The dynamics of production systems in the field sometimes make it difficult to distinguish between new product effects and normal

system variations. One way of overcoming this limitation is through long-term testing. Patterns of normal variation can be distinguished and accounted for when many samples are collected over a long time.

Site selection for field testing resembles benchmark selection. A mixture of representative sites and sites with unique environments helps to determine average performance characteristics and the sensitivity to site variations.

Field testing completes the performance engineering cycle. The data collected in this step not only gives final results for the new product, but if done correctly, provides the customer characterization needed at the beginning of the performance engineering cycle for future products.

Performance testing of the Series 70 cache in the field

was a major priority. Thirteen HP 3000 Series 68 sites were selected to participate in the alpha and beta test phases. All sites ran a series of HPSnapshot measurements. The characteristics include the number of users, percentage of CPU busy, the type of work done, and the amount of main memory. These were then compared to corresponding averages of a large set of customers. Of the thirteen sites, seven were picked to receive the hardware monitor.

The hardware monitors were installed on each site from two weeks to two months before installation of the Series 70, and for two weeks to one month after installation of the Series 70. The hardware monitors were able to separate the effects of the cache from the rest of the system. Data was recorded for 23.5 hours of each day, in half-hour samples. These baseline measurements were long enough and detailed enough to determine the variations caused by shift changes and weekends. In some cases monthly effects were noticed. Once started, the hardware monitors required human intervention just once a week, and then just to replace a flexible disc. The measurements had no effect on the systems under test.

A total of 10,500 half-hour samples were collected over the seven sites. This represents over 5 billion cycles. Of the total, over 3000 samples were from prime time, representing over 1.5 billion cycles.

The difference between the Series 68 and 70 caches was calculated for several variables for each site. Table II summarizes the results.

The calculations treat each site as one aggregate sample. The 90% confidence interval is a two-tailed t-test with six degrees of freedom. Statistically, it represents the range where the population mean is expected to be.

Table II
Prime-Time Field Measurements




Hardware Metric	Series 68	Series 70	Difference	90% Confidence Interval of Difference
Hit Rate	92.6%	98.7%	6.1%	±0.35%
Frozen Cycles	29.5%	8.5%	21.0%	±0.96%
Paused	34.4%	48.5%	14.1%	±7.92%

The cache hit rate improvement is consistent over the seven sites. Every site experienced significant improvement. The improvement is further confirmed by the decrease in frozen cycles. Again, there was consistent improvement over all the sites. It is expected that the average improvement over all customers will lie in the range of 20.0% to 22.0% cycles recovered during prime time.

Fig. 5 shows the progressive estimates of the difference in hit rate and frozen cycle percentages. Each point is the best estimate for the mean difference over all customers. The box surrounding each point indicates the 90% confidence range. Note that the hit rate confidence intervals were very narrow, indicating uniformity in the measurements. However, the effects of workload differences are apparent in the means actually measured on customer systems. The confidence interval for the frozen cycle difference progressively narrowed during the project. The added information at each step helped to define more clearly the performance of the cache. The final mean is within the

HIT RATE AND FROZEN CYCLE DIFFERENCES

BETWEEN SERIES 68 AND 70 CACHE

HIT RATE DIFFERENCE  FROZEN DIFFERENCE  90% CONFIDENCE 

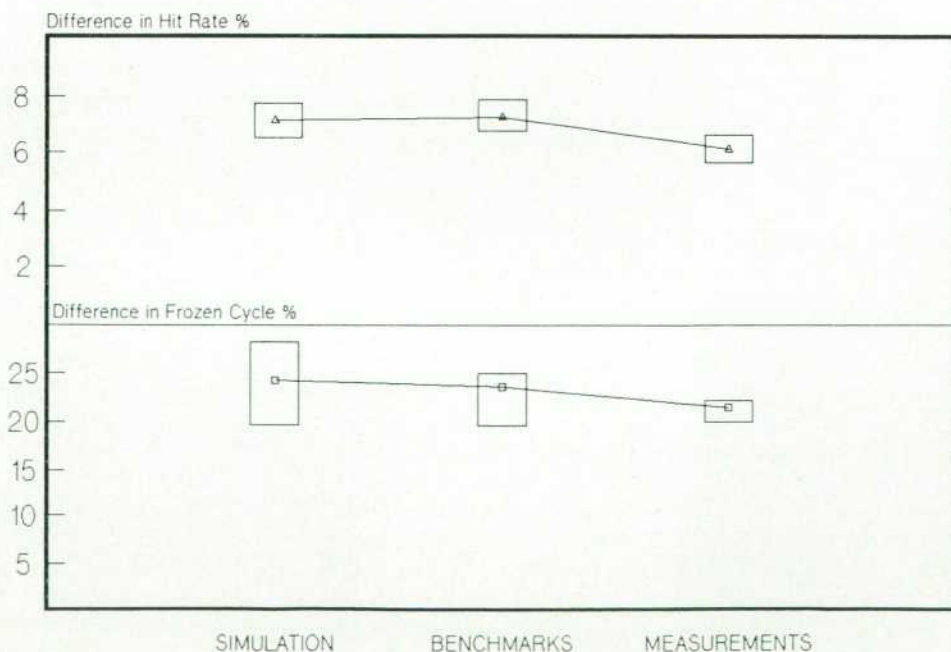


Fig. 5. Differences between Series 68 and Series 70 cache hit rates and frozen cycle percentages.

original confidence interval, thus validating the estimation methodology.

CPU pause time is often used as a measure of improvement. It is included here for comparison with the other metrics. While the Series 70 cache can add significant CPU capacity to the system, CPU paused is not a very stable statistic in most customer environments.

The Series 70 cache not only exhibited the expected average performance improvement over all sites, but was found to be fairly insensitive to variations in workload and configuration. This can be seen in the distributions of the hit rates and frozen cycles in Figs. 6 and 7.

The distributions of hit rates for the Series 68 and 70 are shown in Fig. 6. Not only has the Series 70 moved the distribution significantly higher, it has also narrowed the distribution, which is evidence of the saturation effect. The saturation effect reduces the sensitivity of the cache to different operating conditions.

The distribution of frozen cycles also shows a significant change in the distribution (see Fig. 7). The difference in the distributions is the measure of performance improvement. The fact that the overlap of the distributions is very small emphasizes that the performance improvement applies across all sites.

HPSnapshot studies and customer observations confirm the predicted performance improvement for the entire Series 70 product. The data collected is now being used as part of customer characterization for future members of the HP 3000 product family.

Summary

The HP 3000 Series 70 is the result of precise measure-

ments applied to an existing system. The current customer environment was extensively characterized and analyzed for performance opportunities. Potential enhancements were modeled and a set of enhancements was selected that best met the criteria of higher performance, lower cost, and short development time. The system design and the design of each component were analyzed and tracked through development. Benchmark tests were run on the prototypes. Finally, the product was measured over several months in the field to confirm the performance. The measurements also serve as input for future products.

The Series 70 cache memory subsystem, as the major performance component, was a direct result of careful measurements and analysis. The performance analysis has also advanced the state of the art in cache measurement and prediction.⁶

Acknowledgments

The success of the HP 3000 Series 70 is the result of extra effort by many people. Sawsan Ghanem and Ben Wang did extensive analytical modeling and system measurements. Gordon Matheson provided hardware measurement and valuable I/O analysis expertise, and Mike Curry and Kevin McCarthy debugged the initial data collection hardware and collected the simulator input data. Dan Homan, David Kennedy, and students Jennifer Gill and John Mount conducted the field measurements and analysis. Many thanks to Ardis Gage, Mark Kroot, and Chris Shaker for their innumerable weekends spent on the cache hardware development and debug, to Ken Foot, Norm Galassi, G.A. Gordon, Skip LaPetra, Woody Woodward,

SERIES 68 VS 70 HIT RATE IN PRIME TIME

ALPHA & BETA SITES

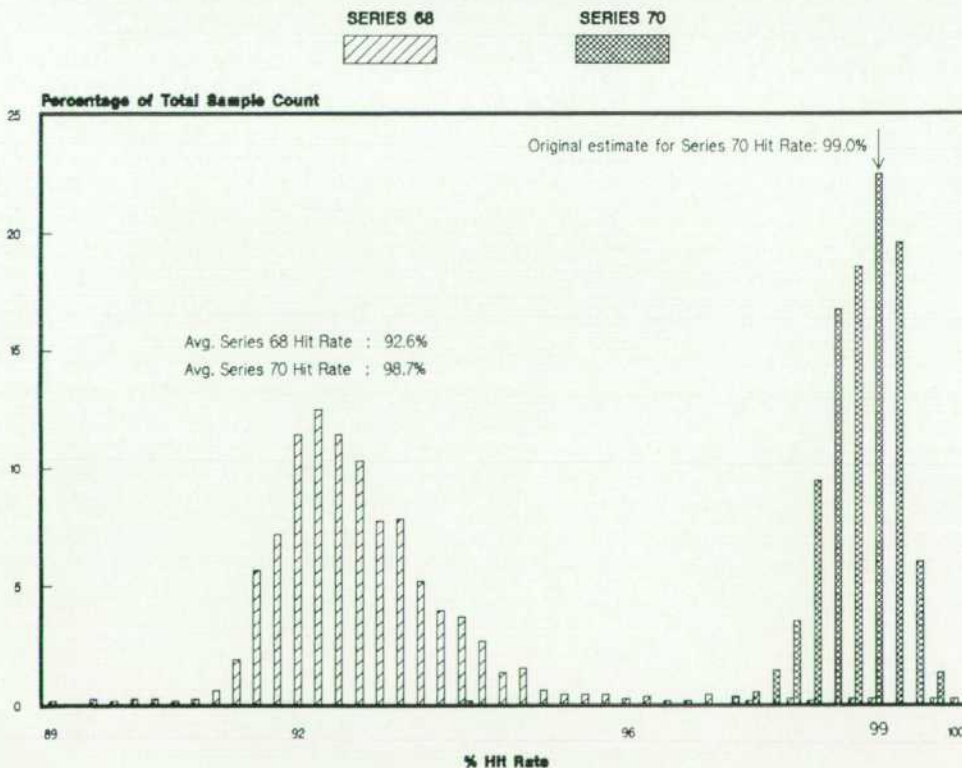


Fig. 6. Series 68 hit rates versus Series 70 hit rates.

SERIES 68 VS 70 FROZEN CYCLES IN PRIME TIME

ALPHA & BETA SITES

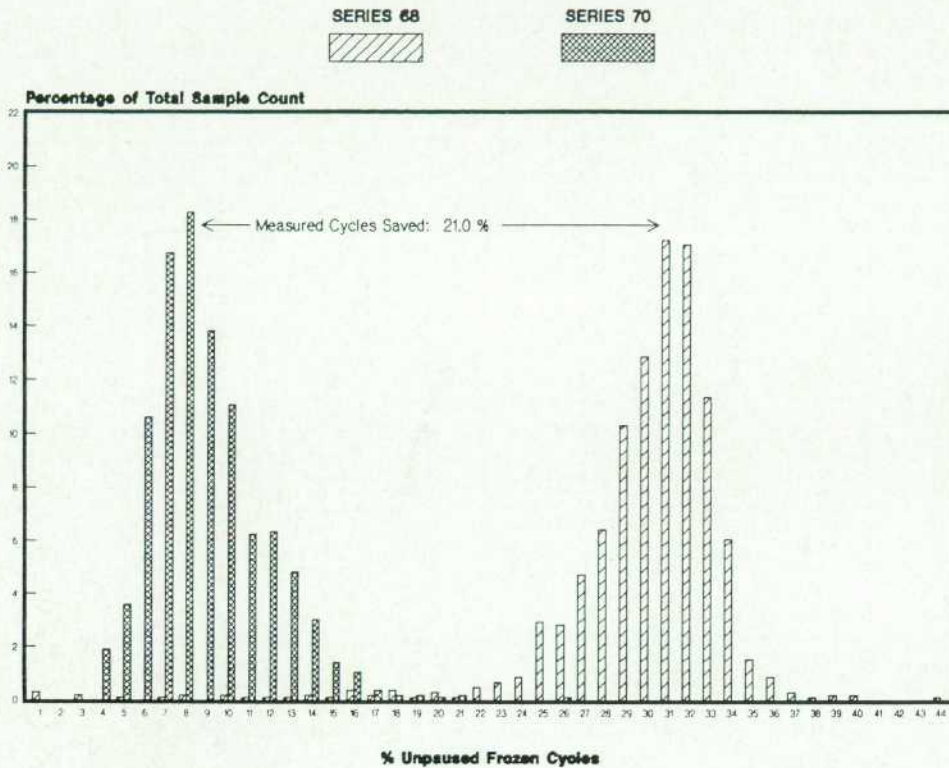


Fig. 7. Series 68 frozen cycles versus Series 70 frozen cycles.

and student Lenny Rosenthal, who provided tools and logistics support for the hardware design effort, and to Ken Kimi and his microcode development team, who provided microcode enhancements and tools for the cache hardware. Thanks also to Tom Henneman and the performance measurement and test center for the many extra hours of benchmark work, and to the MPE development lab for their assistance during the debugging of the cache hardware. Finally, special thanks to Terry Jackson, performance analysis project manager, and to Jim Holl, high-performance hardware section manager, under whose leadership and guidance the Series 70 was born and became a reality.

References

1. A.J. Smith, "Cache Evaluation and the Impact of Workload Choice," Report UCB/CSD85/229, March 1985, *Proceedings of the 12th International Symposium on Computer Architectures*, June 1985, pp. 64-75.
2. J.A. Lukes, "Hewlett-Packard Precision Architecture Performance Analysis," *Hewlett-Packard Journal*, Vol. 37, no. 8, August 1986.
3. D.L. Wolpert, "HP-IB Command Library for MS-DOS Systems," *Hewlett-Packard Journal*, Vol. 37, no. 5, May 1986.
4. C.H. Sauer and K.M. Chandy, *Computer Systems Performance Modeling*, Prentice-Hall, 1981.
5. D. Ferrari, G. Serazzi, and A. Zeigner, *Measurement and Tuning of Computer Systems*, Prentice-Hall, 1983.
6. A.J. Smith, "Bibliography and Readings on CPU Cache Memories and Related Topics," *Computer Architecture News*, Vol. 14, no. 1, January 1986.

Hewlett-Packard Company, 3200 Hillview Avenue, Palo Alto, California 94304

HEWLETT-PACKARD JOURNAL

January 1987 Volume 38 • Number 1

Technical Information from the Laboratories of Hewlett-Packard Company

Hewlett-Packard Company, 3200 Hillview Avenue
Palo Alto, California 94304 U.S.A.
Hewlett-Packard Central Mailing Department
P.O. Box 529, Startbaan 16
1180 AM Amstelveen, The Netherlands
Yokogawa-Hewlett-Packard Ltd., Suginami-Ku Tokyo 168 Japan
Hewlett-Packard (Canada) Ltd.
6877 Goreway Drive, Mississauga, Ontario L4V 1M8 Canada

Bulk Rate
U.S. Postage
Paid
Hewlett-Packard
Company

CHANGE OF ADDRESS: To subscribe, change your address, or delete your name from our mailing list, send your request to Hewlett-Packard Journal, 3200 Hillview Avenue, Palo Alto, CA 94304 U.S.A. Include your old address label, if any. Allow 60 days.